A Platform Architecture for m-Health Internet of Things Applications

Pedro Mestre[1,2,4(✉)], Ertugrul Dogruluk[1], Carlos Ferreira[1], Rui Cordeiro[1], João Valente[1], Sérgio Branco[1], Bruno Gaspar[3], and Jorge Cabral[1,4]

[1] CEiiA Centro de Engenharia e Desenvolvimento de Produto, 4450-017 Matosinhos, Portugal
{pedro.mestre,ertugrul.dogruluk,carlos.ferreira,rui.cordeiro, joao.valente,antonio.branco,jorge.cabral}@ceiia.com
[2] Universtity of Trás-os-Montes e Alto Douro, Vila Real, Portugal
[3] Mobile Centric Architecture and Solutions, NOS Technology, Lisbon, Portugal
bruno.gaspar@nos.pt
[4] ALGORITMI Research Centre/LASI, University of Minho, 4800-058 Guimarães, Portugal
https://www.ceiia.com,https://www.utad.pt,https://www.nos.pt, https://algoritmi.uminho.pt

Abstract. In the last few years, several researchers have been focusing their research work on some specific applications of the Internet of Things, such as the Health Internet of Things (m-Health IoT). As in other IoT applications, integrating IoT devices and applications/platforms from different manufacturers/developers in Health-IoT can be challenging. Even though standards are available and used, there are many possible standards and different devices use different communication protocols and data formats. Also, the integration leads to the need to develop new ad hoc code that will fit only that particular integration. This paper presents a proposal for a platform that uses a modular architecture and enables the seamless integration of different components of an m-health IoT platform. Components can be dynamically added or removed from the platform, in run-time without impacting the already existing components and without the need of downtime of the platform to include the new components.

Keywords: m-Health · IoT · Dynamic Configuration · Seamless Integration · gRPC

1 Introduction

The Internet of Things (IoT) can be defined as an interaction between computing devices via the Internet. Based on their configuration, these devices are

A. Cunha et al. (Eds.): MobiHealth 2022, LNICST 484, pp. 168–179, 2023.
https://doi.org/10.1007/978-3-031-32029-3_16

responsible for sending and receiving data. IoT devices can cover several applications, such as automobiles, smart industries, smart cities, smart homes, and healthcare applications. Also, the IoT ecosystem has several entities such as consumers, smart devices, communication, security protocols, and platform to communicate, which are used to connect with the IoT services, e.g., healthcare applications [2,8].

Therefore, all IoT devices must be considered critical, especially those health-related. The IoT healthcare applications can provide services for hospitals, emergency transport vehicles such as ambulances, nursing, stakeholders, and allow the consumers to analyze or store the collected data.

In recent works, several authors have focused their research on health Internet of Things applications. For instance, authors of [12] studied and implemented remote health monitoring on wearable vital sensors to alert critical measurements to hospital monitors. This application is supported by cellular such as NB-IoT protocols to provide efficiency and reliability.

The work [9] studied non-contact Internet-of-Medical-Things (IoMT) systems that are implemented between cloud platforms and edge nodes for cardiopulmonary functions. IoMT application was applied to analyze the recovery performance of respiratory results for COVID-19 patients.

In [10], IoT e-health services architecture was studied to increase the data processing, availability, and low-cost management of fog and cloud computing. This work also discusses the research challenges of providing a good e-health application based on performance, availability, and accessibility to achieve the lowest maintenance cost and deployment.

The work [16] implemented an intelligent Heath-IoT platform to serve IoT devices such as wireless wearable sensors and medicine packages for healthcare applications. Das et al. [5] proposed an IoT-enabled health automated monitoring platform that is used to monitor emerging conditions of the patient. This platform provides various vital conditions and parameters such as an electrocardiogram (ECG), blood oxygen level, heartbeat, and body temperature for the patients.

In this area of Heath-IoT, as in other IoT areas, as we can see from the above implementations, there are many possible types of devices and applications. Consequently, several data formats, communication technologies, and protocols can be used by IoT devices and platforms. Integration of devices from these different manufacturers/developers with applications and platforms from diverse providers/developers is a challenge.

In fact, integration can be a very time-consuming task and not as seamless as it would be desirable. Either because of the different formats and protocols used and because the architectures might be different or new code must be written, not only for data conversion but also to code link between the components to be integrated.

This is a problem that also arises in health IoT services, where equipment from different manufacturers need to be integrated, for example, in hospital patient monitoring software or in software to track, in real-time, the location of emergency services (e.g., ambulances).

Even if the standards are used, there are standards for different types of applications, there are standards that are niche specific, and when researchers need to integrate applications from different areas, sooner or later, they end-up with an integration problem to solve. For instance, if the civil authorities and the hospitals need to know the location of ambulances in real time, we might be dealing with several and various standards that need to be integrated.

From a simplified point of view, a seamless way to integrate devices and applications is required. This integration must have a low impact on the already running platforms and require a minimal need for an integration platform. This paper presents a modular architecture (that can be implemented using micro-services) that allows the seamless integration of different components of an IoT platform (including those that are used in the case study in this paper, i.e., Health Services) that allows in run-time new components (e.g., adding a service or feature) to be added and establishment of links between components.

This work is organized as follows: Introduction, the current section, summarizes the related works within the motivation of this paper for the IoT healthcare services and presents the motivations and objectives. Section 2 summarizes the literature on IoT healthcare transport services and IoT platforms. Section 3 introduces the main contribution of this work by presenting the IoT platform architecture. Section 4 presents the implementations and results related to the m-health IoT platform. Finally, Sect. 5 concludes this work.

2 Internet of m-Health Things

In recent years an increasing demand for wirelessly connected devices and IoT applications is evolving new communication technologies. Especially emerging health-related IoT applications use wireless sensor networks (WSN) to manage, sense, and monitor the application needs. For instance, several WSN are defined to increase IoT device battery lifetime, reliability, communication coverage, and efficient data rate. The latest LPWAN technologies can be identified as LoRaWAN, Sigfox, and the fifth generation (5G) family, including narrowband IoT (NB-IoT) and LTE-M. These technologies are selected depending on the m-health IoT application, or device [6,7].

Even though these technologies all use standard protocols (TCP/IP) for transmitting information, the upper layer protocols (Application Layer) can be different. Also, the way these devices use the network may impact the way protocols and platforms have to work. For example, NB-IoT and LTE-M have different latency values (around 300 ms with good network coverage for NB-IoT and 50–100 ms for LTE-M), which can impact how these devices are integrated with an application, for example, IoT healthcare.

The survey [14] studied the challenges with frameworks and IoT trends in IoT Forensics by highlighting IoT security practices. The work also stated that the IoT healthcare applications market size is placed as 3rd by following Smart Cities and Industrial IoT applications market sizes.

2.1 Mobile Health (m-Health)

In IoT architectures, mobile computing, emergency services, and medical IoT sensors technologies are defined as the Internet of m-Health Things in healthcare-related devices. To create reliable m-health applications, the latest low power-based design (LPWAN), such as narrowband NB-IoT with 4G and the latest 5G communication technologies, are engaging in several m-health related IoT applications. For instance, wearable sensors are used to monitor patient health conditions, trace emerging services (ambulance, firefighter, patient tracking, and more services) to serve the patient as soon as possible, and so on [2].

Smart monitoring devices in IoT infrastructures provide health services. For instance, in the m-health IoT approach, health-related information such as patient records, hospital billing, emerging transportation management, insurance records, etc. Because of these critical systems, keeping the integrity of confidentiality, data security, and data privacy are also stated as important, as described in [1].

2.2 Internet of m-Health Things Platform

Considerable m-health applications are proposed for various applications, e.g., iMedBox [16], wearable IoT application for safety and health use cases [15], low-powered based NB-IoT devices for various sensing technologies (humidity, temperature, etc.) [3,11], Cardiopulmonary monitoring [9], provide healthcare solution called eHealthCare defining the non-adherence to health medication for elderly individuals by using NB-IoT and artificial intelligence algorithms [13], and other m-health related IoT applications.

Managing distinctive IoT device brands, protocols, and database sets can be a challenge to communicating with other emerging m-health IoT applications. For instance, hospital records (billing, data, medical records, etc.) must be unified or simplified with other related IoT services (insurance company, accessible information for doctors, database structure, and more).

Figure 1 illustrates the Internet of m-health Things platform architecture with its main components. In this architecture, the m-health IoT platform is mainly responsible for the unification of the IoT services between the m-health IoT hardware (e.g., NB-IoT connected devices, such as patient monitoring, hospital records, etc.) and the m-health IoT applications (e.g., hospital billings, emerging messages, emerging transportation, health software, etc.).

To obtain such an m-health IoT platform, several entities are required. For instance, the API translation is responsible for translating different programming languages or database sets into one readable place. Also, other services such as bare-metal server configurations, data security, data management, Kubernetes pods, and databases are the entities used for robust communication, platform integrity, and load balancing of the platform services.

The next section presents a proposal for a platform to support m-health IoT applications.

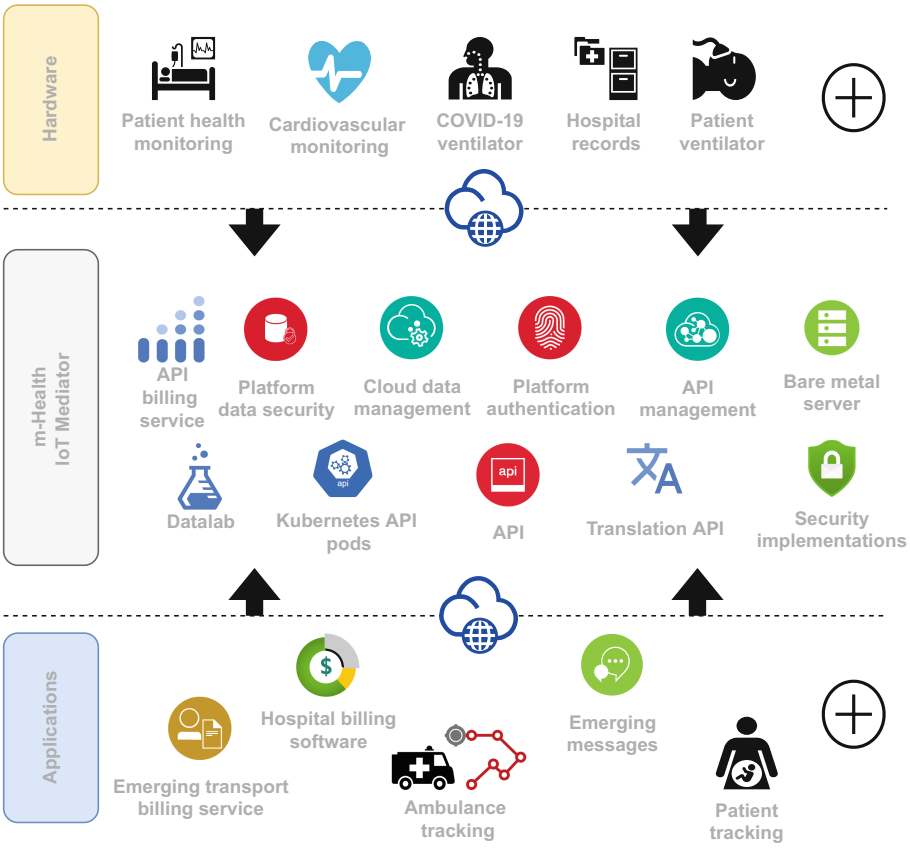

Fig. 1. Internet of m-health Things Platform Architecture

3 Internet of m-Health Things Platform (IoT-mP) Architecture

The rationale behind the platform architecture is the ability to link any data source(s) to any data sink(s), independently of the data formats and/or protocols that are used. It is an objective of the proposed platform to allow, dynamically, and at run-time, to link any data source to any data consumer, without "hard-coding" these links in the software application, or even to restart it. This means that we can have a set of devices that connect to the platform using CoAP (Constrained Application Protocol [4]), and another set that uses MQTT (Message Queuing Telemetry Transport). The data from these devices are simultaneously sent to a database (e.g., MongoDB) for storage, and some are sent to Artificial Intelligence (AI) algorithm for processing. Later, during the lifetime of the platform, the data can also be sent to another consumer without impacting its already existing components. Because of its modular architecture, the platform

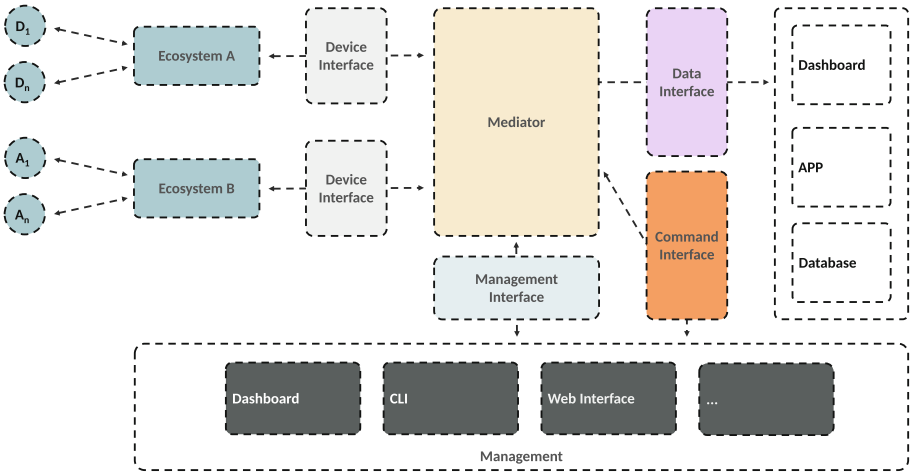

Fig. 2. Health IoT Platform structure

is ready to use any protocol, even those that are still not available at the time of its development and/or deployment.

As a case study, the platform was built having as a starting point a custom protocol (called Black-Wing), the same protocol as used in [11], and later, without the need to modify the architecture, MQTT and CoAP were also integrated into the platform, and tested successfully.

One can describe the developed platform either from the point of view of its functional components or through a set of interfaces that were developed to allow the flow of information, as it is shown in Fig. 2.

3.1 Platform Functional Architecture and Components

From the functional point of view, the architecture comprises five components, which provide the main functions of the architecture. To be noticed that a software component can implement more than one function. These five functions are:

- Mediator: the core of the platform;
- Device Ecosystems: that provide interface with the IoT devices;
- Data Consumers: which are the sinks of the data;
- Command Producers: that generate commands to be sent to the IoT devices.

Mediator. In the core of the platform, there is a component called the Mediator, which is responsible to know what paths data must follow. These data paths are set dynamically at run-time, allowing the data paths to change to accommodate any change needed, such as new data sources, new data sinks, new processing algorithms, etc.

The Mediator receives/sends data from/to IoT devices. Typically the information sent by IoT devices is data from sensors and configuration values, and data sent to the devices include actuation commands, configuration data, and firmware updates.

Device Ecosystems. Device Ecosystems (DE) are entities that provide the infrastructure for the devices to operate using their own protocols and data formats. For example, a set of devices can use MQTT to send its data and receive commands from the IoT platform. Besides providing the infrastructure for devices to communicate, DE is the bridge between the Mediator and the devices. When data is received from a device, it is the responsibility of the DE to send it to the Mediator. Also, when a command has to be sent to a device, the DE receives that command and forwards it to the device. The devices do not need to be aware of the existence of the Mediator, which allows the integration of already existing platforms and protocols. The Mediator does not need also to be aware about the communication details of the devices, making it device/platform/protocol agnostic. The DE might also include some data adaptation or translation functions to allow the data to arrive correctly to the consumers. However, it is not mandatory to have the data translation in this layer. In fact, any component of the platform can provide these services.

Data Consumers. After processing the messages arriving from the Device Ecosystems, the Mediator forwards data to their consumers. A consumer can be any application to which data has meaning (Dashboard, Database, APP, etc.). As previously stated, these consumers are defined in run-time, and therefore consumers can be added or removed without impacting other consumers. We can have multiple consumers that receive the same data.

Command Producers. Besides receiving data from devices, the mediator also handles instructions that are sent to the devices. These instructions can be actuation commands, configuration instructions, or even firmware updates, and are generated by entities called Command Producers (CP). When a CP needs to send a command to a device, it sends it to the Mediator, that will forward the request to the corresponding DE. When (if any) answer from the command is sent back to the Mediator, it will be forwarded to the origin Command Producer.

3.2 Interfaces

Although a sophisticated technology was used for communication between the components and functions of the platform, a set of "standard" technology and protocol-independent interfaces were specified. These interfaces allow communication between the platform components, as well as extend it and add new features to the platform or even use it in different target applications. This means that, for example, new algorithms that calculate which emergency team

should go to a call can be easily added, without impacting the other elements of the platform, or that patient monitoring applications can be built on top of this platform.

As a case study, these interfaces were implemented using gRPC, but they can be implemented using any other technology. For this case, the interface that is used for management was also implemented using REST Web-services, to be compatible with web browsers.

There are four interfaces used to interconnect the functional components:

- Device Interface (DE-Interface);
- Data Interface (D-Interface);
- Command Interface (C-Interface);
- Management Interface (M-Interface).

Device Interface. Used for communications between the Device Ecosystems and the Mediator, defines all functions needed by the Device Ecosystem to forward data from devices to the Mediator, and to send back to the Mediator the response to the instructions received from the Command Producers.

Data Interface. Data from devices is sent to the Data Consumers using the Data Interface. Upon the arrival to the Mediator, the rules established by the administrator are matched against the ID of the device, if there is a match, data is sent to the Consumer.

Command Interface. In the opposite direction (in comparison to the Data Interface), requests for the devices are sent to the Mediator using the Command Interface. This is also the interface that is used by the Device Ecosystems to send the response to these requests to the Mediator.

Management Interface. It is the Management Interface that allows the administrator to setup the rules in the Mediator. The administrator, using, for example, a Command Line Interface (CLI), a Web Interface, or any other tools, can add/remove Device Ecosystems, Devices, Data Consumers, and Command Producers, as well as define a set of links between these components. How the data will flow inside the Mediator is set using this interface.

4 Implementation and Results

4.1 Implementation

The above-presented architecture can both be implemented in a monolithic or in a distributed way. While the first can be interesting in very specific situations, where a very low memory footprint is one of the main constraints (e.g., in

a low-level embedded system), in most computing systems, a distributed approach using micro-services is better. Using a distributed approach allows us to achieve the desired run-time flexibility, and we can, if needed, use load-balancing strategies between the nodes of the platform.

Therefore the distributed approach, using a micro-services architecture, was chosen. In this architecture, each node is one of the functional components of the architecture. To provide the computational power, a Kubernetes cluster is being used. Kubernetes (K8S) provide us with features related to scaling and self-healing that are very useful in a system with a low tolerance for failure. Besides that, K8S provides a very flexible way to automate the deployment of the components and manage their life cycle.

To implement the Mediator, Java was the chosen programming language. Even though Java was chosen, any otter programming language with a stable support gRPC can be used. However to be noticed that performance was always the top requirement when implementing it, because it needs to route as fast as possible all the messages that are sent to it. Its current speed and memory footprint are suitable to run it in some of the mid-level computational platforms such as Raspberry Pi 4 and Orange Pi 0. The final performance and memory needs will depend on the number of rules, the number of associated nodes, the number of messages arriving at the Mediator and the available bandwidth.

4.2 Tests and Results

Preliminary tests with these platforms allow to conclude that the platform can be used from the bottom to the top in a distributed architecture for IoT systems. This makes this architecture also very useful for edge computing and data aggregation.

To test the proposed platform, it was implemented a test platform that consists of:

- A Device Ecosystem that receives data from devices using CoAP;
- A Device Ecosystem that receives data from devices using the protocols described in [11];
- One Mediator;
- One Data Consumer that receives data from the devices and sends it via web-socket to a Web Application;
- An example web application (Fig. 3) that shows the data.

For CoAP devices, a simulator was used, built in Java, which sends a set of pre-recorded coordinates, was used. For the other protocol, it was used a CPS Cyber-Physical System), a device that is under development, and an Android application. The data sent by the devices are the GPS coordinates.

While the simulator allowed to do performance and stress tests to the platform, the use of the CPS device and the Android Applications allowed for monitoring, in real-time, the location of the system.

Using an 11th Gen Intel(R) Core(TM) i7-1185G7 @ 3.00 GHz PC, with 16 GBytes of RAM, running Ubuntu Linux 22.04 and Java 11, without any load

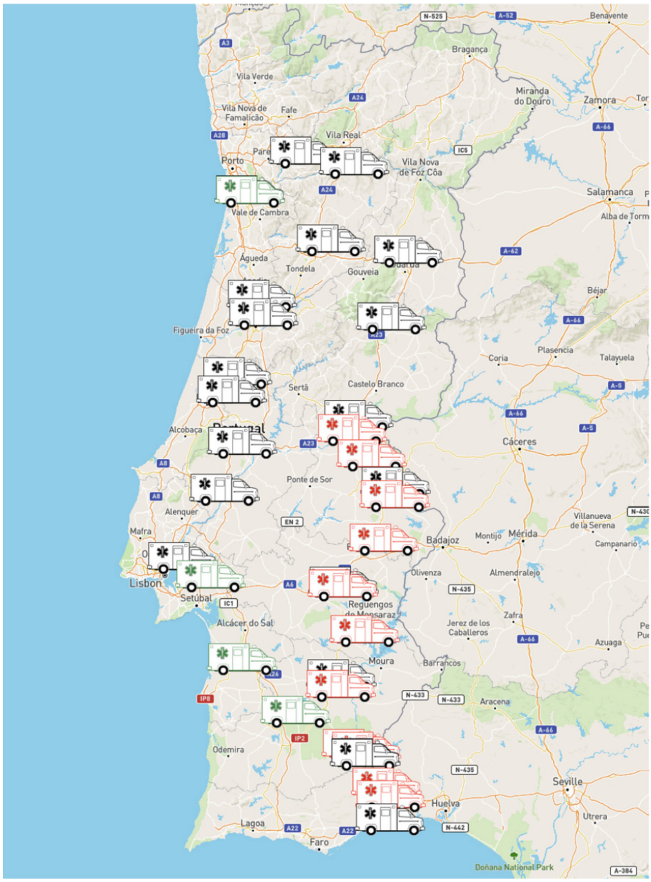

Fig. 3. Example of an output dashboard.

balancing strategy it was possible to achieve the following indicative values, when flooding the system with a message:

- Send messages between Device Ecosystem and Mediator (messages are validated and then dropped), around one million messages per minute;
- Send messages from a device (using CoAP) to a DE, the DE then sends the message to the Mediator that validates the message and then drops it, around 380K messages in a minute;
- The same as above, but the messages are sent to a Consumer that decodes the message and makes it available to the front-end (as the one presented in Fig. 3), around 285K messages per minute.

5 Conclusions

To make the integration of services even more seamless, besides the specification of the interfaces (protobuf for gRPC and OpenAPI for REST) that are made available to the customers that want to integrate products with the platform, currently, it is also under development a framework to help build Device Ecosystems, Data Consumers and Command Producers. This framework, developed both in Java and Python, is currently under test by a third party, which had already built their own independent IoT platform and are now integrating it with the platform presented in this paper.

Obviously that the scope and applications of the platform is not limited only to the application presented here. In fact, it can be applied to any IoT application, and it has also been tested in applications related to city mobility. Future developments of this platform include the inclusion of a new functional component, and respective interface, that is going to provide data translation (this component is now integrated in other functional blocks, removing it into a standalone functional block will allow an improved re-usability of the components).

Regarding the concerns of security and data protection, the communication between components uses gRPC, which can be done over HTTPS, providing data encryption. Regarding data protection, the platform per se does not store any data from devices. That concern is delegated to the providers of the applications, e.g., Data Consumers, because the Mediator is message content agnostic and therefore does not need to store any user data.

Acknowledgements. Project "(Link4S)ustainability - A new generation connectivity system for creation and integration of networks of objects for new sustainability paradigms [POCI-01-0247-FEDER-046122—LISBOA-01-0247-FEDER-046122]" is financed by the Operational Competitiveness and Internationalization Programmes COMPETE 2020 and LISBOA 2020, under the PORTUGAL 2020 Partnership Agreement, and through the European Structural and Investment Funds in the FEDER component.

References

1. Almotiri, S.H., Khan, M.A., Alghamdi, M.A.: Mobile health (m-Health) system in the context of IoT. In: Proceedings - 2016 4th International Conference on Future Internet of Things and Cloud Workshops, W-FiCloud 2016, pp. 39–42 (2016). https://doi.org/10.1109/W-FiCloud.2016.24
2. Bhuiyan, M.N., Rahman, M.M., Billah, M.M., Saha, D.: Internet of things (IoT): a review of its enabling technologies in healthcare applications, standards protocols, security, and market opportunities. IEEE Internet Things J. **8**(13), 10474–10498 (2021). https://doi.org/10.1109/JIOT.2021.3062630
3. Borges, M., Paiva, S., Santos, A., Gaspar, B., Cabral, J.: Azure RTOS ThreadX design for low-End NB-IoT device. In: Proceedings - 2020 2nd International Conference on Societal Automation, SA 2020 (2020). https://doi.org/10.1109/SA51175.2021.9507191

4. Bormann, C., Lemay, S., Tschofenig, H., Hartke, K., Silverajan, B.: CoAP (constrained application protocol) over TCP, TLS, and WebSockets. Technical report (2018). https://doi.org/10.17487/RFC8323, https://www.rfc-editor.org/info/rfc8323
5. Das, A., Katha, S.D., Sadi, M.S., Ferdib-Al-Islam: an IoT enabled health monitoring kit using non-invasive health parameters. In: 2021 International Conference on Automation, Control and Mechatronics for Industry 4.0, ACMI 2021 (July), pp. 8–9 (2021). https://doi.org/10.1109/ACMI53878.2021.9528227
6. Kanj, M., Savaux, V., Le Guen, M.: A tutorial on NB-IoT physical layer design. IEEE Commun. Surv. Tutor. **22**(4), 2408–2446 (2020). https://doi.org/10.1109/COMST.2020.3022751
7. Khalifeh, A., Aldahdouh, K., Darabkh, K.A., Al-sit, W.: A survey of 5G emerging wireless technologies. In: 2019 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET), pp. 561–566 (2019)
8. Lee, E., Seo, Y.D., Oh, S.R., Kim, Y.G.: A survey on standards for interoperability and security in the internet of things. IEEE Commun. Surv. Tutor. **23**(2), 1020–1047 (2021). https://doi.org/10.1109/COMST.2021.3067354
9. Liu, J., Miao, F., Yin, L., Pang, Z., Li, Y.: A noncontact ballistocardiography-based IoMT system for cardiopulmonary health monitoring of discharged COVID-19 patients. IEEE Internet Things J. **8**(21), 15807–15817 (2021)
10. Monteiro, K., Rocha, E., Silva, E., Santos, G.L., Santos, W., Endo, P.T.: Developing an e-health system based on IoT, fog and cloud computing. In: Proceedings - 11th IEEE/ACM International Conference on Utility and Cloud Computing Companion, UCC Companion 2018, pp. 17–18 (2019). https://doi.org/10.1109/UCC-Companion.2018.00024
11. Paiva, S., Branco, S., Cabral, J.: Design and power consumption analysis of a NB-IoT end device for monitoring applications. In: IECON Proceedings (Industrial Electronics Conference), vol. 2020-Octob, pp. 2175–2182 (2020). https://doi.org/10.1109/IECON43393.2020.9254374
12. Pathinarupothi, R.K., Durga, P., Rangan, E.S.: IoT-based smart edge for global health: remote monitoring with severity detection and alerts transmission. IEEE Internet Things J. **6**(2), 2449–2462 (2019). https://doi.org/10.1109/JIOT.2018.2870068, https://ieeexplore.ieee.org/document/8464257/
13. Pinto, A., Correia, A., Alves, R., Matos, P., Ascensão, J., Camelo, D.: eHealthCare - a medication monitoring approach for the elderly people. In: Gao, X., Jamalipour, A., Guo, L. (eds.) MobiHealth 2021. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 440, pp. 221–234. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-06368-8_15
14. Stoyanova, M., Nikoloudakis, Y., Panagiotakis, S., Pallis, E., Markakis, E.K.: A survey on the internet of things (IoT) forensics: challenges, approaches, and open issues. IEEE Commun. Surv. Tutor. **22**(2), 1191–1221 (2020). https://doi.org/10.1109/COMST.2019.2962586
15. Wu, F., Wu, T., Yuce, M.R.: Design and implementation of a wearable sensor network system for IoT-connected safety and health applications. In: IEEE 5th World Forum on Internet of Things, WF-IoT 2019 - Conference Proceedings, pp. 87–90 (2019). https://doi.org/10.1109/WF-IoT.2019.8767280
16. Yang, G., et al.: A health-IoT platform based on the integration of intelligent packaging, unobtrusive bio-sensor, and intelligent medicine box. IEEE Trans. Industr. Inf. **10**(4), 2180–2191 (2014). https://doi.org/10.1109/TII.2014.2307795