Pervasive and Mobile Computing 56 (2019) 1–17

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

A test methodology for evaluating architectural delays of LoRaWAN implementations

Dhiego Fernandes Carvalho [a], Paolo Ferrari [a,*], Emiliano Sisinni [a], Alessandro Depari [a], Stefano Rinaldi [a], Marco Pasetti [a], Diego Silva [b]

[a] Department of Information Engineering, University of Brescia, Brescia, Italy
[b] Federal University of Rio Grande do Norte, Natal - Rio Grande do Norte, Brazil

HIGHLIGHTS

- Analysis of LoRaWAN network for optimization/comparison of different solutions.
- Identification of logical entities in the LoRaWAN network reference model.
- Identification of the different traffic flows among LoRaWAN backend services.
- Definition of metrics for assessing LoRaWAN implementation performance.
- Demonstration of feasibility and effectiveness in real-world use cases.

ARTICLE INFO

Article history:
Received 28 November 2018
Received in revised form 21 January 2019
Accepted 19 March 2019
Available online 21 March 2019

Keywords:
LoRaWAN
LPWAN
IoT
Network delay
Cloud
Sensor network

ABSTRACT

The Low Power Wide Area Networks (LPWANs) have been proposed as viable wireless connection method for the implementation of the Internet of Things (IoT), thanks to their wide coverage, low complexity and low power consumption. In this scenario, LoRaWAN emerged as a de-facto leading technology, because of its openness and the current availability of many devices (like sensors and gateways). Several implementations of the LoRaWAN specifications have appeared in the market, each one showing support for different hardware, different software architectures, and different operating systems. Thus, the need of test procedures for comparing them is evident. In this paper, a test methodology for the experimental assessment of architectural delays of LoRaWAN implementations is presented. The new approach is composed of: a mapping phase, needed for highlighting the LoRaWAN blocks inside the implementation under test; followed by a measurement phase, in which the relevant timestamps are taken along the information path from LoRaWAN node to LoRaWAN customer application. The use and the effectiveness of the proposed methodology are shown by means of use cases that involve different LoRaWAN implementations and several hardware platforms. The experiments described in the paper are not intended to evaluate particular implementations but they are aimed to assess the suitability of the proposed methodology. The results demonstrate that the proposed approach can be used to compare the performance of entire LoRaWAN systems, helping the owner of the infrastructure and the user in making choices and optimizations.

* Corresponding author.
E-mail address: paolo.ferrari@unibs.it (P. Ferrari).

https://doi.org/10.1016/j.pmcj.2019.03.002
1574-1192/

1. Introduction

Today, the availability at low cost of system-on-chip with enhanced performance and embedded connectivity boosts the era of connected smart objects to build the Internet of Things (IoT). The smart objects will soon change the way we live and work since their number will rise to billions in few years. Currently, the communication infrastructure is the Internet. However, many smart objects are not directly connected to the Internet but gateways are used to balance the costs. Last, if flexibility and mobility are required, wireless technologies are applied preferring low-cost and low-power consumption.

Among the various wireless technologies used to bridge the gap between sensors and gateways to the Internet, there are solutions inspired in the cellular communication approach. In these cases, the wireless end-devices are linked to a base station, which defines a one-hop cell; the base stations are connected together with a backbone network (wired or wireless). However, despite current cellular network infrastructure technology (4G) could potentially address many IoT scenarios, the complex architecture and the demanding protocol stack result in high computational capability of the end-nodes, which in turn implies high-cost and high power consumption, something not compatible with most of actual applications. Only the next generation technology (5G) promises a mass deployment in the future. Therefore, other technologies are used today in this field: the Low Power Wide Area Networks (LPWANs) [1–4]. Their main advantages are the wide area coverage and the reduced number of base stations, thanks to: the innovative radios that trade off the sensitivity with the throughput; the simplified communication protocol stack that, in turn, reduces the infrastructure complexity. Consequently, both the power consumption and the cost are diminished [5,6]. Currently, the consumer market, the industry, and the academic world are proposing/using several LPWANs; the 3rd generation partnership project (3GPP) itself standardized the so called "Narrowband-IoT" or NB-IoT in LTE release 13, focusing on machine-to-machine, delay-tolerant communications. Nevertheless, the use of licensed bands affects costs as well and makes device management cumbersome (e.g. due to SIM cards). At this time, most of the adopter consensus is achieved by LoRaWAN, that has been turned in a de-facto standard. It is an open standard defined by the LoRa Alliance, that now is implemented in many commercially available devices from multiple manufacturers. The LoRaWAN wide success is demonstrated by both the constantly increasing applications in various scenarios, and the huge available scientific literature.

Year after year, many implementations of the LoRaWAN specifications have appeared in the market, each one showing support for different hardware, for different backend software architectures, and for different operating systems. When many implementations are offered to network managers and end users, the need of test procedures for comparing them arises.

The scientific focus on LoRaWAN is on various aspects. There are papers providing discussions about theoretical limits imposed by the protocol, for instance medium access policies or duty cycle restrictions imposed by regulatory bodies [7–10]. There is an analytical model of the LoRaWAN uplink (UL) performance, in terms of collision rate and latency, under the influence of regulatory and aggregated duty cycling [11], with the hypothesis of exponential inter-arrival times. Finally, some works tried to experimentally evaluate communication performance [12–14]. The impact of quasi-orthogonal chirp sequence has been investigated as well [15]. The authors of this paper are active in the LoRaWAN area with works evaluating performance of the LoRaWAN wireless link [16–22].

In general, the most requested metrics for Quality of Service (QoS) evaluation in any business-critical applications are network latency and, especially, application response time. Nevertheless, end-to-end delay measurements and estimation technique for generic LoRaWAN implementations have not been presented in literature. In particular, discussion of appropriate methodologies, that can significantly improve quality, accuracy, and representativeness of measurements, is completely missing. What is an appropriate definition for LoRaWAN QoS metrics? How should each metric be measured? These questions currently have no answer.

This paper tries to solve this gap, expanding the ideas presented in [23]. The central observation is that LoRaWAN specifications include a network reference model that all the compliant solutions must follow. Accordingly, some probing points for timestamping can be identified for any LoRaWAN application, independently of peculiarities of actual implementation. In particular, original contributions of the paper are:

- Identification of logical entities comprised in the LoRaWAN network reference model, highlighting the communication "boundaries" (i.e. interfaces) among them;
- Identification of the different traffic flows existing between endpoints, focusing on the LoRaWAN backend services other than the wireless connectivity;
- Proposal of "universal" probing points for timestamping, and definition of metrics able to identify LoRaWAN network delays and to evaluate LoRaWAN implementation performance;
- Demonstration of the feasibility and the effectiveness of the proposed approach in real-world use cases leveraging on reference LoRaWAN implementations.

It has to be noticed that the proposed methodology is mainly aimed at evaluating the LoRaWAN QoS from the offered service point of view. Since backbone/backhaul network delays and backend server delays are "tunable" by choosing different implementation details (transport medium, bandwidth, etc…), the proposed methodology can help a service provider to easily compare and tradeoff features of different implementations.

The paper organization is the following: in Section 2 an overview of LPWANs is given; in Section 3 the network reference model of LoRaWAN is detailed and the proposed methodology is resumed; in Section 4, two real-world Use Cases are identified and described; in Section 5, some experimental results obtained with the proposed setup are discussed. Finally, the conclusions are drawn.

2. IoT wireless networks: the LPWAN approach

In this section, a LPWAN general overview is given, and the LoRa solution is proposed as the most significant example of LPWAN radio technology currently available. In details, after a general description of the two lowest protocol layers (the physical layer PHY and the data link layer DL) characteristics, the focus will return to the peculiarity of the LoRa radio developed by Semtech.

2.1. LPWAN technologies

The availability of wide coverage, the scalability, the low-cost, the low-power consumption of some wireless communication solutions are the key factor to make real and affordable many IoT applications. LPWAN has been proved to be the answers to these requests, if a relatively low update rate can be accepted. Generally speaking, technologies belonging to this class of wireless networks are characterized by:

- Operations in the different parts of the spectrum alternative to the over-crowded 2.4 GHz region;
- Limited data rate, with small channel bandwidth, efficient modulation schema (for instance based on spread spectrum techniques), and a very good receiver sensitivity (allowing a link budget on the order of 150 dB), and;
- Lightweight protocol stack and low complexity communication infrastructure, in order to minimize the power consumption and the cost.

The network is organized as a single-hop, star topology and medium access is done with random mechanisms like ALOHA [24]; the fulfillment of Regional regulations is usually accomplished using "listen-before-talk" and duty-cycle based strategies. With the aim of further reducing the costs of deployment and maintenance, the network complexity is mainly in the backend system while the end-devices are kept simple. The backend servers manage the network by assigning resources with the goal of satisfying individual application requirements. Several very different technologies have been grouped under the LPWAN umbrella [1]. A very rough classification distinguishes between solutions dominated by technologies operating in license-free bands (typically using proprietary physical layer like the Ultra Narrow Band by SIGFOX, the LoRa by Semtech, and the Weightless led by Neul) versus technologies operating in licensed portions of the spectrum (most notably, the Narrow-Band IoT proposed in the 3GPP Long Term Evolution standard).

Additionally, it has to be remarked that in the last years the development and first experimental deployments of the 5th generation of mobile networks (5G) are happening. The main goal of 5G is to deliver huge capacity and scalability with respect to 4G mobile solutions. Consequently, the LPWANs and 5G solutions may operate together, with 5G as the backbone connectivity of LPWAN. In some papers, such possible integration has been discussed [25,26].

However, today, among all the recently appeared LPWAN technologies, the most interesting is still LoRaWAN that uses LoRa at lower levels. As a matter of fact, many manufacturers offer LoRa based commercial devices and modules, resulting in a large acceptance by both market and academic world. For the same reason, very different application scenarios have been considered as well, ranging from precision agriculture [27] to electric vehicles management [19,28] and, finally, industry [18,29].

2.2. The LoRa approach to spread spectrum

The LoRa radio has been patented by Semtech, and it is used in many communication stacks (including the LoRaWAN solutions) [30]. LoRa is based on Chirp Spread Spectrum (CSS) modulation that codes, with a single chirp frequency trajectory (i.e. a symbol), SF (spreading factor) bits. The choice of CSS has been done because the chirp symbols have good correlation properties and, changing the SF factor, virtual channels or adaptive data rate strategy can be achieved. In details, the chirp bandwidth BW is fixed (BW = 125, 250 or 500 kHz), while the chirp duration can be calculated as $T_C = \frac{2^{SF}}{BW}$. Accordingly, a higher SF means a lower data rate but a better noise immunity (thanks to additional processing gain). There is a Forward Error Correction mechanism that further increases noise and interference immunity, but in turn, it affects (reduces) the actual throughput; however, several coding rate can be specified in the range from CR=4/5 to CR=4/8.

It should be noted that the sub-GHz band must respect Regional regulations, including additional limitations on duty-cycle (that is the ratio between the duration of a transmission and time between transmissions). In Europe, for instance, the duty-cycle in the 868 MHz band is (just) d=1% on all the three mandatory channels.

As other LPWAN solutions, LoRa may be connected to medium access control layer of several stacks, even if simple approach is usually preferred (e.g. ALOHA or clear channel assessment for limiting collisions in dense environment).

Concluding, it has to be stressed that, being the LoRa technology one of the most affordable technologies for implementing IoT applications requiring long range wireless connectivity, many literature works exist assessing its

performance and its limitations, mainly imputable to the relatively poor throughput and bandwidth availability [7–15,22]. A tradeoff between the number of sensors deployed in the field, the chosen spreading factor and the transmission rate exists. Indeed, for all those applications requiring one packet per day transmission, millions of sensors are manageable; on the contrary, a refresh rate on the order of one minute lowers the number of possible nodes to thousands [17].

3. The proposed methodology for evaluating LoRaWAN solutions

A complete communication solution has been designed around the LoRa physical layer, named LoRaWAN. It has been designed in order to fulfill typical requirements of IoT scenarios leveraging autonomous (e.g. battery-powered) end-devices that may be either mobile or mounted at a fixed location and require relatively long refresh rate.

LoRaWAN specifications provide additional restrictions on the LoRa radio depending on the so called Regional Parameters. In Europe the allowed bandwidth is BW $\in$ [125,250] kHz and SF $\in$ [7..12]. Thus, each physical channel can host up to six quasi-orthogonal virtual channels (one per SF value), with data rates ranging from about 300 bps to 11 kbps. The effect of local clock drift in the nodes is limited keeping the message length in the range from 51 byte (at SF=12) to 242 byte (at SF=7).

In this section the architecture of a general LoRaWAN solution is described and, starting from the resulting model, the proposed approach for assessing architecture performance is discussed.

3.1. The LoRaWAN architecture

LoRaWAN specifications, which are publicly available, are managed by the so called LoRa alliance, that includes both device manufacturers, end users and research institutions. In particular, LoRaWAN completes the communication protocol stack defining the data link layer, that implements an ALOHA like medium access strategy. Regarding the network level, the addressed network topology is a hybrid wireless and wired star-of-stars, with the aim of minimizing as much as possible complexity (and thus implementation and maintenance costs) on the wireless side. Such an approach mimics the well-known topology of mobile communication networks, where handsets are wirelessly connected to a base station (thus forming a wireless star) and several base stations are interconnected to a base station controller using wired connections [31]. LoRaWAN specifies that a network comprises gateways (also known as concentrators or base stations) that relay end-device messages to the backend; interesting to note, each gateway contains a software (called "*packet forwarder*") that is able to forward messages using an implementation-specific protocol. It has to be highlighted that the gateway operates only at the physical layer and the frame payload is opaque.

In other words, two tiers exist in the network; one is based on wireless links to provide connectivity to end devices; the other is the backend, where network management is carried out in a centralized way and services to end-users are provided. The actual data exchange consists of: uplink, when the communication occurs from the end-device towards the user; and downlink, when the opposite occurs. According to LoRaWAN target applications, uplink is of main importance and concern, whereas downlink should be limited to increase efficient bandwidth utilization [32,33].

More in detail, the network reference model described in the LoRaWAN specifications [34] consists of two or three different kind of *servers* (depending on the version compatibility): the Network Server (NS); the Application Server (AS) and the Join Server (JS), as shown in Fig. 1. Note that the implementation details are out of the scope of the specifications, while only the operations to be carried out are described.

The NS is the logical entity that resembles the center of the star topology; the NS is in charge of checking proper frame format and authentication, providing acknowledgments if required. Additionally, the NS actually manages the LoRaWAN data link protocol features, e.g. as data rate adaptation strategy. The NS forwards uplink to the appropriate AS and queues downlink from any AS to deliver the useful payload to the proper end-device. If a JS is present, Join-request and Join-accept messages are forwarded to the end-devices and the JS by the NS, respectively. The JS (which is described since release 1.1 of the specs) is in charge of managing end-devices affiliation in a secure, i.e. encrypted, way. Finally, being the application level protocol not described in the specifications, the AS has to implement it for delivering data to the final user.

Roaming is supported as well; accordingly, three roles actually exist for a NS; the only "serving NS" (sNS) controls the data link layer behavior of the end-device; the "home NS" (hNS) is actually connected to the AS; finally, several additional "forwarding NS" (fNS) can be connected to other gateways. If the sNS does not change, passive roaming is carried out; on the contrary, when handover roaming is enabled, the end-device is managed by the visited network, though user data are still forwarded to the original hNS.

It has to be highlighted that the protocols for implementing the interfaces between the fNS and the gateway, and between the hNS and the AS are not in the specifications. Only the communications among NSs and JS–NS interfaces is described. In the latter case, it is required the adoption of the HTTP protocol, encoding the payloads using JSON. As a matter of fact, several different proprietary implementations may exist. Such an architecture makes it easy decoupling the owner of the infrastructure from the owner of the data, enabling new business scenarios.

For security purposes, the message payload is encrypted on a session base. End-device commissioning can occur in two ways; when over the air mode is used, the JS is involved for providing session keys and connecting the end-device with a hNS; otherwise, provisioned mode requires an out of band communication channel to provide the end-device with proper keys. Security on the backend is required as well, but implementation is left out to the implementer.

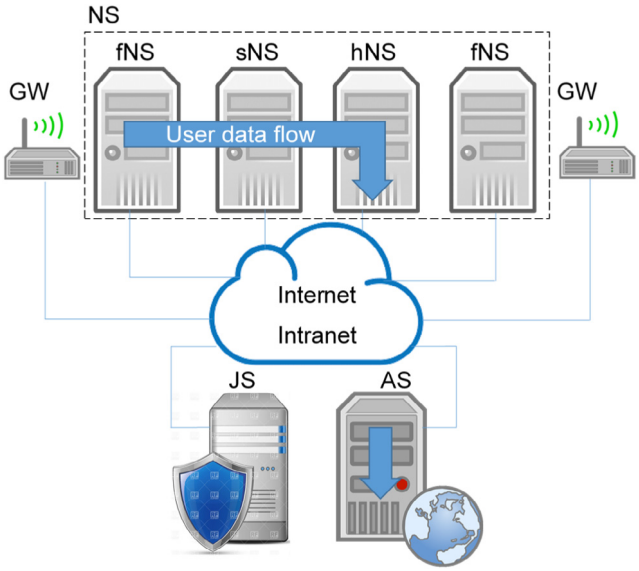


Fig. 1. The LoRaWAN architecture.

3.2. The proposed methodology for evaluating LoRaWAN networks

It is well known from the literature that the very simple wireless protocol adopted by LoRaWAN (exploiting reduced bandwidth, limited number of channels and ALOHA medium access mechanism with duty-cycle limitations) bounds end-to-end performance when very "dense" applications, in terms of both number of nodes and update interval needs, are considered [7–15]. On the other hand, from the previous discussion, it is evident that most of the complexity in a LoRaWAN resides in the backend; consequently, overall network performance is not only affected by the wireless tier, but by the backend as well. However, very few, if any, works exist in literature assessing the backend performance. Additionally, the loose description provided by the specifications (that only identifies the need of an Internet or Intranet as the backbone), allows for very different measurement setup that can make comparison a cumbersome task.

In this paper, authors try to fill this gap, providing a general methodology for evaluating not only the wireless tier, but also the backend tier. In particular, the previously described network entities (i.e. the servers) are considered as black boxes and the focus is pointed on the actual message exchange. In this way, the proposed approach does not need architecture modification or additional code probes; moreover, it is not limited by the details (e.g. the number of wireless nodes or the deployment of servers in the cloud) of the implementation under test.

As asserted before, uplink is of main concern for typical LoRaWAN networks and the focus of the test methodology is about uplink data flow characterization. In this case, end-user traffic is injected in the LoRaWAN network under test by means of a probing node and it is collected out of the AS in accordance with the actual AS implementation. A simplified diagram is shown in Fig. 2, where for sake of simplicity a single NS implementing all the network level functionalities is depicted; for the same reason, the (optional) JS, involved only in the key management, is not shown. In particular, in agreement with the LoRaWAN specifications, the uplink user data flow can be further split into the following sub-flows (it has to be noted that a similar approach can be used for evaluating the joining procedure as well):

- F0: from the end-user data source to the mote (data injection); it is out of the backend scope;
- F1: from the mote to the gateway; it is the only wireless portion of the overall data flow;
- F2: from the gateway to the NS;
- F3: from NS to the AS;
- F4: from the AS to the end-user data destination (data extraction).

Obviously, in order to evaluate time performance of implementation under test, time information must be added to each data flow; in turn the method to assign timestamps to events depends on the way the actual implementation is physically realized.

As a consequence, the proposed methodology has three main steps (two preparation steps and one execution step):

(1) Identification and block mapping: the real-world devices and software must be mapped on the corresponding backend entities described in the specification in order to identify boundaries between them and to highlight the ingress and egress ports of the information.

(2) Timestamping point configuration: once boundaries between functionalities have been found, the position and the method to collect the time at which the information crosses the boundary (timestamping) is determined. For each

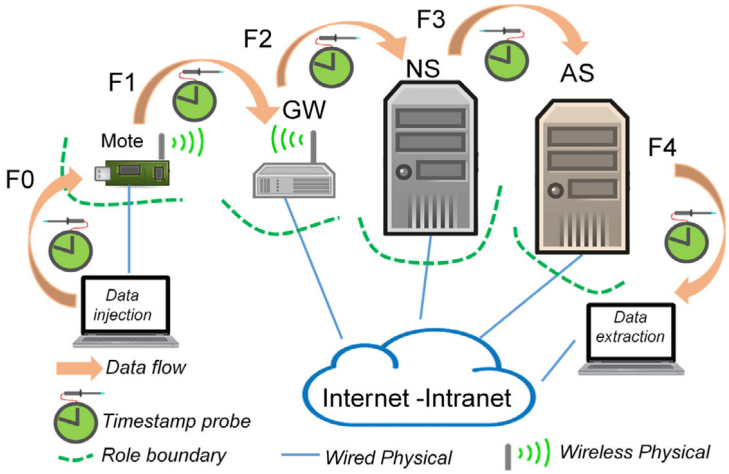


Fig. 2. The simplified diagram of the proposed methodology.

identified sub-flow, up to two timestamps can be taken (origin timestamp and arrival timestamp). Many recording and timestamping techniques may be needed, depending on the physical nature of the information flow (inter-process exchange, network exchange, wireless exchange, cloud exchange). Since backend tier implementation can consist of several different physical devices, possibly spread on a large geographical area, physical layer adaptation and time synchronization to guarantee temporal coherence between timestamping probes may be needed. However, this can be generally achieved, since most of commercial solutions rely on office-like connectivity leveraging on standard Ethernet (IEEE802.3) and/or WiFi (IEEE802.11) links.

(3) Measurements and analysis: the network under test is stimulated, and the information flow across it is monitored and measured. For instance, data are injected in a mote so that messages can be tracked and timestamped at all the timestamping points decided in the previous step. The final analysis is carried out calculating the architectural delays as differences between the timestamps relative to the same data flow. Many metrics can be defined depending on the focus of the analysis; with the proposed methodology, both communication delays through architecture blocks and communication delays due to network could be isolated and measured.

4. Application of the proposed methodology: two Use Cases

Currently, different implementations of LoRaWAN infrastructure exist; some of them are based on private backend servers and proprietary nodes that implements specific services [35–37], others, instead, offer free access for non-profit usage to an open-to-the-public infrastructure (but they pose some limitations to the available bandwidth) [38,39]. Although the generality of proposed methodology makes it suitable for very large and complex real-world deployments, creating Use Cases that involve such big infrastructure is not in the scope of this work. In order to demonstrate the applicability of the proposed methodology, without losing generality, this paper considers two freely available solutions that are also used as basis for many LoRaWAN networks.

Note that the two implementations used in this paper have not been optimized nor configured. They are simple reference implementations and this paper uses them just to demonstrate how to highlight differences between systems under tests.

4.1. Use Case 1: The Semtech reference implementation

The Semtech corporation invented the LoRa radio and also proposed the first implementation of the LoRAWAN stack in order to promote its LPWAN transceivers. For instance, a reference kit is available, allowing easily setup of small, private LoRaWAN networks for demonstration purposes [40], while the document "LoRaWAN Network Server Demonstration: High Level Description" [41] provides an overview of the reference architecture. Interesting to note, the Semtech packet forwarder quickly spread among developers, thus becoming a de facto standard solution, adopted in both commercial and academic pilot tests [42,43]. All the installation and configuration manuals are made for developers that are willing to use Semtech devices to build end-nodes and gateway. Not all the source code or the datasheets are publicly available, some of them are released by Semtech after a Non-Disclosure Agreement is signed. The code of this "reference implementation" has been ported on several platforms, including Javascript, without any relation with Semtech official releases. As a consequence, the probability that given a LoRaWAN device is using some parts of the original work done by Semtech, is quite high.

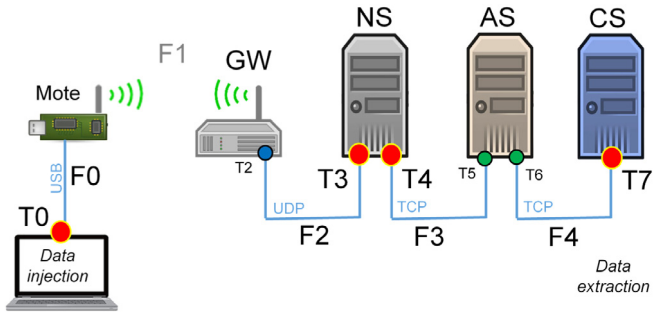


Fig. 3. The Semtech architecture.

The application of the first step of the methodology to the Semtech reference implementation reveals that it is strictly stuck to the LoRaWAN specification. The "actors" in this implementation are: the LoRaWAN end-device(s), the LoRaWAN gateway(s), the Network Server, and the Application server. The interactions within blocks are well defined and follow the path described in the specifications. Additionally, a Customer Server (CS) is defined as well, for allowing the end-user to access its own data.

There is direct communication between adjacent blocks; the coding scheme of this communication is defined in [44] and it is based on JSON objects encapsulated into IP/UDP or IP/TCP-based proprietary protocols. Fig. 3 shows the block diagram of the Semtech reference implementation and the path of an uplink message containing customer data.

The second step of the methodology requires the identification of the boundaries and the determination of the timestamping points for the exact calculation of the architectural delays, as in [45]. As discussed earlier, up to two timestamps can be assigned to each sub-flow. Thanks to the plain structure of the Semtech implementation, the placements of the timestamping points is straightforward (see Fig. 3):

- The data flow F0, out of the backend scope, is timestamped at T0 when the end user sends a message to trigger the LoRaWAN message transmission by the probing mote.
- The data flow F1, representing the wireless link, is timestamped at T1 capturing the over-the-air actual LoRaWAN user message; e.g. T1 could be recorded by the gateway when the packet arrives at its antenna, since the variation of the propagation time in air is negligible and nodes are located at known fixed position and propagation time can be compensated. Note that T1 is not collected in the considered experiments.
- The data flow F2 leaves the gateway at T2 and enters the NS at T3. The timestamps are taken looking at the network message containing the information about the received packet (coded with Semtech UDP/IP JSON format, thus allowing tracking of an individual message) with destination the NS.
- The data flow F3 leaves the NS at T4 and enters the AS at T5. The timestamps are taken looking at the network message containing the information (coded with Semtech TCP/IP JSON format, thus allowing tracking of an individual message) with destination the AS.
- The data flow F4 exits from the AS at T6 and arrives at the customer server at T7. Again, the timestamps are taken looking at the network message containing the information (coded with Semtech TCP/IP JSON format, thus allowing tracking of an individual message).

4.2. Use Case 2: the LoRa Server open source implementation

The LoRa Server is a LoRaWAN implementation which provides open-source components for building LoRaWAN networks. It is a ready-to-use solution that requires a low effort to have the system up and running. It includes a user-friendly web-interface, and REST API interface for integration with third party applications [46]. This implementation is used in several scientific works including [43].

The first step of the methodology was applied also to the LoRa Server implementation, highlighting some major architecture changes with respect to the LoRaWAN architecture as described in the specification. This does not mean that the LoRa Server implementation is not compliant with LoRaWAN; simply, the functionalities are spread in their own way in the system, or implemented with different techniques. From the documentation, the actors in this implementation are: the LoRaWAN end-device(s), the LoRa gateway (and the LoRa Gateway Bridge), the LoRa Server, the LoRa App Server, the Application and MQTT Broker.

The main innovation of this architecture, shown in Fig. 4, is the use of a central MQTT Broker that acts as a central point through which all the information exchanges pass. This allows to make the system modular, separating the various services in a better way. Differently from the Semtech reference implementation, the LoRa Server gateways include a packet forwarder exchanging MQTT messages with the Broker. In this way, it is required only the knowledge about the

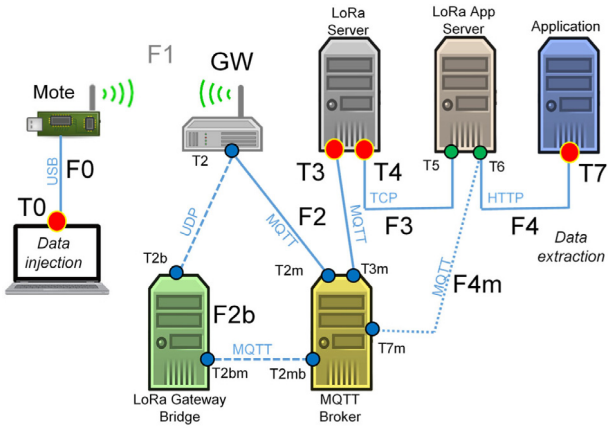


Fig. 4. The LoRaIO architecture.

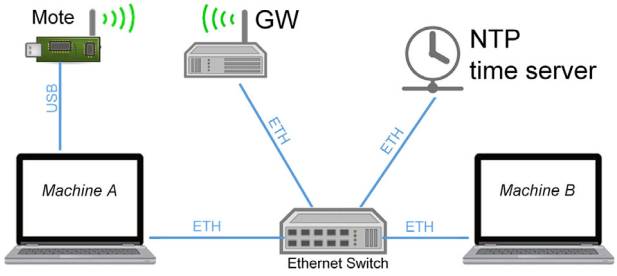


Fig. 5. Hardware architecture for the experiments.

corresponding MQTT topic in order to talk with the gateway (simpler and flexible configuration). Interesting to note, the LoRa Gateway Bridge has been proposed as well, i.e. a (software) component that transforms the IP/UDP protocol of the Semtech packet forwarder into messages over MQTT for the MQTT Broker. The LoRa Server component has the functionalities of the LoRaWAN Network Server component, while the LoRa App Server component has the functionalities of the LoRaWAN Application Server. Last, the Application is intended as the customer application that is using the end-device data.

In the second step of the proposed methodology, the identification of the boundaries and the determination of the timestamping points is done. The placement of the timestamping points (up to two for each sub-flow) is shown in Fig. 4, and listed in the following:

- The data flows F0 and F1 are timestamped the same way of the Semtech reference implementation.
- The data flow F2 leaves the gateway at T2 and enters the NS at T3 as for the Semtech reference implementation. However, actual data exchange occurs via the MQTT Broker, but its presence is transparent from the LoRaWAN backend point of view and the related processing time is included in the F2 data flow. Additionally, if the LoRa Gateway Bridge is considered, an intermediate step occurs. The timestamps T2 and T2b, taken looking at the UDP/IP transaction, refer to the egress time from the gateway and the ingress into the LoRa Gateway Bridge, respectively. LoRa Gateway Bridge egress timestamps is T2bm. The presence of the MQTT Broker allows to further identify timestamps T2 mb, T2m and T3m.
- The data flow F3 leaves the LoRa Server at T4 and enters the AS (called LoRa App Server) at T5. The timestamps are taken considering the TCP messages exchanged by servers.
- The data flow F4 leaves the LoRa App Server at T6 and arrives at the Application at T7. Depending on the selected type of integration (leveraging the MQTT Broker or the REST interface), timestamping refers to the HTTP POST message or MQTT Publish message (T7m)

5. Experimental setup and results

Real-world experiments on the two previously described Use Cases have been carried out using a local network infrastructure (implemented by means of an Ethernet 100BaseT switch, as shown in Fig. 5), without generality loss; the

same experiments could be carried out if the backend was installed in the cloud. In particular, two different configurations have been considered: in the former, representing a possible small private LoRaWAN network, the complete backend is executed on a single machine; in the latter, the backend components are spread over two different machine, mimicking a larger public network where the NS is managed by the infrastructure owner, while the AS is handled by a different data/service provider. Both the machines are equal; they host 4 GB of main memory and rely on a AMD Athlon 64X2 5600 @ 2.9 GHz. Regarding the operating system, it is different for the two considered Use Cases: the machines run Windows 7 Professional for the Semtech reference implementation and Linux Debian 9 for the LoRa Server open source implementation.

It has to be highlighted that the aim of the experiments is to verify the feasibility and flexibility of the proposed methodology. The considered configurations are not exhaustive, but represents possible real-world scenarios. The testbeds are not designed to stress the network infrastructure and evaluate performance in extreme conditions, but are intended to confirm the applicability of the proposed approach in real, possible very different, deployments.

More in detail, the LoRaWAN traffic is injected via a Microchip RN2483 LoRaWAN mote hosting a Semtech SX1276 transceiver and a Microchip microcontroller PIC18LF46K22. The RN2483 is a LoRaWAN 1.0 compliant modem that supports a virtual serial link over USB. The whole end device communication stack in managed by the modem, which, once properly configured, indefinitely listen for incoming user data to be transmitted. For sake of completeness, 18 B and 113 B long messages are sent by the mote, including the 13 B long LoRaWAN header. In both cases, messages are transmitted every 30 s using BW = 125 kHz, SF = 7 and CR = 4/5, on one of the default channel (centered around 868.3 MHz), having a duty cycle limitation d = 1% (according to EU limitations).

The LoRaWAN gateway is implemented by a Sentrius RG186; it hosts a pair of Semtech SX1257 analog frontend chips and the SX1301 baseband chip. The radio section is managed by a Linux-based WB50NBT module leveraging on an Atmel Cortex A5 @ 536 MHz. It natively supports different packet forwarders, including those compatible with the Semtech and the Lora Server implementation. Additionally, only for Use Case 2, we performed experiments installing a LoRa Gateway Bridge on the same machine running the LoRa Server.

Timestamps have been collected using the Wireshark protocol analyzer, which uses time information from the local machine operating system kernel. The USBPcap software in Windows and USBmon in Linux are used for monitoring the USB link to the LoRaWAN mote. Regarding the Ethernet network connection, when the one-machine scenario is considered, timestamps refer to the internal loopback interface (RawCap software is installed for the Windows machine). On the contrary, when two-machine configuration is considered, timestamps refer to the network interface card.

It has to be highlighted that in the one-machine configuration, time synchronization is not an issue since all the timestamps natively refer to the same clock reference. On the contrary, the two-machine configuration is an example of a distributed system and an adequate synchronization strategy must be considered. Two possible scenarios have been identified:

1. the Wireshark is installed in one machine only and all the timestamps refer to the entering/exiting traffic, mimicking the acquisition strategy of the one-machine configuration.
2. NTP synchronization strategy is exploited and each machine has its own Wireshark instance. In the considered scenario the NTP is installed in the local network at a single hop distance from the two-machines (Fig. 5). The quality of the synchronization has been evaluated considering the residual offset estimated by the NTP daemon [47,48]. During the experiments, in the worst case the mean offset value with the NTP time server was 0.4 ms, with a standard deviation of 0.2 ms.

An overview of the different setup configurations considered in this paper is provided in Table 1.

5.1. Definition of LoRaWAN metrics

As previously stated, both the single and the two-machine configurations allow the acquisition of T0, T3, T4 and T7 timestamps, that are highlighted in RED in Figs. 3 and 4. In this way, four metrics can be evaluated:

- D_{IN} = T3-T0, related to the time needed by a new uplink data to move along the F0, F1 and F2 data flows (including the mote and the gateway processing time);
- D_{NS} = T4-T3, related to the time needed by the NS to process the end-device data;
- D_{BE} = T7-T3, related to the time needed by a new uplink data to traverse the backend moving along the F3 and F4 data flows (including the NS and AS processing times);
- D_{LW} = T7-T0, related to the time needed by the data to traverse the entire LoRaWAN system.

Additionally, for the two-machine configuration, it is possible to consider timestamps T5 and T6 as well (highlighted in GREEN in Figs. 3 and 4), thus allowing the computation of the additional metric:

- D_{AS} = T6-T5, related to the time needed by the AS to process the end-device data.

Table 1
Hardware and Software configurations for the experiments regarding Use Case 1 and Use Case 2.

Configuration		One-machine	Two-machine		Two-machines + NTP	
Hardware setup (both Use Case 1 and Use Case 2)		Mote GW Switch –	Mote GW Switch –		Mote GW Switch NTP time server	
Software setup	Use Case 1	*Machine A* NS AS CS Wireshark USBpcap RawCap –	*Machine A* NS – CS Wireshark USBpcap – –	*Machine B* – AS – – – – –	*Machine A* NS – CS Wireshark USBpcap – NTP client	*Machine B* – AS – Wireshark – – NTP client
	Use Case 2	*Machine A* Lora Server LoRa App Server Application LoRa Gateway Bridge MQTT Broker Wireshark USBmon –	*Machine A* Lora Server – Application LoRa Gateway Bridge MQTT Broker Wireshark USBmon –	*Machine B* – LoRa App Server – – – – – –	*Machine A* Lora Server – Application LoRa Gateway Bridge MQTT Broker Wireshark USBmon NTP client	*Machine B* – LoRa App Server – – – Wireshark – NTP client

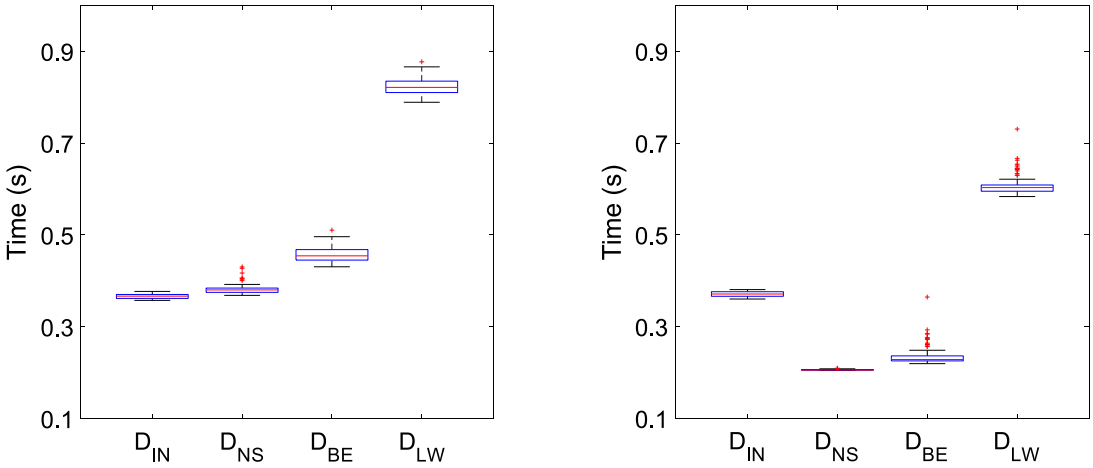


Fig. 6. Boxplots of the considered metrics for Use Case 1 (left) and Use Case 2 (architecture with full MQTT approach) (right) when the one-machine configuration is considered and an information of 100 bytes is traversing the LoRaWAN network.

5.2. Evaluation of LoRaWAN metrics

The evaluation has been carried out estimating the previously described metrics over 200 experiments. In Figs. 6 and 7, boxplots are used for resuming results of the one-machine configuration for the Use Case 1 (Semtech reference implementation) and the Use Case 2 (LoRa Server implementation), when the dimension of the data varies from 100 bytes to 5 bytes. In Figs. 8 and 9, the results related to the two-machine configuration are proposed.

Thanks to the proposed methodology, it is possible to highlight that the metric D_{IN} is almost the same in both Use Cases, and it depends primarily on the dimension of the data to be exchanged. This result was expected since the hardware for the wireless transmission and reception is the same in the two implementations. It has to be highlighted that the use of MQTT for exchanging data from gateway and Lora Network Server in the Use Case 2 implementation does not affect the D_{IN} metric.

The general performance of the two Use Cases slightly decreases when two-machine configuration is considered because of the extra time (and greater variability) introduced by the network connection between the machines.

A more detailed overview of the performance of the two Use Cases is available in Tables 2 and 3, where average and standard deviation values of the considered metrics are reported. It is evident that the Use Case 2 implementation seems faster in the conditions set by the experiments; however, these results are obtained under specific constrains in order to show the proposed methodology at work.

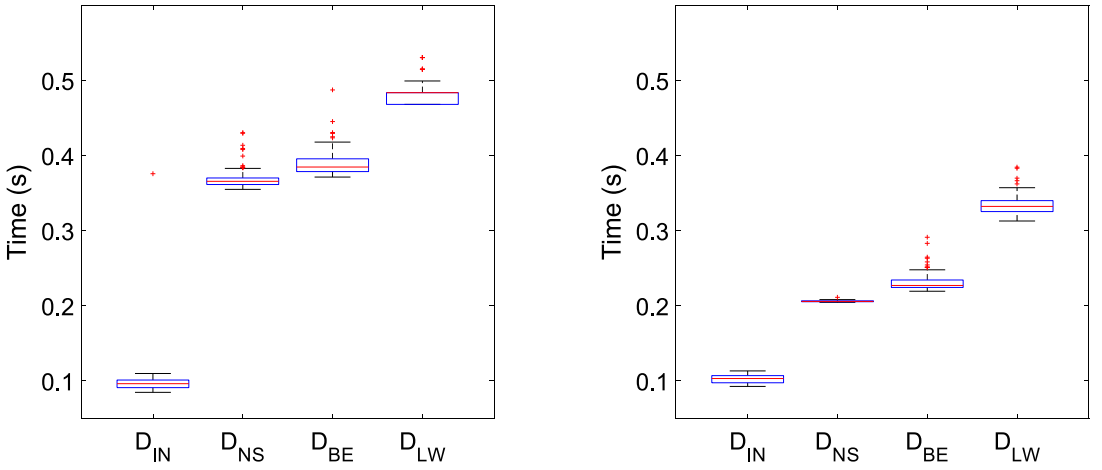


Fig. 7. Boxplots of the considered metrics for Use Case 1 (left) and Use Case 2 (architecture with full MQTT approach) (right) when the one-machine configuration is considered and an information of 5 bytes is traversing the LoRaWAN network.

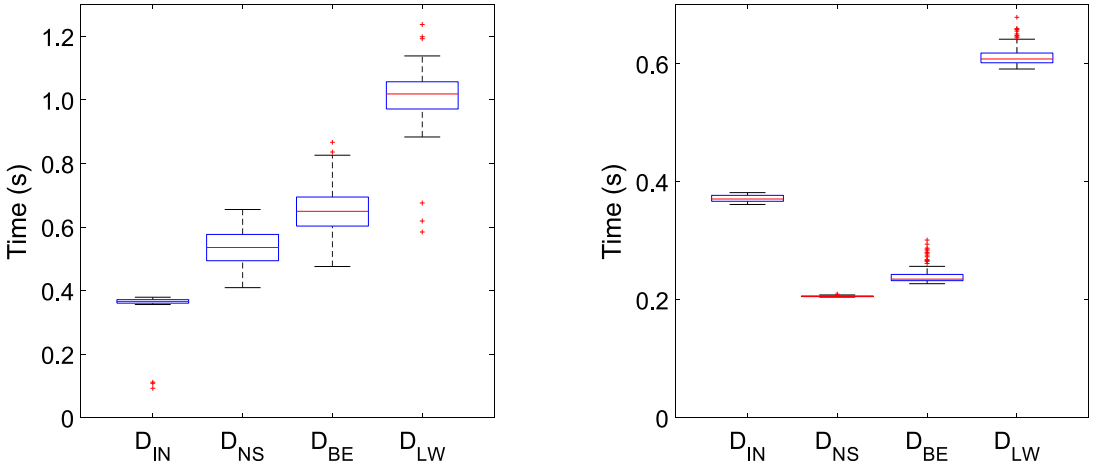


Fig. 8. Boxplots of the considered metrics for Use Case 1 (left) and Use Case 2 (architecture with full MQTT approach) (right) when the two-machine configuration is considered and an information of 100 bytes is traversing the LoRaWAN network.

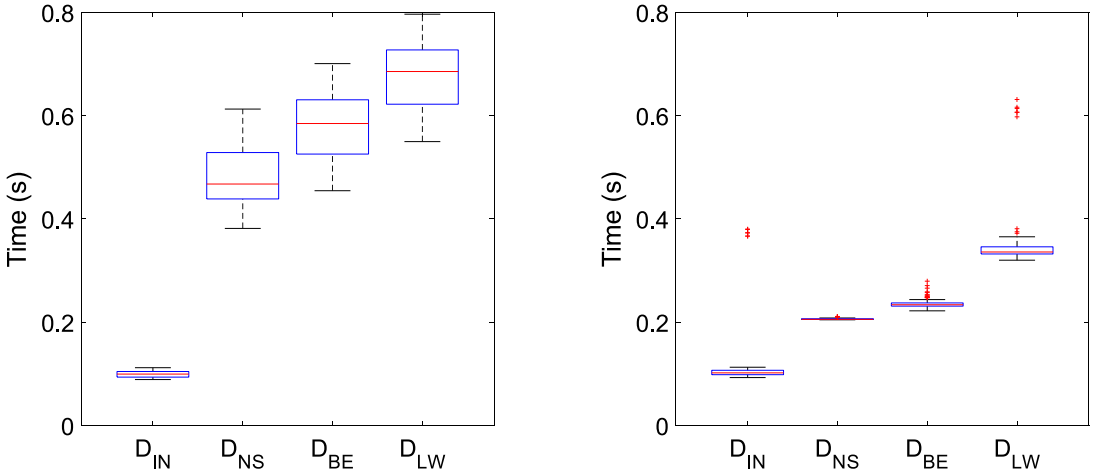


Fig. 9. Boxplots of the considered metrics for Use Case 1 (left) and Use Case 2 (architecture with full MQTT approach) (right) when the two-machine configuration is considered and an information of 5 bytes is traversing the LoRaWAN network.

Table 2
One-machine configuration: average and standard deviation values of the considered metrics for the Use Case 1 and the Use Case 2 (architecture with full MQTT approach).

One-machine configuration									
Use Case 1					Use Case 2				
Metric	5 bytes		100 bytes		Metric	5 bytes		100 bytes	
	AVG [ms]	STD [ms]	AVG [ms]	STD [ms]		AVG [ms]	STD [ms]	AVG [ms]	STD [ms]
D_{IN}	95.9	7	366.5	5	D_{IN}	102.8	6	371.6	6
D_{NS}	369.7	15	381.7	11	D_{NS}	205.9	1	205.9	1
D_{BE}	390.8	21	457.2	16	D_{BE}	231.5	13	236.1	21
D_{LW}	486.7	23	823.8	18	D_{LW}	334.5	13	607.4	21

Table 3
Two-machine configuration: average and standard deviation values of the considered metrics for the Use Case 1 and the Use Case 2 (architecture with full MQTT approach). The D_{AS} metrics is available only when the machines are synchronized with NTP.

Two-machine configuration									
Use Case 1					Use Case 2				
Metric	5 bytes		100 bytes		Metric	5 bytes		100 bytes	
	AVG [ms]	STD [ms]	AVG [ms]	STD [ms]		AVG [ms]	STD [ms]	AVG [ms]	STD [ms]
D_{IN}	99.3	6	366.4	6	D_{IN}	102.1	5	371.3	6
D_{NS}	478.5	59	539.9	53	D_{NS}	206.1	1	205.9	1
D_{AS}	20.4	8	75.5	13	D_{AS}	29.0	21	34.6	21
D_{BE}	577.2	64	654.0	67	D_{BE}	236.5	9	241.8	16
D_{LW}	676.6	63	1020.4	66	D_{LW}	338.7	11	613.2	17

For the considered testbed, in which there is no contention for the medium access, the greatest part of the D_{LW} is due to the backend time D_{BE}. In particular, in the use cases the Application is simple and the most time consuming entity seems to be the Network Server (D_{NS}). The D_{AS} is 10 to 20 times faster than D_{NS}, as shown in Table 3 for the two-machine configuration synchronized with NTP.

The proposed methodology allows to highlight the diverse behaviors imputable to the very different implementations of the two considered use cases: they have important architectural differences; they run under different operating systems, i.e., Windows for Use Case 1 and Linux for Use Case 2; and they use different network socket libraries. For instance, the performance of Use Case 1 worsens passing from one-machine to two-machine configuration, whereas this does not happen for Use Case 2. Hence the performance of Use Case 1 implementation is deeply affected by the use of network sockets between NS machine and AS machine.

The proposed methodology allows to easily evaluate the impact of communication protocol stack as well; indeed, the interval D_{BE}-D_{NS} is somehow related to the AS activity and can be compared with respect to the D_{AS} (when available). As previously stated, the time spent for accessing the network is included moving from one-machine to two-machine scenario. Looking at Tables 2 and 3 it is evident that this access time is particularly significant for Use Case 1 (in the order of 80 ms), whereas it is reduced for Use Case 2 (less than 1 ms).

Additionally, the proposed methodology allows to highlight how peculiarities of backend implementation affect time-related performance. For instance, it is possible to study how NS implementation handles the multiple messages generated by co-located gateways that are listening at the same wireless frame from a single mote. The D_{NS} is insensitive to the message length but the NS dispatches the actual message only after a timeout expiration (sometimes addressed as the *deduplication delay*). The deduplication delay must be long enough to collect as many duplicates as possible, but short enough to give the AS the opportunity to reply to a message in time (for LoRaWAN Class A device, the first receive window may be opened just 1 s after the uplink transmission!). The experimental results are in agreement with the deduplication delay settings reported in both Semtech and LoRa Server documentation and satisfy typical value suggested by service providers (e.g. The Things Network operator reports an average delay lower than 100 ms [49]), and further validate the proposed methodology.

Analyzing the metric standard deviation is useful as well, since it provides some insights about the server(s) implementation and scheduling policy. For instance, large deviation is typically imputable to multimodal distribution emerging from task scheduling periodicity.

5.3. Implementation related metrics and their evaluation

The previously analyzed metrics can be evaluated on any LoRaWAN implementation, since they rely on mandatory network components, whose behavior is described in the specifications. However, depending on the peculiar implementations, some ancillary components may exist as well and additional metrics can be consequently defined. For example, the Use Case 2 includes the LoRa Gateway Bridge and the MQTT Broker; the proposed approach can be easily further extended including the following metrics:

Table 4
Use case 2 related metrics: time needed to traverse the LoRa Gateway Bridge (D_{GB}) and time related to the MQTT Broker processing a new message (D_{BR}).

Use Case 2 – impact of specific architecture parts									
One-machine configuration					Two-machine configuration				
Metric	5 bytes		100 bytes		Metric	5 bytes		100 bytes	
	AVG [ms]	STD [ms]	AVG [ms]	STD [ms]		AVG [ms]	STD [ms]	AVG [ms]	STD [ms]
D_{GB}	0.4	0.2	0.4	0.1	D_{GB}	0.4	0.1	0.4	0.1
D_{BR}	0.7	0.4	0.6	0.4	D_{BR}	0.3	0.3	0.6	0.4

Table 5
Use Case 2 with LoRa Gateway Bridge: average and standard deviation values of the considered metrics with one-machine and two-machine configuration.

Use Case 2 – with LoRa Gateway Bridge									
One-machine configuration					Two-machine configuration				
Metric	5 bytes		100 bytes		Metric	5 bytes		100 bytes	
	AVG [ms]	STD [ms]	AVG [ms]	STD [ms]		AVG [ms]	STD [ms]	AVG [ms]	STD [ms]
D_{IN}	99.7	6	367.6	6	D_{IN}	99	6	367.1	6
D_{NS}	205.8	1	205.7	1	D_{NS}	205.5	1	206.0	1
D_{BE}	229. 9	11	235.9	16	D_{BE}	228.1	11	241.7	20
D_{LW}	330.7	13	604.7	17	D_{LW}	328.2	12	609.9	20

Table 6
Use Case 2 one-machine configuration without the LoRa Gateway Bridge or with the LoRa Gateway Bridge when the integration mode is MQTT or HTTP.

	Use Case 2 – impact of Application interface (One-machine)									
	Without LoRa Gateway Bridge					With LoRa Gateway Bridge				
	Metric	5 bytes		100 bytes		Metric	5 bytes		100 bytes	
		AVG [ms]	STD [ms]	AVG [ms]	STD [ms]		AVG [ms]	STD [ms]	AVG [ms]	STD [ms]
MQTT	D_{BE}	231.5	13	236.1	21	D_{BE}	229. 9	11	235.9	16
	D_{LW}	334.5	13	607.4	21	D_{LW}	330.7	13	604.7	17
HTTP	D_{BE}	344.9	241	396.0	313	D_{BE}	422.4	344.9	356.2	254
	D_{LW}	447.8	241	767.3	313	D_{LW}	523.3	346.1	724.9	256

- D_{GB} = T2bm-T2b, it is related to the time needed to traverse the LoRa Gateway Bridge along the F2b sub flow (see Fig. 4);
- D_{BR} = T3m-T2m, related to the time needed by the MQTT Broker to receive and distribute to subscribers a new message.

As highlighted by results shown in Table 4, the impact of both the LoRa Gateway Bridge and the MQTT Broker is negligible in the considered simple testbed.

Table 5 resumes results of Use Case 2 experiments when the LoRa Gateway Bridge is executed on the same machine running the LoRa Server. These results can be compared with Tables 2 and 3. As expected, D_{IN} is smaller, since the LoRa Gateway Bridge is executed by a PC-class machine and a simpler packet forwarder (without MQTT socket support) is executed by the reduced capability microprocessor of the gateway. Additionally, despite the D_{NS} parameter has similar values in all scenarios involving Use Case 2, the D_{BE} for small packets changes depending if the LoRa Gateway Bridge is executed or not. Hence, the AS (i.e. the LoRa App Server) has a different execution time. As previously stated, the time spent for the AS activity can be estimated by means of the D_{BE}-D_{NS} quantity. The relative frequency histogram of the D_{BE}-D_{NS} quantity is plot in Fig. 10, confirming the time interval distribution is wider when the LoRa Gateway Bridge is not executed.

Use Case 2 also offers different integration modes for the end-user application: based on MQTT Broker subscription or via HTTP message using the POST method. In Table 6, the two modalities are compared for the one-machine scenario. It can be highlighted that HTTP based integration shows larger delays, probably due again to task scheduling, as can be inferred from the analysis of the D_{BE}-D_{NS} time interval. For instance, in the case of messages with 5 bytes information and HTTP integration, Fig. 11 shows a bimodal distribution, usually related to software timeouts.

6. Conclusions

Recently, the Low Power Wide Area Networks became one of the most diffused wireless connection methods for the Internet of Things (IoT), because of their wide coverage, low complexity and low power consumption. LoRaWAN emerged

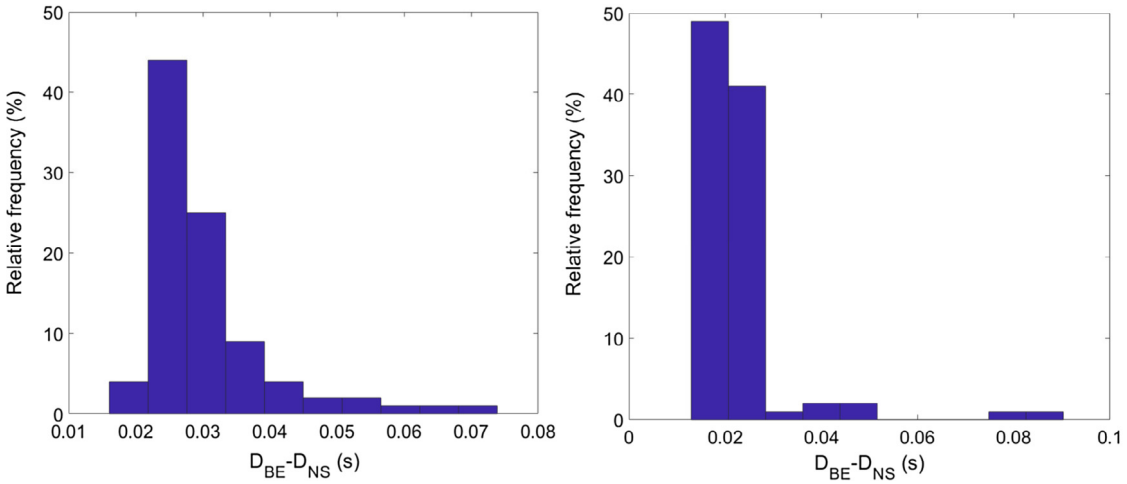


Fig. 10. Relative frequency (in percentage) of the time interval D_{BE}-D_{NS} for Use Case 2 — two-machine configuration without the LoRa Gateway Bridge (left) or with the LoRa Gateway Bridge (right) and an information of 5 bytes is traversing the LoRaWAN network.

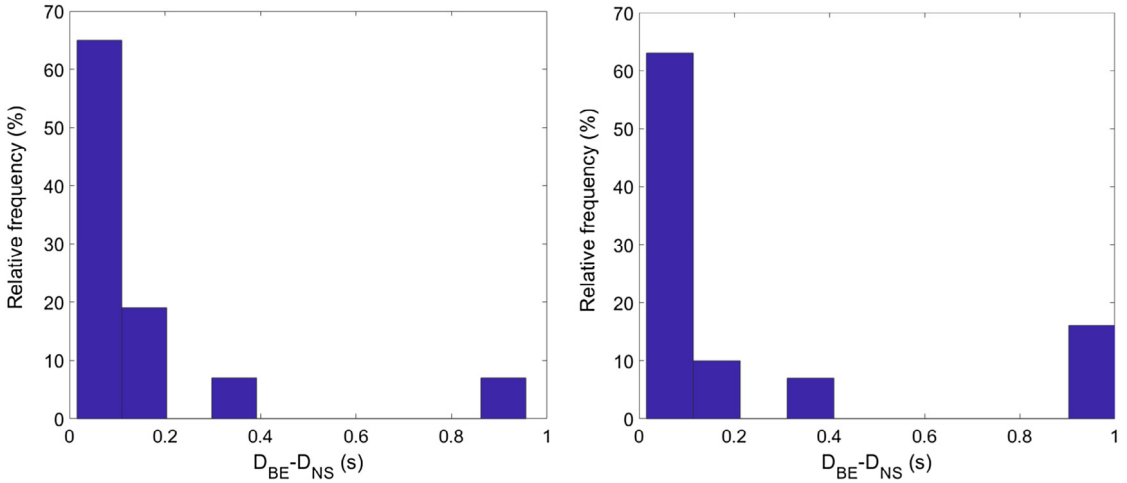


Fig. 11. Relative frequency (in percentage) of the time interval D_{BE}-D_{NS} for Use Case 2 — one-machine configuration and HTTP integration mode without the LoRa Gateway Bridge (left) or with the LoRa Gateway Bridge (right) and an information of 5 bytes is traversing the LoRaWAN network.

as one of the leading technologies with a wide offer of implementations, each one showing support for different hardware and software architectures. This paper presents a methodology for the experimental assessment of architectural delays of LoRaWAN implementations. The new approach has three phases: a mapping phase that identifies the LoRaWAN blocks, as defined in the specification, inside the implementation under test; a setup phase in which the relevant timestamping points are isolated inside the architecture; and last, a measurement phase where the delays are evaluated along the information path from LoRaWAN node to LoRaWAN user application. The paper is completed by two Use Cases that demonstrate the use and the effectiveness of the proposed methodology.

The experimental results clearly show that the proposed methodology is useful to compare the performance of entire LoRaWAN systems (Use Case 1 versus Use Case 2), with great benefit of the owner of the infrastructure and of the end user. The high level of details of the calculated metrics can be directly used to make implementation choices or to maximize results of optimization actions. Specific view of implementation particularities (e.g. for Use Case 2) makes the proposed methodology also suitable for architecture designers and stack developers.

Acknowledgment

The authors thank Dr. Ing P. Palazzoli of the A2A Smart City SpA for the discussion about application requirements.

Funding

The research has been partially funded by: research grant MIUR SCN00416, "Brescia Smart Living: Integrated energy and services for the enhancement of the welfare", Ministero dell'Istruzione dell'Università e della Ricerca, Italy; and research grant "H&W Smart Break 2". University of Brescia, Italy.

References

[1] U. Raza, P. Kulkarni, M. Sooriyabandara, Low power wide area networks: An overview, IEEE Commun. Surv. Tutor. 19 (2) (2017) 855–873, Secondquarter.
[2] C. Goursaud, J.-M. Gorce, Dedicated networks for IoT: PHY/MAC state of the art and challenges, EAI Endorsed Trans. Internet Things 1 (1) (2015).
[3] F. Montori, L. Bedogni, M. Di Felice, L. Bononi, Machine-to-machine wireless communication technologies for the internet of things: Taxonomy, comparison and open issues, Pervasive Mob. Comput. 50 (2018) 56–81.
[4] Q. Song, L. Nuaymi, X. Lagrange, Survey of radio resource management issues and proposals for energy-efficient cellular networks that will cover billions of machines, EURASIP J. Wirel. Commun. Netw. 2016 (1) (2016) 140.
[5] H.H.R. Sherazi, M.A. Imran, G. Boggia, L.A. Grieco, Energy harvesting in LoRaWAN: a cost analysis for the industry 4.0, IEEE Commun. Lett. 22 (11) (2018) 2358–2361, http://dx.doi.org/10.1109/LCOMM.2018.2869404.
[6] Y.-B. Lin, Y.-W. Lin, C.-Y. Hsiao, S.-Y. Wang, Location-based IoT applications on campus: The IoTtalk approach, Pervasive Mob. Comput. 40 (2017) 660–673.
[7] F. Van den Abeele, J. Haxhibeqiri, I. Moerman, J. Hoebeke, Scalability analysis of large-scale LoRaWAN networks in ns-3, IEEE Internet Things J. 4 (6) (2017) 2186–2198.
[8] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, T. Watteyne, Understanding the limits of LoRaWAN, IEEE Commun. Mag. 55 (9) (2017) 34–40.
[9] K. Mikhaylov, Juha Petaejaejaervi, T. Haenninen, Analysis of capacity and scalability of the LoRa low power wide area network technology, in: 22th European Wireless Conference, Oulu, Finland, 2016, pp. 1–6.
[10] D. Magrin, M. Centenaro, L. Vangelista, Performance evaluation of LoRa networks in a smart city scenario, in: 2017 IEEE International Conference on Communication, ICC, Paris, 2017, pp. 1–7.
[11] R.B. Sørensen, D.M. Kim, J.J. Nielsen, P. Popovski, Analysis of latency and MAC-layer performance for class a LoRaWAN, IEEE Wirel. Commun. Lett. 6 (5) (2017) 566–569.
[12] M.C. Bor, U. Roedig, T. Voigt, J.M. Alonso, Do LoRa low-power wide-area networks scale? in: Proc. 19th ACM Int. Conf. Model. Anal. Simulat. Wireless Mobile Syst., 2016, pp. 59–67.
[13] Dvornikov, P. Abramov, S. Efremov, L. Voskov, QoS metrics measurement in long range IoT networks, in: 2017 IEEE 19th Conference on Business Informatics, CBI, Thessaloniki, 2017.
[14] F. Delobel, N. El Rachkidy, A. Guitton, Analysis of the delay of confirmed downlink frames in class B of LoRaWAN, in: 2017 IEEE 85th Vehicular Technology Conference, VTC Spring, Sydney, NSW, 2017.
[15] D. Croce, M. Gucciardo, S. Mangione, G. Santaromita, I. Tinnirello, Impact of LoRa imperfect orthogonality: Analysis of link-level performance, IEEE Commun. Lett. 22 (4) (2018) 796–799.
[16] M. Rizzi, P. Ferrari, A. Flammini, E. Sisinni, Evaluation of the IoT LoRaWAN solution for distributed measurement applications, IEEE Trans. Instrum. Meas. 66 (12) (2017) 3340–3349.
[17] P. Ferrari, A. Flammini, M. Rizzi, E. Sisinni, M. Gidlund, On the evaluation of LoRaWAN virtual channels orthogonality for dense distributed systems, in: IEEE M & N 2017 IEEE Measurement and Networking, Naples, 2017, pp. 1–6.
[18] M. Rizzi, P. Ferrari, A. Flammini, E. Sisinni, M. Gidlund, Using LoRa for industrial wireless networks, in: 2017 IEEE 13th International Workshop on Factory Communication Systems, WFCS, Trondheim, 2017, pp. 1–4.
[19] P. Ferrari, A. Flammini, S. Rinaldi, M. Rizzi, E. Sisinni, On the use of LPWAN for EVehicle to grid communication, in: 2017 AEIT International Annual Conference, Cagliari, Italy, 2017, pp. 1–6.
[20] S. Rinaldi, M. Pasetti, E. Sisinni, F. Bonafini, P. Ferrari, M. Rizzi, A. Flammini, On the mobile communication requirements for the demand-side management of electric vehicles, Energies 11 (5) (2018) 1220.
[21] M. Rizzi, A. Depari, P. Ferrari, A. Flammini, S. Rinaldi, E. Sisinni, Synchronization uncertainty versus power efficiency in LoRaWAN networks, IEEE Trans Instrum. Meas. 68 (4) (2019) 1101–1111, http://dx.doi.org/10.1109/TIM.2018.2859639.
[22] A. Mahmood, E. Sisinni, L. Guntupalli, R. Rondón, S.A. Hassan, M. Gidlund, Scalability analysis of a LoRa network under imperfect orthogonality, IEEE Trans. Ind. Inf. 15 (3) (2019) 1425–1436, http://dx.doi.org/10.1109/TII.2018.2864681.
[23] D.F. Carvalho, P. Ferrari, A. Flammini, E. Sisinni, A test bench for evaluating communication delays in LoRaWAN applications, in: 2018 Workshop on Metrology for Industry 4.0 and IoT, Brescia, 2018, pp. 248–253.
[24] Q. Song, X. Lagrange, L. Nuaymi, Evaluation of macro diversity gain in long range ALOHA networks, IEEE Commun. Lett. 21 (11) (2017) 2472–2475.
[25] J. Navarro-Ortiz, S. Sendra, P. Ameigeiras, J.M. Lopez-Soler, Integration of LoRaWAN and 4g/5g for the industrial internet of things, IEEE Commun. Mag. 56 (2) (2018) 60–67.
[26] R. Yasmin, J. Petäjäjärvi, K. Mikhaylov, A. Pouttu, On the integration of LoRaWAN with the 5G test network, in: 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications, PIMRC, Montreal, QC, Canada, 2017, pp. 1–6.
[27] M.C. Vuran, A. Salam, R. Wong, S. Irmak, Internet of underground things in precision agriculture: Architecture and technology aspects, Ad Hoc Netw. 81 (2018) 160–173.
[28] A. Ouya, B.M. De Aragon, C. Bouette, G. Habault, N. Montavont, G.Z. Papadopoulos, An efficient electric vehicle charging architecture based on LoRa communication, in: 2017 IEEE International Conference on Smart Grid Communications, SmartGridComm, Dresden, 2017, pp. 381–386.
[29] M. Luvisotto, F. Tramarin, L. Vangelista, S. Vitturi, On the use of LoRaWAN for indoor industrial IoT applications, Wirel. Commun. Mob. Comput. 2018 (2018).
[30] X. Ouyang, O.A. Dobre, Y.L. Guan, J. Zhao, Chirp spread spectrum toward the nyquist signaling rate—Orthogonality condition and applications, IEEE Signal Proc. Lett. 24 (10) (2017) 1488–1492.
[31] M. Toril, V. Wille, Optimization of the assignment of base stations to base station controllers in GERAN, IEEE Commun. Lett. 12 (6) (2008) 477–479.
[32] M. Centenaro, L. Vangelista, R. Kohno, On the impact of downlink feedback on LoRa performance, in: 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications, PIMRC, Montreal, QC, 2017, pp. 1–6.

[33] A. Pop, U. Raza, P. Kulkarni, M. Sooriyabandara, Does bidirectional traffic do more harm than good in LoRaWAN based LPWA networks? in: GLOBECOM 2017-2017 IEEE Global Communications Conference, Singapore, 2017, pp. 1–6.
[34] LoRa Alliance, LoraWAN specification v 1.1, 2018, Available online: https://lora-alliance.org/resource-hub/lorawantm-specification-v11. (Accessed 3 October 2018).
[35] G. Yang, H. Liang, A smart wireless paging sensor network for elderly care application using LoRaWAN, IEEE Sensors J. 18 (22) (2018) 9441–9448, http://dx.doi.org/10.1109/JSEN.2018.2870674.
[36] A. Lozano, J. Caridad, J.F. De Paz, G.V. González, J. Bajo, Smart waste collection system with low consumption LoRaWAN nodes and route optimization, Sensors 18 (5) (2018) 1465.
[37] N. Bebelaar, R.C. Braggaar, C.M. Kleijwegt, R.W.E. Meulmeester, G. Michailidou, N. Salheb, S. van der Spek, N. Vaissier, E. Verbree, Monitoring urban environmental phenomena through a wireless distributed sensor network, Smart Sustain. Built Environ. 7 (1) (2018) 68–79.
[38] Official web site, 2018, https://www.thethingsnetwork.org/, Accessed 3 October 2018.
[39] Official web site, 2018, https://www.thethingsnetwork.org, Accessed 3 October 2018.
[40] Semtech Ltd, Lora iot starter kit user guide, 2018, Available onnline: http://semtechlorakit.blogspot.com/. (Accessed 3 October 2018.
[41] Semtech Ltd, LoRaWAN Network Server Demonstration: High Level Description, Application Note ANNWS.05.3.1.W.SYS.
[42] B.C. Fargas, M.N. Petersen, GPS-free geolocation using LoRa in low-power WANs, in: 2017 Global Internet of Things Summit, GIoTS, Geneva, 2017, pp. 1–6.
[43] S.A. A'ssri, F.H.K. Zaman, S. Mubdi, The efficient parking bay allocation and management system using LoRaWAN, in: 2017 IEEE 8th Control and System Graduate Research Colloquium, ICSGRC, Shah Alam, 2017, pp. 127–131, http://dx.doi.org/10.1109/ICSGRC.2017.8070581.
[44] Semtech Ltd, 7, available via subscription to the LoRa Community https://semtech.force.com/lora.
[45] P. Ferrari, A. Flammini, E. Sisinni, S. Rinaldi, D. Brandão, M.S. Rocha, Delay estimation of industrial IoT applications based on messaging protocols, IEEE Trans. Instrum. Meas. 67 (9) (2018) 2188–2199, http://dx.doi.org/10.1109/TIM.2018.2813798.
[46] The lora server project, 2018, official web site: http://www.loraserver.io. (Accessed 3 October 2018).
[47] S. Rinaldi, D. Della Giustina, P. Ferrari, A. Flammini, E. Sisinni, Time synchronization over heterogeneous network for smart grid application: Design and characterization of a real case, Ad Hoc Netw. 50 (2016) 41–57.
[48] J.A. Sherman, J. Levine, Usage analysis of the NIST internet time service, J. Res. Natl. Inst. Stand. Technol. 121 (2016) 33–46.
[49] The things network official web site, 2019, online at https://www.thethingsnetwork.org/docs/network/architecture.html. (Accessed 15 January 2019).

Dhiego Fernandes Carvalho holds a major in Computer Engineering and Master's degree in computer networking at the Federal University of Rio Grande do Norte. He is member of Ph.D.'s Technology For Health course since December of 2017, he is working on Low Power Wide Area Networks for smart cities and smart applications for Industry 4.0.

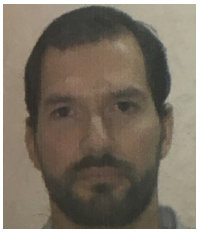
Paolo Ferrari was born in Italy, in 1974. He received the M.Sc. (Hons.) degree in electronic engineering and the Ph.D. degree in electronic instrumentation from the University of Brescia, Brescia, Italy, in 1999 and 2003, respectively. He is currently an Associate Professor with the Department of Information Engineering, University of Brescia. He has authored more than 150 international papers. His current research interests include embedded measurement instrumentation, smart sensors, sensor networking, smart grids, real-time Ethernet, and fieldbus applications for Industry 4.0. Dr. Ferrari is a member of IEC SC65C MT9, IEC TC65C WG10, and CENELEC/IEC TC65X IRWC. In 2013, he received the Technical Award from the IEEE Instrumentation and Measurement Society.

Emiliano Sisinni graduated (M.Sc.) in Electronics Engineering and received the Ph.D. degree in Electronic Instrumentation from the University of Brescia, Italy, in 2000 and 2004, respectively. Currently, he is an Associate Professor in Electronics with the Department of Information Engineering, University of Brescia. In the past, his activity focused on numerical signal analysis, with particular interest in DSP-based instrumentation. His current research interests include smart sensors, wireless sensor networking, wired and wireless industrial communications, and smart devices. He is an author of more than 140 international papers, published on international journals, books, patents and conference proceedings. He is a member of IEC SC65C WG16 and WG17 and IEC TC65 WG17.

Alessandro Depari received the M.Sc. Degree in "Electronics Engineering" and the Ph.D. Degree in "Electronic Instrumentation" from the University of Brescia, Italy, in 2002 and 2006, respectively. Since 2007, he has been Assistant Professor with the Department of Information Engineering, University of Brescia, in the field of electrical and electronic measurements. His research activity is various and includes: sensor signal conditioning and processing, in particular concerning chemical sensors for artificial olfactory systems; embedded systems based on microcontrollers, DSPs and FPGAs; development of smart sensors and sensor networks for distributed measurement with industrial communication systems; design of methods and digital electronic circuits for numeric measurement instrumentation; design and development of wearable systems for mobile health (m-Health) applications. He is author or co-author of more than 80 scientific works, published on international journals, books and conference proceedings.

Stefano Rinaldi received the M.S. degree (Hons.) in electronic engineering and the Ph.D. degree in electronic instrumentation from the University of Brescia, Brescia, Italy, in 2006 and 2010, respectively. He is currently an Assistant Professor with the Department of Information Engineering, University of Brescia. His current research interests include industrial real-time Ethernet network, smart grids, wireless smart sensors, time synchronization methods, and field-programmable gate array system-on-chip design.

Marco Pasetti is currently Research Fellow at the Department of Information Engineering of the University of Brescia, where he carries out research activities in the field of energy systems and distributed generation, with specific focus on the modelling and management of distributed energy resources and on their integration into distribution networks. His main research interests include: energy systems, distributed generation, renewable energy sources, solar engineering, photovoltaics, energy storage, energy management systems; supervisory control and data acquisition, smart grids, and micro-grids.



Diego Silva holds a degree in Computer Engineering from the Federal University of Rio Grande do Norte (2002), a Master's degree in Electrical Engineering from the Federal University of Rio Grande do Norte (2005) and a Ph.D. in Electrical Engineering from the Federal University of Rio Grande do Norte (2008). He is currently an associate professor at the Federal University of Rio Grande do Norte. He has experience in Computer Engineering, with emphasis on Industrial Automation, where he has been developing projects for the Brazilian industry since 2003 in various topics such as alarm management, connectivity and industrial protocols, asset management and data visualization. In addition, he has also been involved in technological development projects in the areas of education and building automation. His interests include Industry 4.0, IoT, Ilot, signal processing, cloud computing, among others. He did postdoctoral studies at the University of Southern California between 2015 and 2016, where he delved into the subject of Digital Signal Processing in Graphs.