VITASENIOR-MT: A distributed and scalable cloud-based telehealth solution

Diogo Mendes
VITA.IPT Lab
Polytechnic Institute of Tomar
Tomar, Portugal
diogo.santos.mendes@ipt.pt

Renato Panda
VITA.IPT Lab
Polytechnic Institute of Tomar
Tomar, Portugal
renato.panda@ipt.pt

Pedro Dias
VITA.IPT Lab
Polytechnic Institute of Tomar
Tomar, Portugal
diaspedro@ipt.pt

Dário Jorge
VITA.IPT Lab
Polytechnic Institute of Tomar
Tomar, Portugal
jorge.dario@ipt.pt

Ricardo António
VITA.IPT Lab
Polytechnic Institute of Tomar
Tomar, Portugal
ricardma1996@gmail.com

Luis Oliveira
VITA.IPT Lab
Polytechnic Institute of Tomar
Tomar, Portugal
loliveira@ipt.pt

Gabriel Pires
VITA.IPT Lab
Polytechnic Institute of Tomar
Tomar, Portugal
gppires@ipt.pt

***Abstract*— VITASENIOR-MT is a telehealth platform that allows to remotely monitor biometric and environmental data in a domestic environment, designed specifically to the elderly population. This paper proposes a highly scalable and efficient architecture to transport, process, store and visualize the data collected by devices of an Internet of Things (IoT) scenario. The cloud infrastructure follows a microservices architecture to provide computational scalability, better fault isolation, easy integration and automatic deployment. This solution is complemented with a pre-processing and validation of the collected data at the edge of the Internet by using the Fog Computing concept, allowing a better computing distribution. The presented approach provides personal data security and a simplified way to collect and present the data to the different actors, allowing a dynamic and intuitive management of patients and equipment to caregivers. The presented load tests proved that this solution is more efficient than a monolithic approach, promoting better access and control in the data flowing from heterogeneous equipment.**

Keywords—Telehealth, Cloud Computing, Fog Computing, IoT, Microservices.

I. INTRODUCTION

In the European Union (EU), demographic trends point to a rise in aging population, with an estimated growth of 74% in the population over 65 years of age until 2060, and a decline of 42 million in the working-age population [1]. This fact, together with the high desertification of the countryside caused by the rural exodus, leads to an increased number of isolated elderly people. Coupled with this reality is the high number of age-related diseases, such as type 2 diabetes and cardio-cerebrovascular diseases, or immobility conditions due to fractures caused by falls. As an example, in 2015 there were almost 49 million people diagnosed with cardiovascular diseases in the EU [2]. Continuous monitoring and follow-up of the elderly can help to mitigate such situations.

Telehealth has become an important research and development topic in the area of Cloud systems, promoting the use of mobile devices in healthcare services for more vulnerable populations [3]. To address the abovementioned issues, we propose a telehealth platform, VITASENIOR-MT, that allows to monitor the patients' clinical parameters as well as the environment where they live [4]. The elderly interact with the system through a television, a device that is already familiar to them, thereby avoiding the introduction of new technology which could lead to constraints and rejection of the system. A central device, that we call VITABOX, supports the interaction between the user and the television. It is also responsible to collect data from the medical devices and environmental data from the wireless sensor network, as well as to send all data to the Cloud infrastructure (see details in [4]).

Cloud computing is a well-defined and consolidated concept to get hosting for our telehealth system with distributed processing power without the need to know the hardware physical location. To better organize the business logic and information storage, it is necessary to model all data in a homogeneous way, abstracting all devices dissimilarity and impose access restrictions and personal data protection with different access layers. The entire process starts in the VITABOX with the pre-processing of the sensors' collected data, followed by transmission to the cloud, where it is stored and processed to later notify the patient and their caregivers of possible issues. Data information is also presented to the users, namely, informal caregivers, doctors or the patients themselves.

In this paper we describe the implementation of VITASENIOR-MT telehealth solution, which aggregates data from distinct equipment, based on different protocols. In addition, it also provides a platform to remotely monitor the environment and biometric state of the users in a plug-and-play way, without requiring special technological knowledge. With that aim, we resort to concepts such as fog computing and microservices to provide a secure, scalable and flexible system where each

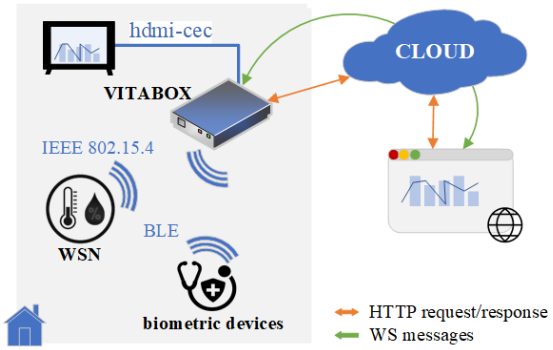

Fig. 1. General architecture of the VITASENIOR-MT project.

module can be scaled as required to accommodate more users, equipment or analytical processing needs, independently.

II. State of the Art

Cloud Computing promotes the provision of new business models for the providers, such as SaaS (Software as a Service) and PaaS (Platform as a Service) [3]. Unlike the use of the classic IaaS (Infrastructure as a Service) model, in which the provider is only responsible to provide the physical processing power, storage and network resources, leaving to client the configuration and maintenance responsibility. In SaaS it is possible for the client to use an application in a Cloud environment without requiring any development (e.g., e-mail and calendar tools). In PaaS, the provider sets all the ecosystem required for a given technology (operating system, framework and deployment tools) and the client is required only to develop over the provided structure [3].

The PaaS model give us the opportunity to set up the initial system requirements, with the possibility of scaling them as needed in the future. As a result, the user is billed only on the resources needed, in a model known as pay-as-you-go (PAYG). Moreover, all tasks related with guaranteeing system stability, data backups, updates, load balancing and the risks of hardware failures are delegated to the Cloud provider. This model also allows for some independence from the Cloud provider. After all, we are the code owners and thus it is possible to quickly migrate all the system to another provider without compromising customer service [5]. Despite the advantages of Cloud Computing, there are still some limitations of scope, such as the Internet of Things technologies (IoT) and private systems, often causing security holes in the access to the private network. The major issue in the integration between IoT and Cloud is the interoperability between the sensors and the web services. Then, it is necessary to use a service to translate the different vocabularies present in the various communication protocols and technologies. That process is referred to as data semantification, formatting the data with specific properties or tags to get a unified architecture [7]. There are some projects to address this interoperability problem, but they present a solution limited to a specific technology, such as the Watson IoT Platform on the IBM Cloud [8] or the AWS IoT Core from Amazon Web Services (AWS) [9], which are limited to MQTT based devices.

To promote full communication between different types of applications, a new concept has been introduced, the Fog Computing [3]. In this architecture all the aggregation logic to heterogeneous devices from the IoT network is transferred to a layer usually located at the edge of the Internet. Taking advantage of this processing power, it is possible to model the data in an acceptable format to the Cloud business logic, and to ensure the filtering of the access requests by new devices to the IoT network. With this, we can take full advantage of the IoT, enjoying the proximity to users and the environment that surrounds them, and of the Cloud, with the virtually almost unlimited processing and storage capabilities.

Some projects similar to VITASENIOR-MT have been proposed, implementing Cloud systems to remotely monitor health parameters. For example, in [6] blood pressure and electrocardiogram devices are connected to a platform that associates users with healthcare professionals (e.g., doctors and nurses) and let them check the results in an Android application. However, it does not collect home environmental data as in our telehealth solution. The major limitation of their project is that the Cloud architecture was not designed to evolve over time in new functionalities like as devices and users' management. This could be achieved by implementing a microservices architecture and developing a Backoffice where the administrators could manage equipment, being able to add new device types or alter the threshold rules to abnormal values, as we are proposing in VITASENIOR-MT.

III. General Architecture

The VITASENIOR-MT project is divided in 3 parts (Fig. 1):

- Wireless Sensor Network (WSN) and Bluetooth medical devices: provides the readings of the home environment and biometric sensors;
- VITABOX: promotes patient interaction with the TV, requests data from the sensors network, performs pre-processing of the data and sends it to the Cloud;
- Cloud: analyses the data, provides a web interface to the users, such as caregivers, healthcare professionals, patients and administrators, and notifies them of any irregularities.

The WSN is organized in a tree topology, promoted by Routing over Low Power and Lossy Networks (RPL), and the communication between the nodes is based on the IEEE 802.15.4 standard. The root node is assigned as Border Router (BR), which connects the WSN network and the VITABOX through a tunslip6 bridge. It allows access to new nodes, which were previously defined from the Cloud interface, and enables the access to the resources from the external network, through the Constrained Application Protocol (CoAP). This way the VITABOX is able to obtain temperature, humidity, carbon monoxide and carbon dioxide readings, in JSON format, from each node.

The VITABOX collects data from the biometric sensors using Bluetooth Low Energy (BLE). These sensors provide blood pressure, weight, oximetry or blood glucose measurements at the patient's initiative. There is

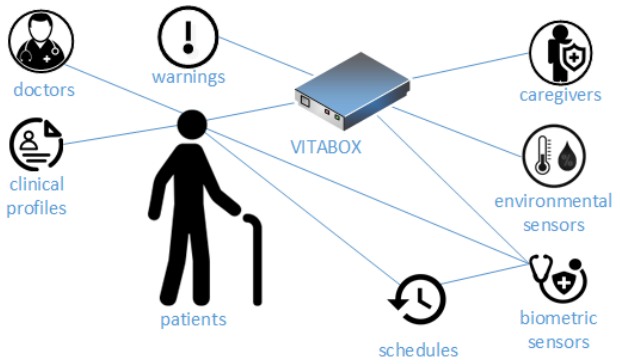

Fig. 2. Object relationships in VITASENIOR-MT system.

also a bracelet for measuring heart rate and steps, which are collected automatically by the VITABOX, when it senses the bracelet signal within its range. VITABOX also centralizes user-TV interaction features, allowing the user to initiate certain exams, view his/her historical data and receive alerts in dangerous situations. All requested data, either from WSN or medical devices, are stored in a local database with a validity of 24 hours for environmental records and 1 month for biometrics records. Finally, the records are forwarded to the Cloud, where protocols and conditions are applied. If abnormal values are detected, the persons responsible for the elderly and the VITABOX itself are notified.

The relationships between VITABOXes, sensors, patients and the different roles of users, such as the doctors and the caregivers, are also defined in the Cloud (Fig. 2). Thereby, it is possible to set different permissions to users to access data, edit profiles, schedule exams or manage equipment. For example, only the doctors associated with a specific patient can define his/her clinical profile and schedule exams. To this end, the caregiver needs to indicate the respective doctor and associate the biometric device to the patient. These features can be performed through a web interface available for any device with a modern browser.

A set of sensors' thresholds and clinical profiles are defined in the Cloud to generate warnings or recommendations, as well as to notify the caregivers when some irregularity occurs. To promote a faster reaction from house dwellers, these thresholds are sent to the WSN nodes through the VITABOX. This way, patients may act immediately, for example, opening a window if the CO is too high. As this process is done locally, this ensures the autonomous operation of the VITABOX even when no connection can be established with the Cloud infrastructure. The same is not applied to biometric values because to be interpreted and validated, these data require more computational power not offered by VITABOX hardware and need also the involvement of more actors of the overall system.

IV. Implementation

A. *Fog Computing*

One of the problems when designing an IoT/Cloud architecture is the routing of sensor data to the Cloud, namely the lack of interoperability between different technologies used to gather data. Due to this reason, it was necessary to implement an intermediate proxy that could

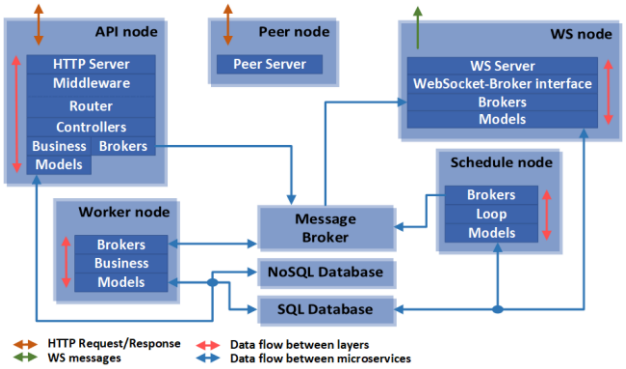

Fig. 3. Cloud microservices architecture.

aggregate and connect all the equipment and process the data, so that the data followed the same schema pattern [10]. This task is carried out by the VITABOX, because it is located at the edge of the local network, facilitating the distribution of the system and the collection of data from authorized devices. All records received in the VITABOX are modeled into a standard JavaScript Object Notation (JSON) and forwarded to the Cloud, thus abstracting the Cloud from the devices heterogeneity and avoiding the authentication of each equipment. Since the source of the data is the VITABOX, that is already authenticated, it only accepts the equipment that was previously registered in the Cloud. Another particularity of our implementation is that sensor data are collected by the initiative of the VITABOX, unless a sensor detects a value out of the predefined range. In this case, the sensor triggers an event to the VITABOX. This way it is possible to configure the periodicity of data requests to sensors.

B. *Scalability*

Cloud Computing is known by its almost unlimited processing and storage capabilities, however that has an associated cost and thus it is necessary to design the services architecture to get the most out of the Cloud. The model adopted was the PaaS provided by IBM Cloud [11] allowing horizontal scaling of the system.

Five different processing nodes were developed (see Fig. 3):

- API node: Web Application Programming Interface (API) that promotes the data accessibility, via Representational State Transfer (REST), providing the create, read, update and delete actions (CRUD) to manage all data generated by the system;
- WS node: establishes the WebSocket (WS) connections between VITABOXes and browser clients to send messages in real-time. These are sent by the server initiative, used in tasks such as an update to the VITABOX or a warning to the caregivers about some irregularity on the biometric values of a patient;
- Worker node: executes intensive processing actions such as applying filters to data, compare sensors data to predefined thresholds and clinical profiles, generating warnings from abnormal values, and updating VITABOXes and sensors timestamps;

- Schedule node: runs all the functions that execute in loop, e.g., checking if scheduled exams were performed;
- Peer node: registers the clients that intend to connect through a Real Time Communication over the Web (WebRTC).

The communication between the microservices is performed by the RabbitMQ message queue system [12]. When the API node receives sensor records, they are forwarded to the queue broker. The Worker node is listening the broker, and applies the appropriate rules to the received data, verifying whether the sensor values are as expected. If a data abnormality occurs, an alert is registered and sent to the clients through the WS node.

The adopted architecture allows the system to scale according to access types. If, for example, the number of VITABOXes increases dramatically, more sensor data will be generated, which forces an increasingly large number of Worker nodes to be instantiated in order to analyze them. If, on the other hand, an increase of accesses by the caregivers occurs, then more API nodes can be allocated to not compromise the data access. If even more users are associated to VITABOXes, then more WS nodes are required to ensure real-time warnings. Thereby, we are sure that the Schedule node will not need the balancer because the loop time will dynamically change as the number of scheduled actions grows.

Not only the message broker functions as a bridge between the microservices, but it also provides the publish/subscribe (Pub/Sub) used by the WebSockets. This allows them to scale horizontally with the WebSocket node, without restricting access to alerts by the clients. That is, regardless of the WebSocket node type to which the client connects, it will always receive the messages addressed to itself. The message channels that the clients subscribe are related to all VITABOXes to which they have permission to access the data, guaranteeing the privacy in the system.

C. Layered structure

One major concern in Cloud architecture is to guarantee the best latency to requests. We planned and implemented several mechanisms to comply with our system performance requirements. The development platform adopted was NodeJS [13]. Although JavaScript (JS) is an interpreted language, NodeJS runs on Google's V8 engine that compiles JS directly into machine code and optimizes parts of the code in real time [14]. To achieve the most out of the compiler, the code follows a layered pattern [15] enabling the reuse of code, a better organized structure and optimized functions instantiation in hidden classes [14], maintaining them to a minimum. This promotes better long-term optimization in the JIT (just-in-time) of the most requested pieces of code (hot functions). Four backend applications were developed for different purposes (Fig. 3). The defined layers were:

- HTTP Server: represents the starting service to listen clients' requests;
- Middleware: first code executed on every request, that defines accepted headers, validates the authentication tokens and extracts the client idiom;
- Router: interprets the requested route and version and forwards the request to the respective controller;
- Controller: filters the access according to the user's permissions, collects the parameters sent, and calls the functions of the business logic;
- Business: defines CRUD functions and data analysis to business logic;
- Broker: establishes communication with a message queue server, and performs the messages sent and received;
- Model: defines the data models and establishes the connection to the databases;
- WebSocket server: establishes the WebSockets with the clients;
- Broker-WebSocket interface: establishes the connection between the WebSockets and the messages queues;
- Peer Server: supports the peer discovery by the clients register.

D. Database

The Cloud infrastructure is continuously receiving sensors' data generated by the various VITABOXes. Thus, a database service designed for high frequency reads and writes of large volumes of data is required. For this reason, it was decided to use a non-relational database, since these have a linear search capability and allow to scale horizontally. The non-relational database adopted was the MongoDB [16], a document-oriented database that allows to deal with the constant registration of user logs, sensor values and warnings.

The drawbacks of this solution are the lack of data organization, since NoSQL services are schemaless, not requiring a structure for the saved data, and the lack of maturity in ACID (Atomicity, Consistency, Isolation, Durability) transactions. These characteristics may represent a flaw to the users' data safety. Thus, the addiction of a relational database, such as MySQL [17], is justified to guarantee the accounting process and relationships of the VITABOXes with users, sensors and patient profiles.

E. Access

For the user to access the system, the API Web service provides the documentation files and the web application interpreted by the browsers. The documentation is generated through the apiDoc libraries [18], forming a web page based on API source code annotations, defining in each RESTful route the method to be used, the parameters and headers to send and the possible responses.

To free some processing power from the Cloud, instead of rendering the pages on the server (server-side view), the web application is being developed with Vue.js [19], a progressive JavaScript framework for frontend

development. This approach allows for a better project structuring, making the development of the frontend independent of the development of web services in the Cloud. This way, all data processing for graphics displaying and interfaces creation are transferred to the client, feeding on JSON structures provided by the web API. This increases Cloud performance by avoiding the waste of computational resources in rendering web pages and decreases network traffic, since JSON data packets are smaller than transferring full HTML files. The choice of Vue.js facilitates bidirectional communication between components, as well as listening for events between components without the need of direct dependence.

The frontend also supports WebRTC, allowing videocalls between clients (patients from VITABOXes and caregivers/doctors from the web application) over a Peer-to-Peer (P2P) communication [20]. This solution reduces the latency typically introduced by an intermediary server, since the clients only need to register in a Peer Server to coordinate the connection between peers and to cope Network Address Translators (NAT) and firewalls. Another positive feature of this architecture is the ability to deploy internationalization libraries, local storage and access to other web services without having to overload the Cloud, since these functionalities are all supported by the client.

F. Security

According to the General Data Protection Regulation (GDPR), all data that directly or indirectly identify a user must follow a set of technical and organizational measures to guarantee access levels security [21]. Therefore, several mechanisms were implemented, from communication to storage, to avoid undue access to critical information. All communication connections are established over secure channels (https, wss, ampqs and authentication in the MySQL and MongoDB drivers), ensured by the Cloud provider.

In the proposed telehealth solution, several user roles are implemented, such as administrator, home sponsor, healthcare professional and the patient himself. The administrator has access to all equipment, reviews the equipment state and location, but cannot access data obtained from sensors to safeguard the patient privacy. The user responsible for a VITABOX has access to all data of related devices and can grant access to other users. The healthcare professional has access only to the data related to the patients that were associated to him. The VITABOX is seen by the Cloud as a client.

User role validation is obtained by the JSON Web Token (JWT) authentication method, whether it is a VITABOX or a web application user. JWT provides the maximum potential of a stateless service that guarantees user authenticity, in which the server does not save clients' sessions. This token is also used to validate WebSocket connections.

As the storage services are hosted in the Cloud, we must assume "Honest but Curious" behaviour of the respective provider [22]. Since the encryption process implies an extra computational cost, it is necessary to understand what data must really be encrypted, to avoid unnecessary CPU expenses. Thereby, only users' attributes as names, addresses, emails, contacts and photos filenames were encrypted using a symmetric key based mechanism, leaving unencrypted the data collected by the sensors. In fact, the data collected by the sensors can also be considered as personal data, but although unencrypted, these data cannot be associated with its owner. So, eventual attackers only know that the data exist, but they cannot know their ownership. This approach results in a significant resource saving, without compromising the user's privacy.

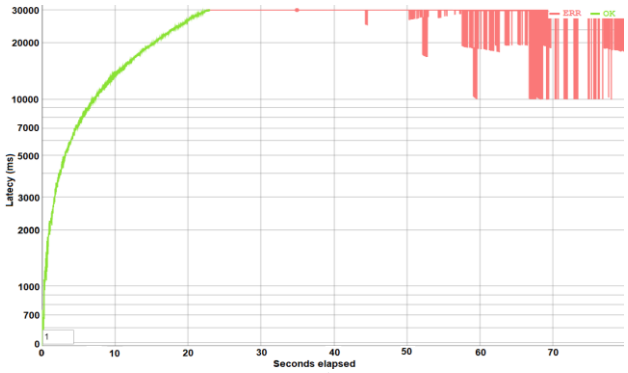

Fig. 4. Load test without separation of services.

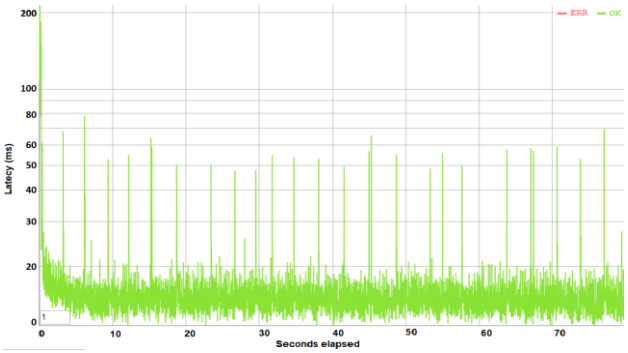

Fig. 5. Load test with sensors data brokering approach.

The databases are accessed by web services through an Object-Relational Model (ORM) and an Object-Document Model (ODM), facilitating the organization of the business logic in the development process and avoiding possible sources of attacks such as the case of SQL injection.

V. RESULTS

During the system development, several approaches were considered to make the services available. Initially the Worker and API functionalities were integrated into the same service, but during load tests, a significant delay was noticeable in the endpoint of sensors data collection. The test tool used was Vegeta [23], with which we were able to realize that the response time was greater in processing the data in a single service, than sending them to a message broker and analyze them in a separate service, not blocking the response. Sending 50 sensor values per request with a frequency of 50 requests per second resulted in a service outage, leading to a latency of more than 30 seconds per request (Fig. 4), which harmed not only the reception of data but also the access to other endpoints. After testing the broker implementation, with the same requests per second and the same amount of data per request, the API node showed a significant improvement, rarely exceeding the response time of 60 ms

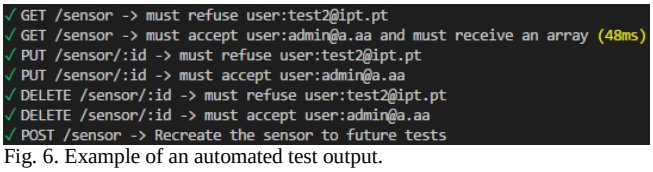

Fig. 6. Example of an automated test output.

(Fig. 5). In this approach the API node just validates the source of the data (VITABOX) and forwards them to the broker.

The results of the tests were obtained from testing just one API node and a Worker node, showing that the microservices approach give us evident improvements. As previously stated, this performance can even be improved with the launching of new service instances (nodes) as needed. In addition, automated functional tests were also implemented, through the JavaScript testing framework Mocha [24], to check if all the endpoints were performing the desired tasks, validating the outputs for irregular situations and estimating the execution time of each endpoint (Fig. 6), allowing the programmer to reshape the code to obtain better performance.

VI. Conclusion and Future Work

A solution was created to remotely monitor biometric and environmental data of patients at their homes. These data are processed and analyzed in the Cloud, to issue alerts and notifications, as well as to provide historical data to family members and duly authorized health professionals. As showed, the adoption of microservices in the Cloud, following the concept of Fog Computing in an IoT solution, provides a better performance and a sustained growth of the system. In addition, this helps avoiding drastic changes in the project architecture in the future, when new functionalities are added, focusing only in the development of the service that is intended to be changed/introduced without penalizing the operation of the remaining.

As this telehealth solution is still an ongoing research project, there is still work to be done. The main change that may be expected is the replacement of the module of sensors' data analysis in the Worker service, by an artificial intelligence (AI) module, since currently the Worker is just based on a set of rules, with adjustable thresholds, that validate the collected data. Regarding the AI module, our goal is to have a classifier based on historical biometric and environmental data that may predict abnormal events in newly collected data. We also intend to add a microservice to support a real-time communication system for remote configuration of the VITABOXes by a technical team.

Acknowledgment

This work has been financially supported by the IC&DT project VITASENIOR-MT CENTRO-01-0145-FEDER-023659 with FEDER funding through programs CENTRO2020 and FCT.

References

[1] A. Chłoń-Domińczak, I. E. Kotowska, J. Kurkiewicz and A. Abramowska-Kmon, "Population ageing in Europe. Facts, implications and policies," European Commission, Directorate-General for Research and Innovation, 2014.

[2] E. Wilkins et al., "European Cardiovascular Disease Statistics 2017," European Heart Network, Brussels, 2017.

[3] A. Botta, W. d. Donato, V. Persico and A. Pescapé, "Integration of Cloud computing and Internet of Things: A survey," *Future Generation Computer Systems*, pp. 684-700, 2016.

[4] G. Pires et al., "VITASENIOR-MT: a telehealth solution for the elderly focused on the interaction with TV," in *20th IEEE Int. Conf. on e-Health Networking, Application & Services*, Ostrava, 2018.

[5] A. M.-H. Kuo, "Opportunities and Challenges of Cloud Computing to Improve Health Care Services," *J. of Med.Internet Res.*, 2011.

[6] S. Babu et al., "Smart telemetry kit for proactive health monitoring in rural India: The journey so far and the road ahead," in *20th IEEE Int. Conf. on e-Health Networking, Applications & Services*, Ostrava, 2018.

[7] I. Villanueva-Miranda, H. Nazeran and R. Martinek, "A Semantic Interoperability Approach to Heterogeneous Internet of Medical Things," in *IEEE 20th Int. Conf. on e-Health Networking, Applications and Services*, Ostrava, 2018.

[8] IBM Corp., "MQTT connectivity for devices," 2018. [Online]. Available: https://console.bluemix.net/docs/services/IoT/devices/mqtt.html. [Accessed September 2018].

[9] Amazon Web Services, Inc., "How AWS IoT Works," 2018. [Online]. Available: https://docs.aws.amazon.com/iot/latest/developerguide/aws-iot-how-it-works.html. [Accessed September 2018].

[10] F. Bonomi, R. Milito, J. Zhu and S. Addepalli, "Fog Computing and Its Role in the Internet of Things," *MCC '12 Proc. of the first ed. of the MCC workshop on Mobile cloud computing*, pp. 13-16, 2012.

[11] IBM Corp., "IaaS, PaaS and SaaS – IBM Cloud service models," 2018. [Online]. Available: https://www.ibm.com/cloud/learn/iaas-paas-saas. [Accessed September 2018].

[12] Pivotal Software, Inc., "RabbitMQ is the most widely deployed open source message broker.," 2007. [Online]. Available: https://www.rabbitmq.com. [Accessed September].

[13] Node.js Foundation, "About Node.js," [Online]. Available: https://nodejs.org/en/about/. [Accessed August 2018].

[14] T. Laurens, "How the V8 engine works?," 2013. [Online]. Available: http://thibaultlaurens.github.io/javascript/2013/04/29/how-the-v8-engine-works/.

[15] A. K. Barczynski, "Assessing the Impact of Using Design Patterns of Enterprise Application Architecture" MSc thesis, *Univ. of Oslo, 2014*

[16] MongoDB, Inc, "Introduction to MongoDB," 2008. [Online]. Available: https://docs.mongodb.com/manual/introduction/. [Accessed August 2018].

[17] Oracle Corporation, "What is MySQL?," 2018. [Online]. Available: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html. [Accessed August 2018].

[18] P. Rottmann, "APIDOC Inline Documentation for RESTful web APIs," 2015. [Online]. Available: http://apidocjs.com/. [Accessed September 2018].

[19] Vue.js Team, "What is Vue.js?," 2018. [Online]. Available: https://vuejs.org/v2/guide/. [Accessed September 2018].

[20] S. Dutton, "WebRTC in the real world: STUN, TURN and signaling,". [Online]. Available: https://www.html5rocks.com/en/tutorials/webrtc/infrastructure/. [Accessed November 2018].

[21] European Union, "General Data Protection Regulation," [Online]. Available: https://gdpr-info.eu/. [Accessed Sep 2018].

[22] S. Yu, C. Wang, K. Ren and W. Lou, "Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing," in *2010 Proc. IEEE INFOCOM*, 2010.

[23] T. Senart, "Vegeta," 2018. [Online]. Available: https://github.com/tsenart/vegeta. [Accessed October 2018].

[24] Mocha Backers, "Mocha simple, flexible, fun," 2018. [Online]. Available: https://mochajs.org/. [Accessed October 2018].