

"An Investigation into Heterogeneous IoT Networking Using a Modular Multi-interface Gateway"

Geetanjali Sharma¹, Anoop Tyagi²

Student¹, Professor²

Department of CSE

HMR Institute of Technology & Management

Corresponding Author's E-mail: *geetanjali.sharma11@rediffmail.com¹*

Abstract

Over the last decade, the widespread use of Internet of Things (IoT) designs and applications in a variety of areas has pushed research on low-power wireless networking protocols. Furthermore, various standards have been created, emphasising the need of allowing data to flow between different communication protocols and network interfaces. As a result, devices such as Gateways (GWs) play an important part in many IoT applications and will have an influence on future advancements and possibilities. The design and implementation of a new modular and scalable GW architectural solution is provided in this article, which is suited for a wide range of use case situations and may be used as a starting point for numerous future upgrades and applications. The role of distinct interfaces in specific use situations where the suggested approach might be deployed is addressed based on experimental performance findings.

Keywords: *IoT, Gateways (GWs), Network interfaces, Wireless communication, communication protocols*

INTRODUCTION

The Internet of Things (IoT) is a relatively new sector that has sprung up as a result of massive technical advancements over the previous decade, resulting in the

development of a plethora of nodes with sensors and various (typically wireless) communication interfaces. Wireless Sensor Networks (WSNs) have been used in a variety of industries, including smart

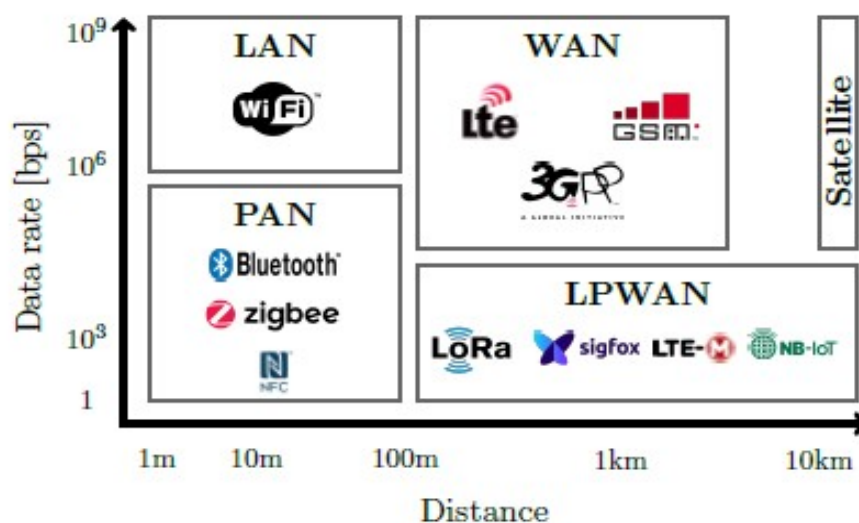
cities, Industry 4.0, and data sensing for smart agriculture. When dealing with wide-range networks encompassing multiple applications and surroundings, heterogeneous WSNs for big applications can be represented by a "network of networks," most likely based on several protocols.

As illustrated in Fig. 1, a generic IoT scenario includes heterogeneous communication solutions that may be separated into long-range (often low data rate) and short-range (generally high data rate) protocols. One of the primary issues in this context is to enable visible (from a high-level perspective) connection across various types of networks. Gateways (GWs) are key pieces responsible for routing data from one network to another, and their characteristics and capabilities

will be critical in the future of vast heterogeneous IoT applications.

GWs, on the other hand, are often complex nodes that enable additional features like as smart data routing, data processing, and queuing mechanisms. Smart GWs will undoubtedly benefit heterogeneous networks with a large number of nodes that require data pre-processing techniques to minimise the transfer of large amounts of data (e.g., collected by hundreds of IoT sensing nodes) through data-constrained protocols (e.g., through the Long Range Wide Area Network, LoRaWAN).

We describe a prototype of a scalable and flexible multi-interface GW that includes pairwise bidirectional smart data routing techniques across IEEE 802.11, Bluetooth Low Energy (BLE), and LoRaWAN communication interfaces in this study.



Furthermore, in order to deal with LoRaWAN's limited transmission capacity, the suggested architecture includes an internal data pre-processing mechanism that evaluates the input data and offers appropriate parsing and conversion rules to maximise throughput while reducing resource consumption. Machine Learning (ML)-based data analysis approaches might be applied to improve the utilisation of limited protocols on our multi-interface GW. As a result, this technique enables resource usage optimization, for example, based on data relevance.

The rest of this paper is laid out as follows. We go into the history of existing solutions in Section II. The built IoT GW architecture is shown in Section III. In Section IV, we assess the GW's throughput performance and describe some potential use cases. Finally, we get to our findings in Section V.

CONNECTED WORKS

The IoT's vast range of situations and applications allows for the consideration of a huge number of communication protocols, each with its own set of benefits and downsides depending on the sector in which it must function. The most frequent protocols are I LoRaWAN, Narrowband IoT (NB-IoT), and Sigfox [2] for long-

range communications; (ii) IEEE 802.11 (Wi-Fi) and ZigBee [3] for mid-range communications; and (iii) Bluetooth Low Energy (BLE) for short-range communications [4]. In terms of data rate, capacity, and use policy, these protocols are highly varied. This must be carefully considered when they are all used by a common multi-interface node that acts as a junction point for heterogeneous communications protocols.

In most cases, two entities (GWs and bridges) are used to connect several networks [5].

- A GW is often a hardware (HW) solution that allows data to flow between two networks. The fundamental distinction between GWs and other networking nodes is that GWs use many protocols to link various types of networks [6]. They can also work at any layer of the ISO/OSI paradigm.
- A bridge (also known as a network bridge) is a networking device that combines various networks into a single aggregate network [7]. A GW can be thought of as a "packet forwarder" between multiple interfaces, however it only operates at the ISO/OSI model's Data Link layer

(Layer 2). Bridges can also only exist across networks that have equivalent performance (e.g., in terms of throughput) because they often lack a "smart" buffering/queuing scheme like GWs.

As a result, GWs may be preferable to bridges because they provide intelligent handling and management of data routing rules, data analysis techniques [8,] and queuing policies for network interfaces with varying throughput characteristics. Because of these factors, intelligent GWs play (and will continue to play) an increasingly important role in the future. This is especially true in the case of big heterogeneous networks, which need "intelligent" data pre-processing capabilities in order to feed protocols with poor throughput enormous volumes of data gathered by hundreds of sensing IoT nodes.

This must be taken into account, especially in light of the channel's and realistic payload's limitations and constraints.

Despite the fact that there are a variety of GWs on the market, the most of them are pricey and limited to a set number of network interfaces [9]. Furthermore, many products are "closed source," preventing the addition of new features and the connection of additional network interfaces. Another issue with these ready-made systems is that their design is neither scalable or modular, making it difficult to add new network interfaces and routing rules between them. As a result, the design and implementation of a new modular and open GW architecture may enable the deployment of new IoT applications based on heterogeneous networks to be more quickly.

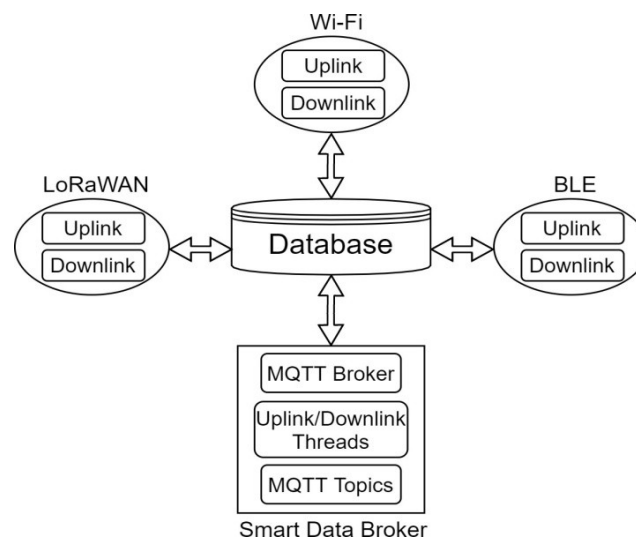


Fig. 2: GW architecture block diagram

PROPOSED SMART IOT GW ARCHITECTURE

A. System Architecture

We use a Raspberry Pi 3 Model B+ (RPI) with an extra Dragino LoRaWAN cap [10] to build and install a prototype of the proposed smart IoT multi-interface GW architecture. The GW can perform and operate on three separate heterogeneous network interfaces based on this HW: BLE and Wi-Fi, which are naturally given by the RPI; and LoRaWAN, which is provided by the Dragino headgear. Internally, the created GW architecture is built on three important components, as illustrated in Fig. 2: one specialised software (SW) entity per network interface (at least), MQTT-based internal administration, and relational SQL-like storage.

Each network interface refers to a specific SW entity that is responsible for handling its own functionalities (packet buffers, payload constraints, transmission policy, advertising services, uplink and downlink operation handling, and so on) as well as storing data obtained from end nodes in the appropriate SQL-like downlink table. The "Smart Data Broker," which is in charge of I retrieving data from interfaces' down-link DB tables; (ii) parsing them into "DataUnits" (DUs); (iii) checking

their recipients (both at interface and even specified-device level); and (iv) in turn sending to the right interface MQTT topic based on the defined routing rules and data recipients, is the connection point among these internal SW modules.

Then, for each MQTT subject, there is a subscriber thread that processes the incoming message and puts the acquired data into the uplink DB queue table of the specified interface, acting as a FIFO-based buffer. As a result, the SW module of each network interface obtains data from the uplink DB table and wraps it in DUs using a good parsing approach, making it ideal for tiny payload packets and numerous messages. Furthermore, IP and MAC addresses are properly converted: the former is encoded as a single integer value to reduce the impact of their size on the overall packet, while the latter is converted into their corresponding HEX values to reduce the impact of their size on the payloads of LoRaWAN messages. Data pre-processing processes might be augmented by ML-based data analysis tools in order to pick just the data, among all data obtained, that needed to be delivered, in order to further optimise the utilisation of such restricted networks (e.g., according to their relevance).

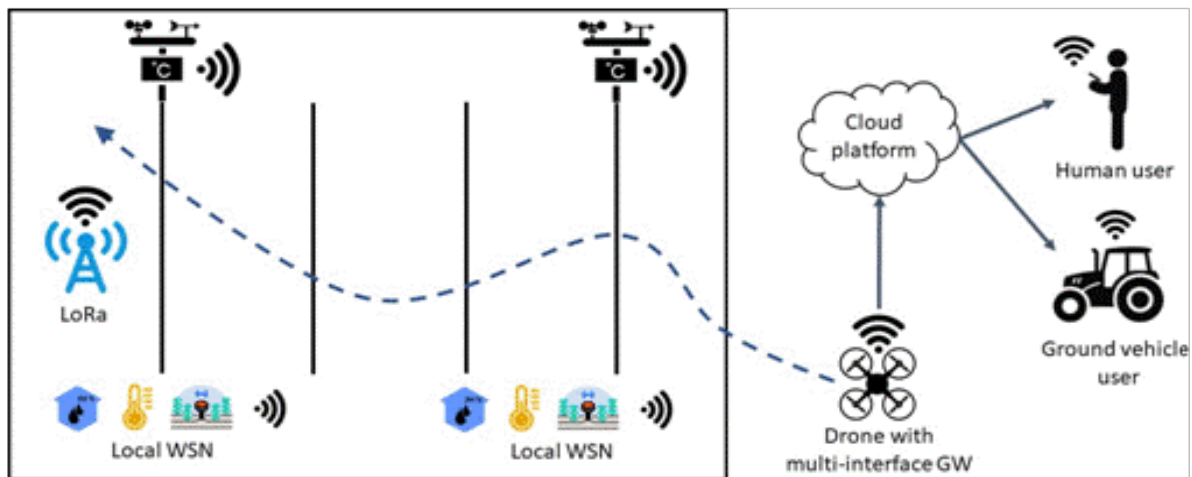


Fig. 3: Example of use case.

The proposed multi-interface GW's whole architecture is therefore modular and scalable, even if the network interfaces involved are diverse. In fact, each interface has: uplink and downlink DB tables; one or two SW entities retrieving/sending data from/to the end nodes connected on the interfaces themselves; one uplink MQTT topic and two data broker entities routing data towards the right MQTT topic and subscribing to the interface-related topic in order to retrieve data that must be sent on another network interface. Because the data flow topology is the same for each component, this technique allows for the inclusion of additional interfaces. Because each communication protocol has its own limits and regulations, only the interface-related SW entities will be different.

Furthermore, the proposed GW architecture's modularity is demonstrated

by the fact that, because the routing policies are based on a publish/subscribe paradigm, the GW can accept traffic flows from additional data collection modules, using either new network interfaces or some of the currently used ones (e.g., BLE). By simply designing additional queue-based-channels and the related suitable data parsing techniques, it is feasible to incorporate data harvesting systems utilising various data formats. As a result, the maintenance of routing policies may be delegated from the GW and (logically) centralised to external controllers (e.g., in the Cloud) to improve infrastructure management (this is out of the scope of this paper and is left as a future development).

Because data transmission between MQTT topics and SQL databases takes time, the deployed GW system is best suited for

non-real-time applications. Even if system performance might be enhanced in order to minimise processing time, the system is acceptable for all IoT applications and situations where data must be gathered among multiple end nodes within a specific period of time (in the range of at least 10 s).

To accommodate new applications, the present prototype employs a separate downlink DB table for each communication interface, which was designed with internal data pre-processing capabilities. However, downlink DB tables may be removed by creating a second MQTT topic for each interface inside the "Smart Data Broker." The downlink DUs received from the interface servers could be handled by this second MQTT subject, which would then route them to the appropriate uplink MQTT topic.

To summary, the suggested IoT GW's possible performance bottlenecks may be mitigated by optimising HW resources, protocols, and SW.

B. Discussion on IoT Use Cases

A smart GW, such as the one presented in this study, is ideal for a variety of IoT use cases, as explained in Section I. Given the limitations of the established

communications protocols, common situations of relevance include environmental data sensing and remote monitoring.

The system is particularly suited to a smart greenhouse or smart farming solution, such as the one shown in Fig. 3, in which a large number of sensors with short-range communication capabilities are scattered across a broad region remote from a steady Internet connection in order to reach the application server. This server might be accessed by connecting to a Low-Power WAN (LPWAN) using our multi-interface GW, for example. In a Farm-as-a-Service (FaaS)-oriented manner, a valued service based on the received data can eventually be offered to a farmer or to a feedback system for further processing.

Remote monitoring via the usage of Unmanned Aerial Vehicles is another use case in which the suggested multi-interface GW might be an essential entity (UAVs). In a smart city scenario, for example (although this also applies to smart farming or remote environmental sensing), a large number of short-range WSNs may be deployed to gather data on the parameter of interest, with the data perhaps pre-processed. By flying over/near these WSNs, a drone equipped with a multi-

interface GW may collect this data (using either short-range or long-range protocols) and transfer them to the final collecting node.

Another inherent benefit of the proposed multi-interface IoT GW is the ability to employ one of the interfaces as a backup option in the event of important communication demands or if one of the remaining interfaces fails. To conclude, all of the aforementioned applications necessitate the deployment of a few heterogeneous connection protocols that can communicate with one another over a multi-interface GW.

EXPERIMENTAL RESULTS

A actual testbed, consisting of 10 nodes based on the ESP32 SoC HW platform [11], has been built to assess system performance and restrictions on each network interface, as shown in Fig. 4. Five BLE end nodes and five Wi-Fi end nodes have been installed in total, all with the same sensors and SoC for quick deployment (Fig. 4a).

The multi-interface GW, based on a Raspberry Pi 3 Model B+ (Fig. 4b) with a Dragino LoRaWAN extension board, has been put in the centre of the test environment to provide adequate coverage

in terms of both BLE and Wi-Fi protocols, as predicted in Section III. Then, in order to ensure robustness against humidity, water, and harsh weather conditions, a Single Channel LoRaWAN GW was developed and put, for better performance, on the antenna's pole on the roof (Fig. 4c) in an IP65 box. As mentioned in Section II, the created GW features three communication interfaces: BLE, IEEE 802.11, and LoRaWAN. Because of its restricted payload and duty-cycled network usage strategy, LoRaWAN is the most limiting of these protocols [12].

Instead, because to its long connection time and payload size, BLE is the more constrained of the two technologies. There are certain limits to the Wi-Fi interface, particularly in terms of the maximum number of connected devices. However, when compared to both LoRaWAN and BLE limitations, the IEEE 802.11 limits are insignificant and have no effect on GW throughput, which is substantially influenced by BLE and LoRaWAN network interfaces.

The throughput study was done to determine the notional number of DUs that the system can handle without hitting a bottleneck at any interface. As a result, the experimental campaign tries to determine

the greatest supported throughput at each interface, assuming that the GW delivers all acquired data to the interface in question.

This is a worst-case situation, as DUs often travel to several interfaces, lowering the traffic strain on a single communication interface (in comparison to the worst-case scenario).

The experimental evaluations were carried out by manually filling the uplink DB table of the tested interface with dummy data (by sending the same DU to the interface's uplink MQTT topic many times), in order to have enough elements in the system's queue to analyse the system's processing times and capabilities. Then, by examining the difference between each uplink message on the specified interface (i.e., the difference between arrival timestamps), the performance results were produced.

A. LoRaWAN Network Interface

With the Spreading Factor (SF) set to 7 and the protocol's duty cycle met, the LoRaWAN interface can handle up to 86 daily messages for a total daily payload of about 19 092 bytes, which corresponds, given the testbed scenario and a 21 byte average DU size, to a total of 90 daily LoRaWAN messages with an average size

of 210 bytes (10 DUs) that could carry up to 18 900 bytes. However, in order to determine the shortest time delay between two consecutive LoRaWAN message broadcasts, a test was conducted without the use of any LoRaWAN fair use policy. Our findings suggest that creating a 10 DU message payload and sending it across the LoRaWAN network takes an average of 14:6 seconds (due primarily to DB contacts), providing a maximum daily payload for the LoRaWAN protocol of about 59; 178 DUs, or 1; 242; 738 bytes.

B. Bluetooth Low Energy Network Interface

In comparison to LoRaWAN, the BLE protocol has various limitations. One restriction, for example, is the time it takes to establish a Bluetooth Low Energy (BLE) connection between a BLE client and a BLE server, which is often based on the Generic ATtribute profile (GATT) feature.

A complete interaction cycle (comprising full service discovery, connection, data exchange, and disconnection) between the end node and the GW takes about 8 seconds, plus the DB queries' processing times, which are dependent on the number of DUs waiting inside the queuing system inserted into the BLE payload.

The GW throughput is severely hampered by this period of time. As a result of the extended connection time required by BLE nodes, an end node requires about 11 seconds to complete a full connection-writing-reading-disconnection cycle for a single DU payload. As a result, the transfer of ten 10 s DUs (as determined by our LoRaWAN experimental assessment) requires a total cycle duration of 32:5 s.

This enables the machine to attain a maximum daily uplink payload of 26; 585 DUs (558; 285 bytes) each day.

C. IEEE 802.11 Network Interface

The Wi-Fi protocol is influenced by the GW architecture processing time in LoRaWAN and BLE, since it takes roughly 1:2 s to get one DU, open a UDP connection towards the end node, and deliver the DU across the socket. The system can only send one DU at a time because to the implementation of the SW module controlling the Wi-Fi uplink thread. As a result, it takes roughly 12 seconds to transfer 10 DUs, as done by BLE and LoRaWAN, limiting the Wi-Fi uplink daily throughput to 72; 000 DUs, or 1; 512; 000 bytes.

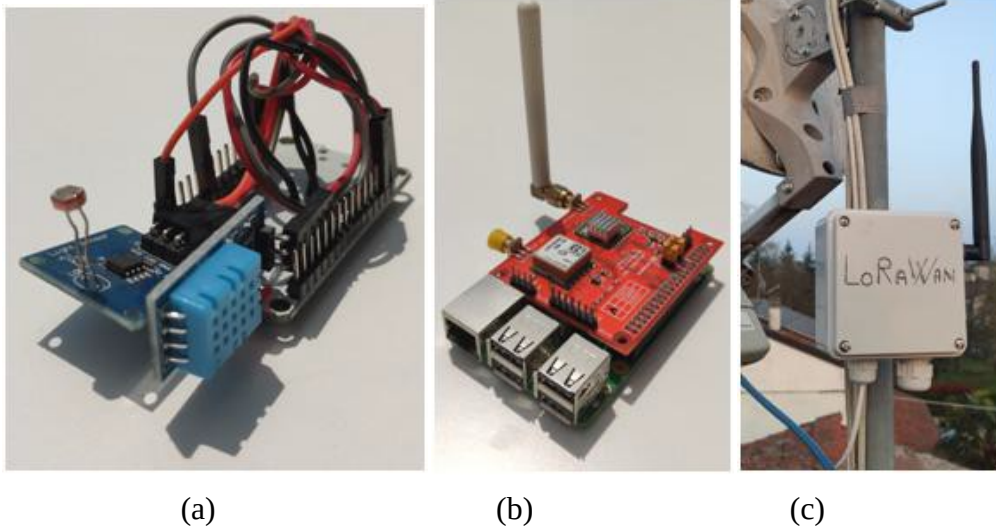


Fig. 4: Experimental testbed used for evaluating the performance of the proposed system, and composed by several IoT sensor nodes, based on ESP32 architecture (a), a multi-interface GW based on a Raspberry Pi 3 Model B+ (b), a Single Channel LoRaWAN GW placed, for better performance, on the antenna's pole on the roof (c).

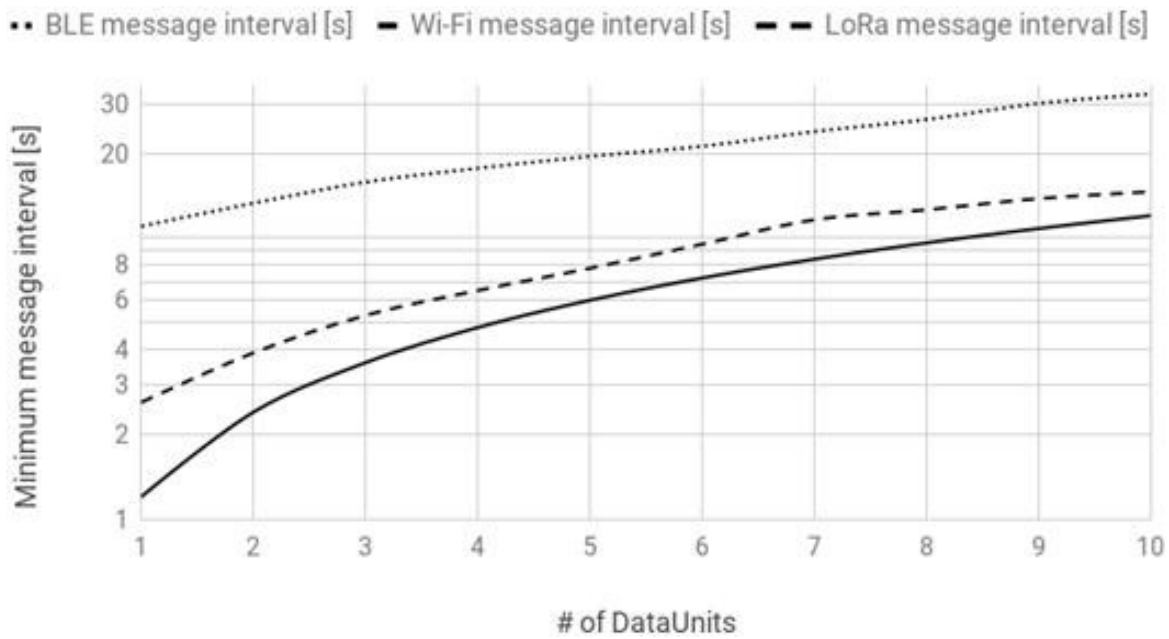


Fig. 5: LoRaWAN, Wi-Fi and BLE minimum uplink message interval for different number of DUs.

D. Discussion

In Fig. 5, a visual depiction of the various minimum message intervals (dimension: [s]) with the accepted protocols (BLE, LoRaWAN, IEEE 802.11) is given for various numbers of DUs—from 1 to 10 (the maximum conveyed by a LoRaWAN message).

The experimental assessment findings reveal the suggested multi-interface GW's potential as well as its bottlenecks. In particular, the SW layer of the system may simply detect which network protocols are limited and the minimum time delay between successive messages on the same network interface (on the basis of the number of DUs carried). The daily system

throughput is determined by this minimal time period. We point out that our findings are "stress tests," because in a real use scenario, data flows across several interfaces to prevent overloading a single one. Finally, we should mention that the suggested implementation (based on the Raspberry Pi) is inexpensive, making it an excellent performance benchmark for IoT applications.

CONCLUSIONS

The design and deployment of a general-purpose and scalable multi-interface GW with three widely-adopted and popular radio communication interfaces, namely LoRaWAN, BLE, and Wi-Fi, are discussed in this work. Our multi-interface

GW may be used in a variety of scenarios, and our experimental findings, which were acquired using a COTS-based GW implementation, demonstrate that it is scalable and performs well. The created GW serves as a springboard for a number of future IoT-related initiatives. Given its open design, which is built on simple and lightweight protocols, system scalability to new interfaces should be assured, at least until one of the employed protocols (MQTT for data routing and relational DB for storage) or HW resources becomes the GW's bottleneck. Alternative interfaces, on the other hand, might be utilised to unload a (temporarily) overloaded interface. This will be one of the topics of our future study.

REFERENCES

1. Ansari, A. Y. Tong, G. A. Cowgill, L. E. Nicholls, J. P. Ramayya, R. Masina, A. Raissyan, and A. R. McQuarters, "Multi-interface wireless adapter and network bridge," Patent US8 649 386B2, 09, 2007.
2. Oniga, A. Munteanu, and V. Dadarlat, "Open-source multi-protocol gateway for Internet of Things," in 2018 17th RoEduNet Conference: Networking in Education and Research (RoEduNet), Cluj-Napoca, Romania, 2018, pp. 1–6, doi:10.1109/ROEDUNET.2018.8514136.
3. K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, "Overview of Cellular LP- WAN Technologies for IoT Deployment: Sigfox, LoRaWAN, and NB- IoT," in 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Athens, Greece, 2018, pp. 197–202, doi:10.1109/PERCOMW.2018.8480255.
4. C. Chen, M. Lin, and C. Liu, "Edge Computing Gateway of the Industrial Internet of Things Using Multiple Collaborative Microcontrollers," IEEE Network, vol. 32, no. 1, pp. 24–32, 2018, doi:10.1109/MNET.2018.1700146.
5. F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne, "Understanding the Limits of LoRaWAN," IEEE Communications Magazine, vol. 55,

no. 9, pp. 34–40, 2017,
doi:10.1109/MCOM.2017.1600613.

6. Ansari, A. Y. Tong, G. A. Cowgill, L. E. Nicholls, J. P. Ramayya, R. Masina, A. Raissyan, and A. R. McQuarters, “Multi-interface wireless adapter and network bridge,” Patent US8 649 386B2, 09, 2007.