IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 1, 2020, accepted June 24, 2020, date of publication June 29, 2020, date of current version July 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3005638

Telemonitoring System for Infectious Disease Prediction in Elderly People Based on a Novel Microservice Architecture

HURIVIADES CALDERÓN-GÓMEZ[1,2], LUIS MENDOZA-PITTÍ[1,2], MIGUEL VARGAS-LOMBARDO[2], (Member, IEEE), JOSÉ MANUEL GÓMEZ-PULIDO[1,3], JOSÉ LUIS CASTILLO-SEQUERA[1,3], JOSÉ SANZ-MORENO[4], AND GLORIA SENCIÓN[5]

[1]Department of Computer Science, University of Alcalá, 28805 Alcalá de Henares, Spain
[2]E-Health and Supercomputing Research Group (GISES), Technological University of Panama, Panama City 0819-07289, Panama
[3]Ramón y Cajal Institute for Health Research (IRYCIS), 28034 Madrid, Spain
[4]Foundation for Biomedical Research, Hospital Universitario Príncipe de Asturias, 28805 Alcalá de Henares, Spain
[5]School of Medicine, Autonomous University of Santo Domingo, Santo Domingo 10105, Dominican Republic

Corresponding author: Miguel Vargas-Lombardo (miguel.vargas@utp.ac.pa)

This work was supported in part by the Ministry of Science, Innovation and Universities of Spain, in part by the Foundation for Biomedical Research of the Prince of Asturias University Hospital (FUHUPA), in part by the National Secretariat of Science, and the Technology and Innovation of Panama (SENACYT-PANAMA) through the National Research System (SNI-PANAMA) [Design and implementation of a low-cost intelligent system for pre-diagnosis and telecare of infectious diseases in older people (SPIDEP)] under Grant AC16/00061, and in part by the 2nd Joint Call for Research and Innovation (ERANet-LAC) of the Seventh Framework Program for Research and Technological Development (FP7) of the European Union under Contract ELAC2015/T09-0819 SPIDEP.

ABSTRACT This article describes the design, development and implementation of a set of microservices based on an architecture that enables detection and assisted clinical diagnosis within the field of infectious diseases of elderly patients, via a telemonitoring system. The proposed system is designed to continuously update a medical database fed with vital signs from biosensor kits applied by nurses to elderly people on a daily basis. The database is hosted in the cloud and is managed by a flexible microservices software architecture. The computational paradigms of the edge and the cloud were used in the implementation of a hybrid cloud architecture in order to support versatile high-performance applications under the microservices pattern for the pre-diagnosis of infectious diseases in elderly patients. The results of an analysis of the usability of the equipment, the performance of the architecture and the service concept show that the proposed e-health system is feasible and innovative. The system components are also selected to give a cost-effective implementation for people living in disadvantaged areas. The proposed e-health system is also suitable for distributed computing, big data and NoSQL structures, thus allowing the immediate application of machine learning and AI algorithms to discover knowledge patterns from the overall population.

INDEX TERMS Artificial intelligence, e-health, elderly people, infectious diseases, microservice architecture, microservices, telemonitoring.

I. INTRODUCTION

It is becoming increasingly difficult to ignore the growth of an aging population in Europe, since a 74% rise in the population aged over 65 is expected by 2060 [1], [2]. This group of people have high rates of comorbidity [3], causing a very high consumption of resources in terms of both primary and specialised care [4].

A very important problem affecting this group of people is the frequent use of emergency services or prolonged hospital stays, caused mostly by infections [5] and especially respiratory and urinary infections [6]–[8]. This is due to the characteristics of the patients; however, these patients tend to go to the emergency room at relatively advanced stages of disease [9]. In many cases, these patients have non-specific neurological symptoms, general symptoms or other problems that are apparently not related to the infection [7].

The associate editor coordinating the review of this manuscript and approving it for publication was Sunil Karamchandani.

In view of this, the diagnosis of infectious diseases constitutes a problem in that limited medical care is available (e.g. areas with difficult access or rural areas), since these patients require special care due to the complexity of clinical management [7], [10]. This is reflected in an increase in the mortality rate of patients due to delays in treatment [8], [11]. A simple solution would be to increase the human resources available to care for the elderly; however, this solution is unsustainable in the current economic environment of the public sector.

For this reason, it is important to solve this problem by employing other sustainable initiatives such as telemonitoring, since this allows for the adequate management of at-risk populations. In the case of at-risk long-term patients, both medical and nursing care are generally available, but infectious diseases are often diagnosed at later stages, when hospitalisation is required. If the vital signs of these at-risk patients are regularly monitored, it is possible to carry out preventive treatments or interventions in order to minimise health problems and decrease emergency assistance [5]. For patients living in environments without easy access to health resources, telemonitoring is a low-cost, effective way of offering preventive treatments for these diseases [12].

A project called ''Design and implementation of a low-cost intelligent system for the pre-diagnosis and telecare of infectious diseases in elderly people (SPIDEP)'' [13] was carried out by a consortium of Latin American and European R&D entities, the aim of which was to build an intelligent system based on information technologies (ICT) to support the early diagnosis of infectious diseases [14] by integrating a machine learning-based inference system to improve decision support in the prevention, treatment and management of infectious diseases [6], [15].

Special emphasis was placed on the design, development and implementation of the services that make up the SPIDEP platform using the architectural pattern of microservices [5], [14].

This article is part of a broad study in the area of infectious diseases developed jointly with the clinical teams of the Hospital Universitario Principe de Asturias Infectious Diseases Unit (Alcalá de Henares, Spain), Chronic Diseases and Cancer Area of the Ramón y Cajal Institute for Health Research (Madrid, Spain) and School of Medicine Autonomous University of Santo Domingo (Dominican Republic). This study was focused on three target groups (acute respiratory infections, urinary tract infections and skin and soft tissue infections). The article presented is also based on previous works in the computation sciences area realized by the partners of the SPIDEP project [4]–[6], [9], [14].

It should be noted that this study aims to contribute to increasing knowledge in the pre-diagnosis of infectious diseases and offer an alternative vision of telemonitoring of elderly people, with the support of ICTs, by the medical team [5], [6], [9].

This present work aims to incorporate and improve the results obtained from the previous studies that have achieve several significant advances for the medical and health care team, which are: the process of taking measurements is very fast, managing to collect and send the measurement data in less than 2 minutes. These features are essential for the wellness of the resident, who perceives the personal care without discomfort [9] and also the quality of health care grows as well as the satisfaction of residents, relatives and nursing personnel [6]. Another important advantage of the speed of the measurement process is that it increases the productivity of the nursing staff, increasing the residents/elderly assistant ratio [5]. In addition, this benefits the patient directly, since pre-diagnosis through a Clinical Decision Support System (CDSS), supported by telemonitoring, allows the detection of the infection in an initial state and can anticipate the timely administration of corresponding medical treatment, thus avoiding the aggressive spread of the infection afflicting the patient [6], [14].

Nowadays, a group of the authors of this research is adapting the software architecture built through SPIDEP to another infectious disease such as (COVID-19) in the Republic of Panama (funded from the National Secretariat of Science, Technology and Innovation of Panama). The purpose of the project is to determine and establish infection sources (clusters) and store the data in a repository that will store the spatial-temporal information of the infection network, which will make it possible to study the behavior of the disease (COVID-19) during and after the epidemic, relying on the use of the proposed architecture linked to AI algorithms for multivariate statistical analysis of the information collected, whose purpose would be to generate forecast curves with granularity at the regional level to support contingency plans by decision-makers.

This project is being developed with funds from the National Secretariat of Science, Technology and Innovation of Panama (SENACYT-PANAMA).

The present article starts with a brief description of related work, including the motivation for the SPIDEP project and its importance within the field of telemonitoring (e-health). We describe each component of the system in the SPIDEP architecture. Section III presents the results of performance tests (peak tests) at the network and hardware level. The pros and cons of the results are evaluated in Section IV. Finally, Section V presents the conclusions of this study, discusses its potential, and suggests future research activities.

II. RELATED WORK

Microservices represent a relatively new approach to software architectural patterns [16]–[18], as they allow for the development of applications as sets of small services that run independently; in other words, it is not necessary to use the same languages or development platforms [19], [20]. Furthermore, they are interoperable with various communication protocols, since they communicate using lightweight mechanisms such as HTTP [21].

Some research studies have therefore suggested combining this architectural pattern (microservices) with various

computational paradigms (cloud, fog or edge) [22], [23] in various health scenarios, with the aim of significantly improving the accuracy of medical diagnosis by establishing predictions based on expert systems or other types of artificial intelligence.

Mendes *et al.* [1] made use of microservices for the remote monitoring of biometrics (blood pressure and electrocardiograms) and environmental data in a domestic environment specifically designed for the elderly population. They chose this architectural pattern in conjunction with the fog paradigm, with the aim of providing a secure and flexible system in which each service scales independently based on the demand for data processing and analysis.

Likewise, Grgurić *et al.* [24] presented a pilot system called SmartHabits, based on existing IoT architecture and microservices practices; this study focused on the provision of an intelligent service that reassures family members that their loved ones (elderly people) are doing well. This system consists of three main modules: a home-sensing platform (located within the home of an older person living alone), the SmartHabits expert system (located in the cloud, and designed to detect anomalous situations within the home and to deliver notifications), and the underlying communication structure. They focused on the aspect of flexibility in order to meet the needs of users (interface), based on the use and management of smart devices at home, such as smartphones, tablets or smart TVs.

In a study by Roca *et al.* [25], a chatbot architecture was proposed for chronic patient support based on three pillars: scalability through microservices; standard data exchange models through HL7 fast healthcare interoperability resources (FHIR); and modelling of standard conversations using AIM. These three pillars relied on a microservices-based logic that aimed to process user information and perform automated tasks to provide scalability in a healthcare chatbot ecosystem.

Semenov *et al.* [26] suggested a FHIR-based microservices platform that integrated hospital information systems and clinical decision support systems into a unified information space using microservices. A set of separate services were developed for the microservices platform, i.e., asynchronous nodes distributed in groups, whose purpose was to ensure the compatibility and interoperability of its services based on 50,000 transactions per day with more than 400 decision support models.

Alvarez *et al.* [27] applied a microservices-based architecture for the implementation of a failure detection and diagnosis scheme for teleoperated knee rehabilitation devices via the internet, in order to facilitate a medical development protocol for the recovery of the patient's mobility in the case of geographical vulnerability. They demonstrated that the proposed architecture increased the efficiency of development, as each component was independently designed, implemented, and validated for heterogeneous systems.

Meanwhile, Andrikos *et al.* [28] developed a system that allowed for real-time advanced teleconsultation services on medical imaging (radiologists). It consisted of three modules: user access control (UAC), user management (UM) and content management (CM), based on the microservices pattern. The focused on persistence of the data, since each service managed its own database, using either different instances of the same database technology or completely different database systems.

Another approach developed by Khoonsari *et al.* [29] described a workflow for the analysis of data from metabolomics, focusing on the performance aspect. Microservices were included in the development of the necessary components in order to allow for the analysis of encapsulated data while providing reproducible data analysis solutions that were easy-to-run and easy to integrate on desktops and on public and private clouds (Docker containers). Their workflow was validated using various types of data; for instance, two mass spectrometry, one nuclear magnetic resonance spectroscopy and one fluxomics study were used to demonstrate that the method applied scaled optimally when more computing resources became available.

The present work provides a new vision of what has already been carried out in the aforementioned studies, since our contribution is oriented towards the context of medical telemonitoring focused on microservices (SPIDEP) within the field of infectious diseases of elderly patients, with the aim of allowing for the design, development and implementation of services based on a layered architecture in microservices [5], [14]. The objective of this proposal is to offer regular remote monitoring of vital signs by medical personnel [9]. Preventive care can minimise the severity of infectious processes, consequently reducing the resources necessary to adequately control the problem [14].

In view of the aforementioned considerations, it is very important to define the functionalities of the proposed software with a focus on addressing the following needs. Firstly, each component must have the ability to replicate to balance the load as required. Secondly, microservices require horizontal scalability, which allows for a faster and more precise reaction to peaks in demand. Finally, it should have the ability to cope with a large volume of medical data generated from the integration, transfer and storage of biometric sensors in SPIDEP.

For the reasons indicated above, versatility and high performance were identified as fundamental characteristics of the SPIDEP project. These characteristics allow the needs of the environment of health organisations to be satisfied via a focus on agile scaling of microservices that are heavily used and replicating them across multiple containers without underutilising computing resources [19], [30].

III. OVERVIEW OF THE SPIDEP ARCHITECTURE

A. DESIGN OF THE ARCHITECTURE

In this work, we used a fusion of computational paradigms (edge and cloud) [31], [32], based on a hybrid cloud architecture, with the aim of supporting the implementation of

versatile high-performance applications using the microservices pattern. The main arguments for this approach are as follows:

- Low-cost implementation.
- Several options for developing different levels of software quality.
- Scalable and adaptable configuration of resources on demand, since each component can be individually duplicated.
- Compatible with several smart devices and communication protocols.
- Users can run their applications without requiring control by the host.

The architecture can be divided into five functional layers:

- Things layer: This interacts with the processing layer and the network layer, to connect the hardware and validate the signals.
- Network layer: This sets up connections with each APP for the asynchronous uploading of medical data.
- Processing layer: Microservices in this layer use the representational state transfer (REST) protocol to communicate with each other. Each type of device has a different form of access to call the GET, POST, DELETE and PUT methods. In this approach, smartphones and tablets will enter through an API gateway, and computers will enter through user interfaces (UIs); these include the admin UI for the administrator and the nurse/doctor UI for the medical work unit. Each access will be associated with several microservices.
- Microservices layer: This contains microservices for the end users, such as medical personnel and nursing home administrative staff, with their respective profiles. Users can look up historical records, add new biomedical data, manage user profiles and roles, and access pre-diagnosis services. Although each database intercommunicates via JSON request/response HTTP, each microservice has a unique database allowing for independent operation.
- Infrastructure layer: This supports communications, scalability, availability and data integrity for the upper layers. It provides network, server and storage resources for the external clients that connect with each microservice [33], with the aim of achieving greater tolerance to errors in the cloud environment, for example by using MapReduce to manage data in distributed processing [34] and Apache Hadoop to store the data corresponding to each group of microservices within NoSQL database storage structures (Cassandra) [35]. The infrastructure layer also processes complex computation tasks from the upper layers, such as those relating to authentication or interoperability, by alleviating overloaded public or private cloud resources for other primary functions [30], [36], running on Docker containers [37].

In addition, all the interactions between the services within the architecture are processed by the REST API gateway, which acts as a proxy for the microservices (single-entry point into the platform) [38]. This allows for the management of other functional capabilities, such as caching, token management, and microservices monitoring [38], [39].

Likewise, the interactions of each microservice are independent of the physical hosting, and these are either treated individually or grouped into servers or containers [16]. It is therefore important to consider the interaction model that will be implemented in the application, for example for remote procedure calls (RPCs) and event-based interactions [26], the purpose of which is to guarantee that the architecture is robust against failures due to continuous stress at different layers [21], [40].

B. NEW ASPECTS OF THE ARCHITECTURE

The microservices architecture is [17], [41] is a variant of the service-oriented architecture (SOA), both of which provide flexible and scalable properties for execution in the cloud. Each microservice takes on a specific role depending on the requirements of the database [19], [42], [43]. Research in different areas such as system quality, smart city clouds, migration or mobile-oriented applications [41], [44] make use of this approach. However, microservices have certain challenges; for example, their external configuration, microservice discovery, load balancing, central login/logout and metrics or auto-scaling require attention.

The novelty of our proposal lies in the use of microservices in clinical forecast scenarios (infectious diseases), as we propose a continuous construction of an online database to obtain predictive models for the detection of infectious diseases in elderly patients, inspired by the SPIDEP project [13]. A recommender system [6] is also used to support remote assistance in interpreting changes in the vital signs of institutionalised people and in triggering early alerts in the case of possible infection.

Before considering the design of the different microservices for the development of different medical platforms (such as the SPIDEP platform) based on this architecture, it is important to take into account the hierarchy and consolidation of the clinical information of the patients. This is not only necessary for generating standard reports, but can also influence decision making using parameterised, consistent and verified indicators of the collected data in conjunction with other medical information systems [25], [45], [46] that aim to correctly filter the data to minimise false alarms by the recommender systems.

In our case studies (SPIDEP) [9], the standardisation of the data from different nursing homes is a fundamental process in the development of microservice-oriented applications [14]. For this reason, unique records (unique transaction identifiers) represented in EMR-JSON format were used to allow for the use of different types of databases, depending on the particular storage requirements, and in turn provide interoperability to the system (microservices layer).

It was also necessary to customise the fifth layer of the architecture (infrastructure layer) to support a high level of traffic of clinical data on a daily basis. As a consequence, each microservice has a load balancer (NGINX) to reduce the load

for remote calls and in turn to answer users' requests from the application [16]. We therefore use a container orchestrator (Kubernetes) [38] with Docker containers [37].

This customisation means that the architecture is compatible with various cloud computing service providers (Azure, AWS, Google Cloud), in order to run thousands of scalable containers regardless of the infrastructure implemented (private, hybrid or public cloud). In this way, a massive and balanced cluster can be implemented with Kubernetes [38], allowing users to consistently meet the demand for e-health services. The advantages of this management system are numerous; for example, each component can be replicated to achieve load balancing as required [47]; when a small functionality scale is applied, the use of microservices allows the reaction to the peak demand to be take place in real time [38]; the use of containers substantially improves autoscaling, both in terms of the response time of applications and the management of IT resources, which is more rapid and efficient than in virtualised environments (VM) [38], [48]; the use of component modularity makes the system robust to failure [40]; and finally, the decoupling provided by the microservices facilitates its maintainability over time [49].

C. ARCHITECTURE IMPLEMENTATION

1) PLATFORM

As part of the concept of a low-cost platform, this work is based on the proposal of a microservices architecture; this is capable of supporting general service communication with the central cloud environment [9], and in conjunction with a machine learning subsystem can improve decision support for the pre-diagnosis of infectious diseases [6]. Structurally, the SPIDEP platform was developed using microservices architecture (with nine separate services), in which these services work as asynchronous nodes distributed as groups [26], [50], [51] which communicate via the REST communication protocol [52]. It should be noted that the breakdown of the components into small independent services was based on our preliminary proposal for a software architecture for SPIDEP (with five layers), referred to here as version Beta v1 [14].

Three versions of this project were developed: version Alpha (partial implementation of three microservices, with instances using MariaDB) [5], version Beta v1 (implementation of five microservices with instances using a MariaDB Galera cluster) [14], and version Beta v2 (current implementation of the new microservices architecture, with hybrid instances in MariaDB Galera Cluster and NoSQL), with the aims of achieving the characteristics necessary for the implementation of versatile high-performance applications and identifying the key services for the optimal operation of version Beta v1. The existing components were therefore decomposed and restructured along with their data [16], [18], [52].

The flow of the SPIDEP platform in relation to microservices is briefly explained below:

- The scheme in SPIDEP is made up of two end points depending on the device (for mobiles, this is the API gateway, and for PCs the web UI).
- The call to the different microservices corresponding to the credentials is established (by a nurse, doctor, IT developer or administrator).
- Based on the considerations described above, microservices are classified based on the common characteristics of their operation [52], [53] into three groups [14]: the first performs the control and management of user accounts, roles, permissions [54] and manages the measurements received from the biosensors [55], [56]; the second group administers the different patients, affiliated institutions and manages the medical data collected by the first group 14]; and the third is responsible for infectious disease registries. It is also responsible for receiving notifications from the clinical decision support system (CDSS) for the prevention, treatment and management of the three groups of infectious diseases, using the model detailed by Baldominos *et al.* [6]. The proposed platform aims to provide new services and functionalities for the changing needs of users, from the standpoint of management, technology, security and legality [57]. The implementation of the developed architecture provides an ecosystem that is scalable according to demand; however, in order to achieve optimal scalability, it is vital to verify the necessary components (language, database, libraries etc.) for each microservice and the data that will be exchanged between them [29]. Failure to do so would increase network traffic between microservices and HTTP resource APIs (causing saturation), and the overall performance of the application would consequently be degraded until it collapsed [19], as shown in Fig. 1.

For the reasons mentioned above, the architecture implemented in the SPIDEP project was chosen (decompose by verb or use case) [58], because it allows for the creation of flexible and scalable software solutions that run in the cloud [41], [59]. However, the strengths and weaknesses of microservices must also be considered.

The main strengths of the microservices are as follows:

- Microservices can be implemented in a given language and can use the libraries that are best suited to provide the functionality needed [25], [29].
- The structure of the platform allows for different types of databases depending on the requirements 25].
- This allows for the agile scaling of individual microservices, which are widely used by replication based on several containers, without needing to replicate underutilised services [19].
- The automation of the infrastructure allows for a reduction in the manual effort involved in building, and for deploying and operating microservices, thus enabling continuous delivery [21].

However, when implementing microservices, the following weak points or difficulties must be considered:

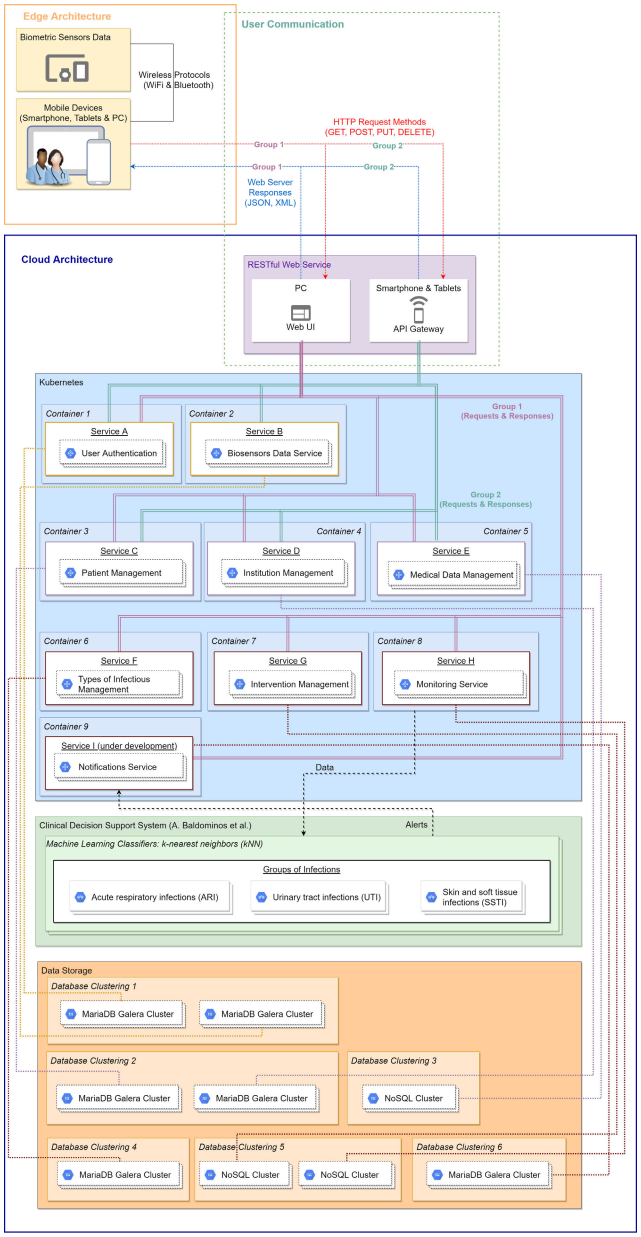


FIGURE 1. General layout of the microservices applied in the SPIDEP platform together with the notification service provided by the CDSS [6].

- This approach requires a clear overview of the data structure and the business processes used in the organisation [40].
- There are several major constraints that directly affect the performance of microservices, including the network, the organisation of complex services, the consistency of data and transaction management at the database level, and the methods of deployment (bare metal server, virtual machine or container). Consequently, cloud infrastructures play a fundamental role in the operation of microservices and their complexity [16], [19], [21].

It should be noted that the user experience in terms of usability of the operation with the different user services was analysed and validated by both the health personnel and the researchers from the international consortium [9], [14]. In addition, the execution and performance of the platform developed in SPIDEP project were shown to meet established standards through various performance tests (spike tests) for each service [42], [59].

2) WORKFLOW FOR THE AUTOMATION OF MICROSERVICES ON THE SPIDEP PLATFORM

Each task involved in an assisted clinical diagnosis in the field of infectious diseases of elderly patients has its own associated microservice. This design allows for the incorporation of new features or the integration of new components developed by third parties.

It is therefore important to define the workflow of the automation for the integration and continuous deployment of services on the SPIDEP platform. This workflow is used by cross-team (typically small groups of six to eight people [60], [61]) for the microservices development process, with the objective of reducing the time between the acceptance of a change into the system and putting it into production [62].

Based on previous research, a proposal was prepared for the automation workflow and was applied to the SPIDEP platform [38], [39], [58], [62]. Fig. 2 shows how this was achieved.

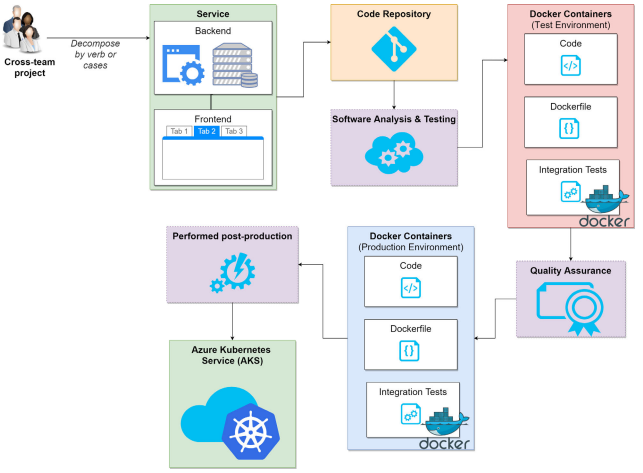


FIGURE 2. Framework for the automation and continuous deployment of services on the SPIDEP platform.

An overview of the proposal is given below.

- Cross-team project: This is responsible for the development, implementation and supervision of the assigned service [63]. It allows for the improvement of health services based on three important characteristics: (i) management, which allows for the creation of personalised interfaces for different groups of users (decision makers, administrators or citizens), according to the requirements and the administrator's credentials; (ii) interoperability, which supports the development of APIs for interconnection with third-party applications

or other medical systems; and (iii) versatility, which gives developers the freedom to choose the information technologies (languages, databases, libraries and others) based on which their services will be developed.

- Service: Each team, if required, has the autonomy to decide which is the best technology for the development of their service in order to meet the requirements [63]. The SPIDEP microservices were developed in two programming languages, PHP and Python. The microservices developed in PHP use the Laravel framework, while the those developed in Python use the Django framework. It should be noted that all microservices developed in SPIDEP have an asynchronous HTTPS server based on TLS for client authentication [25], [54], and the databases applied to each one were SQL (MariaDB) and NoSQL (Cassandra or MongoDB). As a complement to this microservices-based architecture, we use the ''service instance per container'' and ''database per service'' implementation patterns [64].
- Code repository: A Git repository is used to maintain version control of the code from the development stream of each service [29], [65].
- Software analysis and testing: The source code is reviewed to ensure that it meets the required standard and unit tests are performed on each component [62].
- Docker containers (test environment): The tools necessary to run the software are encapsulated inside a test container (Ubuntu server 18.04 LTS, 1 core, 1.5 GB RAM). These containers must meet preliminary test criteria defined by the developers [65], [66], and are then published on the Docker Hub (private repository) for download and used in SPIDEP in the testing environment.
- Quality assurance (QA): Functional tests, load tests and performance tests are carried out to ensure the quality of the microservice [59], [62].
- Docker containers (production environment): If the microservices pass all the tests defined at the QA stage, the container is deployed to a production instance (Ubuntu server 18.04 LTS 1∼2 Core, 3∼4 GB RAM) [38] [49]. Otherwise, it will be necessary to debug the code and repeat the QA tests. The container is then published to a private repository in the Docker Hub.
- Performed post-production: Additional tests (spike tests) are run to ensure that the new version works properly in the production container [16], [59], [62].
- Azure Kubernetes service (AKS) – Kubernetes is used for container orchestration [38], [67], as it allows for initialisation, scaling of container-based jobs, service exposure, and rescheduling of failed jobs and long-running services [29]. We chose Kubernetes rather than other orchestrators (e.g. Swarm) because it is more widely used in production environments (e.g. Azure) and because of its great versatility in container management [37].

IV. VALIDATION OF RESULTS

It was necessary to evaluate the performance of the microservices developed for the SPIDEP project through an analysis of the proposal. We therefore used performance tests (spike tests) and obtained quantitative values that could be used to measure the overall performance of the application and its interaction with end users. [16], [42], [59], [68], [69].

For this purpose, we used two AKS-controlled testing environments consisting of nine Docker instances (Ubuntu Server 18.04 LTS, 1 Core, 3 GB RAM, and no replicas) [14]. Likewise, six SQL instances of a MariaDB Galera cluster and three NoSQL instances were used. The second AKS group used a traditional SQL instance of MariaDB, with the aim of identifying an operational profile needed to host the microservices on SPIDEP in order to maximise the overall performance of the application and its analysis. In addition, we used Apache as an endpoint and NGINX as a load balancer for the implementation. Various user requests are first received by NGINX, and it sends these requests to the corresponding microservices to append to the requested URI.

To carry out the tests, scripts with random variables developed in Apache JMeter (5.2.1) were used to measure the load of virtual users and the performance behaviour of each microservice, together with BlazeMeter servers (US East [Virginia, Google] and EU West [Frankfurt, Google]) in order to generate the workload of the microservices [14], [19], [59].

The default scenario involved accessing the user login (without administrator privileges) and randomly executing a specific instruction for each service (AH), except for Service I (ninth), since this is at an experimental stage of integration with the CDSS [6].

It should be noted that SPIDEP has a medical data set of 6,920 records, and the objective was to generate a workload that drastically increased (10 in 10 active users every five minutes) to emulate a specific number of concurrent user sessions (50 virtual users) according to the specifications of the container (one core, 3 GB RAM).

As a follow-up to this activity, two AKS environments for implementing microservices were used to perform the tests and compare the results. The first environment was based on the implementation of new services with hybrid instances in MariaDB Galera Cluster and NoSQL (SPIDEPMS- T1-HB), while the second was based on the implementation of nine services with a traditional SQL instance of MariaDB (SPIDEPMS-T2-SQL).

We ran the load tests for 20 minutes, with the first test of 50 virtual users on a server located in the USA (NA-Virginia), and the second on a server located in Europe (EU-Frankfurt), since different nursing homes are located on these continents.

To evaluate the performance of the SPIDEP microservices, the results were captured using tabular output reports that were generated after the load tests; these are shown in Tables 1 and 2, which are divided into nine labels (all of the microservices executed [A-H & ALL]).

TABLE 1. Load testing report for SPIDEPMS-T1-HB (time measurements are averages in ms). Each report contains several important values: number of samples, average response time, number of requests processed (hits per second), 90th percentile, 95th percentile, 99th percentile, number and percentage of failed requests, average latency time, and server region.

Label	Samples	Avg. response time	Avg. hits/s	90% line	95% line	99% line	Error count and percentage	Avg. latency	Server location
S-A	2519	1719.470	2.099	2911.000	3119.000	4255.000	0 (0%)	344.504	EU (Germany)
S-B	2512	2112.240	2.099	3615.000	3919.000	4415.000	0 (0%)	358.094	EU (Germany)
S-C	2503	1571.982	2.091	2687.000	2895.000	3199.000	0 (0%)	1548.311	EU (Germany)
S-D	2501	1551.390	2.089	2623.000	2831.000	3119.000	0 (0%)	1529.879	EU (Germany)
S-E	2497	1600.614	2.086	2703.000	2911.000	3199.000	0 (0%)	1577.792	EU (Germany)
S-F	2492	2094.693	2.084	3583.000	3823.000	4191.000	1 (0.04%)	350.259	EU (Germany)
S-G	2483	2105.772	2.076	3615.000	3839.000	4223.000	0 (0%)	340.979	EU (Germany)
S-H	2478	1678.029	2.074	2879.000	3103.000	3487.000	0 (0%)	1654.399	EU (Germany)
ALL	***19985***	***1804.073***	***16.654***	***3119.000***	***3503.000***	***4095.000***	***1 (0.05%)***	***962.057***	***EU (Germany)***
S-A	2404	1854.294	2.007	2895.000	3119.000	4511.000	0 (0%)	479.283	NA (United States)
S-B	2402	2180.154	2.007	3583.000	3807.000	4255.000	0 (0%)	415.681	NA (United States)
S-C	2397	1671.887	2.003	2703.000	2863.000	3215.000	0 (0%)	1586.900	NA (United States)
S-D	2392	1630.239	2.000	2623.000	2815.000	3183.000	0 (0%)	1545.402	NA (United States)
S-E	2386	1669.814	1.997	2703.000	2879.000	3215.000	0 (0%)	1584.658	NA (United States)
S-F	2383	2167.050	1.994	3519.000	3791.000	4127.000	0 (0%)	428.285	NA (United States)
S-G	2371	2186.528	1.986	3519.000	3775.000	4191.000	0 (0%)	443.802	NA (United States)
S-H	2362	1743.218	1.980	2831.000	3039.000	3423.000	0 (0%)	1658.582	NA (United States)
ALL	***19097***	***1887.766***	***15.914***	***3103.000***	***3439.000***	***4063.000***	***0 (0%)***	***1016.982***	***NA (United States)***

TABLE 2. Load testing report for SPIDEPMS-T2-SQL (time measurements are averages in ms). Each report contains several important values: number of samples, average response time, number of requests processed (hits per second), 90th percentile, 95th percentile, 99th percentile, number and percentage of failed requests, average latency time, and server region.

Label	Samples	Avg. response time	Avg. hits/s	90% line	95% line	99% line	Error count and percentage	Avg. latency	Server location
S-A	7676	527.286	6.397	639.000	795.000	2143.000	7636 (99.5%)	464.612	EU (Germany)
S-B	7671	489.840	6.446	615.000	723.000	2175.000	7630 (99.5%)	418.627	EU (Germany)
S-C	7665	673.494	6.512	611.000	719.000	10495.000	7445 (97.1%)	673.138	EU (Germany)
S-D	7657	709.801	6.567	611.000	747.000	10367.000	7403 (96.7%)	709.383	EU (Germany)
S-E	7648	680.486	6.622	615.000	715.000	10559.000	7425 (97.1%)	679.954	EU (Germany)
S-F	7642	489.999	6.680	607.000	711.000	2159.000	7605 (99.5%)	421.735	EU (Germany)
S-G	7635	473.286	6.757	607.000	719.000	2111.000	7605 (99.6%)	413.221	EU (Germany)
S-H	7632	669.365	6.826	611.000	711.000	11071.000	7431 (97.4%)	668.887	EU (Germany)
ALL	***61226***	***589.164***	***51.022***	***615.000***	***727.000***	***10431.000***	***60180 (98.3%)***	***556.149***	***EU (Germany)***
S-A	6131	644.917	5.109	571.000	915.000	4191.000	6082 (99.2%)	559.790	NA (United States)
S-B	6125	616.532	5.147	543.000	715.000	2159.000	6083 (99.3%)	525.174	NA (United States)
S-C	6122	891.623	5.201	547.000	715.000	10623.000	5876 (96.0%)	888.171	NA (United States)
S-D	6111	802.986	5.241	535.000	675.000	10431.000	5915 (96.8%)	800.232	NA (United States)
S-E	6106	861.751	5.282	539.000	703.000	10687.000	5886 (96.4%)	857.184	NA (United States)
S-F	6096	629.875	5.324	547.000	735.000	5215.000	6052 (99.3%)	533.966	NA (United States)
S-G	6092	606.190	5.377	543.000	699.000	2207.000	6055 (99.4%)	524.718	NA (United States)
S-H	6090	846.392	5.438	543.000	683.000	11327.000	5885 (96.6%)	842.092	NA (United States)
ALL	***48873***	***737.500***	***40.728***	***543.000***	***711.000***	***10687.000***	***47834 (97.9%)***	***691.374***	***NA (United States)***

To determine the versatility and robust architecture we have analyzed the global behaviors of the microservices in a real environment of constant and gradual stress, as shown in Figs. 3 and 4. The first scenario (SPIDEPMS-T1-HB) showed that the average response time and the performance statistics (request/hits per second, 90th percentile and 95th percentile), correspond to the established acceptance criteria (5,000 ms); that is, keeping up with the behavior of the identified parameters we can appreciate that the trend increases in a constant and controlled way, according to the amount of active users per second consulting simultaneously the microservices from different regions (United States and Germany) according to the specifications of the container (one core, 3 GB RAM). Meanwhile, the second scenario (SPIDEPMS-T2SQL) showed a chaotic and irregular behavior up to the point of total collapse of SPIDEP Platform.

Several stress simulations were also carried out on the infrastructure hosting SPIDEP, with the aim of exposing the problems that may arise on the platform. Regarding CPU performance, memory, network I/O and connections (Engine Health - BlazeMeter) [70], we analyzed whether the SPIDEP infrastructure itself is capable of supporting the bottlenecks

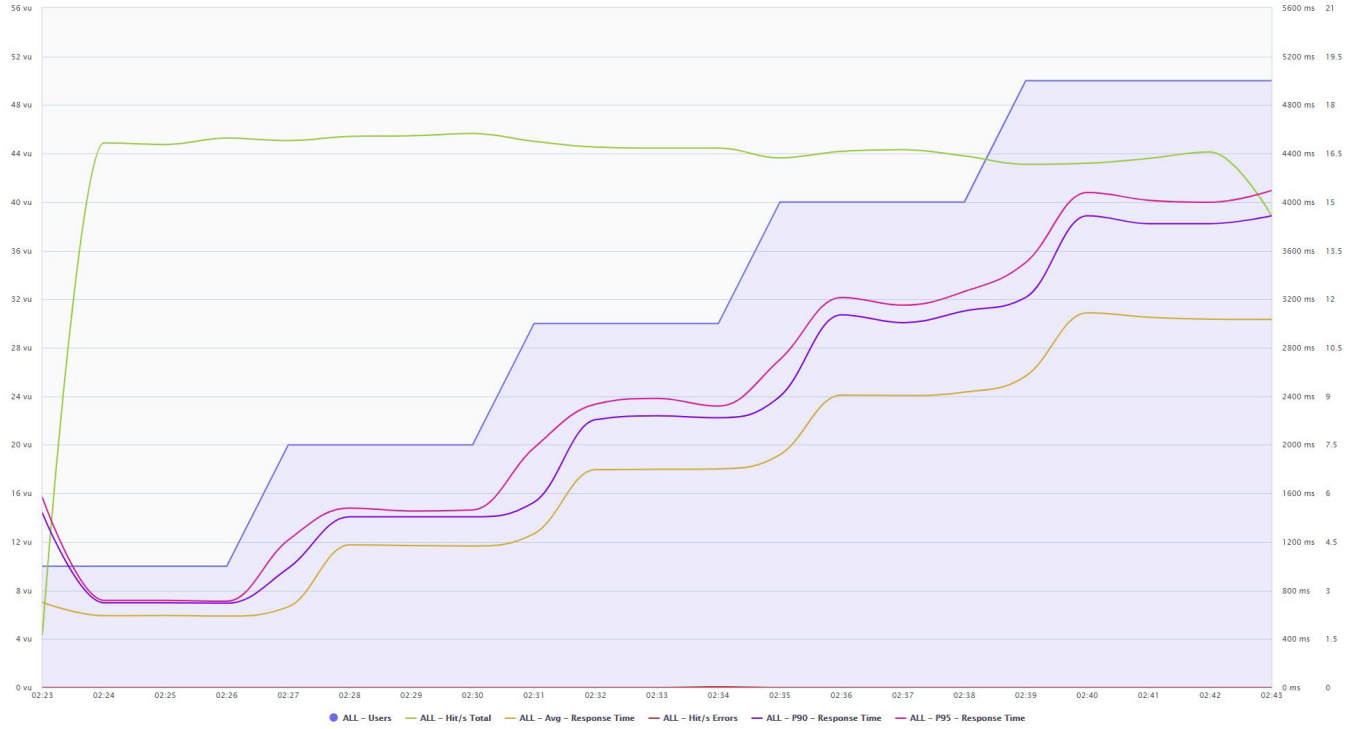


FIGURE 3. Microservice performance patterns for SPIDEPMS-T1-HB (first scenario). The coloured lines represent various parameters: blue: active connections (vu); light green: request/hits per second; red: error/hits per second; light orange: average response time; magenta: 90th percentile response time; purple: 95th percentile response time.

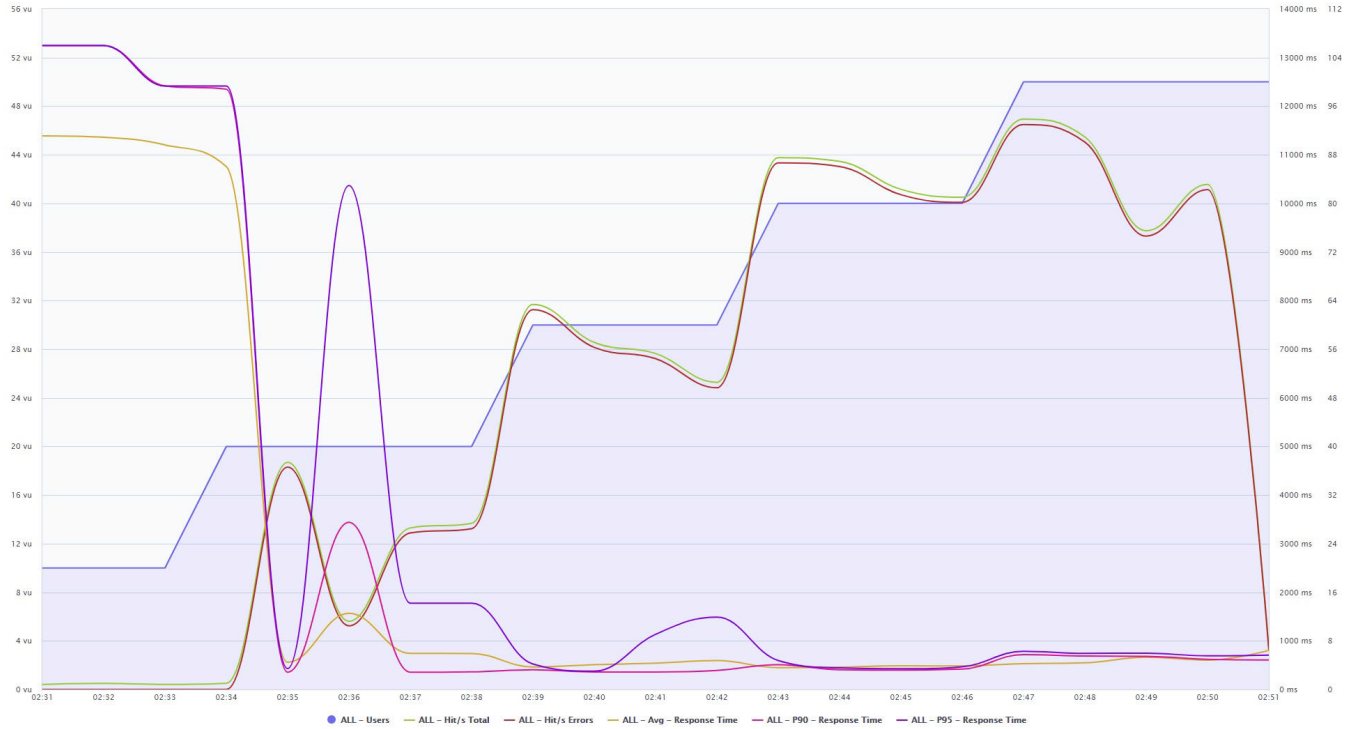


FIGURE 4. Microservice performance patterns for SPIDEPMS-T2-SQL (second scenario). The coloured lines represent various parameters: blue: active connections (vu); light green: request/hits per second; red: error/hits per second; light orange: average response time; magenta: 90th percentile response time; purple: 95th percentile response time.

or errors that occur due to demand; however, the only component that differs in each application is the bandwidth traffic (network I/O), as shown in Figs. 5 and 6.

V. DISCUSSION

To carry out the analysis of the data obtained from the previous section (validation of results), we based ourselves on the

FIGURE 5. Engine Health Report for SPIDEPMS-T1-HB. The coloured lines represent various parameters: purple: bandwidth traffic; blue: memory load generated by users; light blue: CPU load generated by users; black: active connections within the test.

FIGURE 6. Engine Health Report for SPIDEPMS-T2-SQL. The coloured lines represent various parameters: purple: bandwidth traffic; blue: memory load generated by users; light blue: CPU load generated by users; black: active connections within the test.

case studies of the Francisco de Vitoria and Cisneros nursing home in Spain and Carls George and Ntra. Sra del Carmen medical elderly care institution at Dominican Republic, who generously collaborated with their work to obtain the clinical and physiological data of the residents (6,920 records for this study) [6], [9].

This study was carried out with the objective of evaluating the performance and the analysis of each microservice applied to the SPIDEP project within the context of e-health, and for which we analysed, identified and described the main tasks adopted for the design and use of the proposals to perform this work [14].

There are several significant findings from this study. Firstly, it is important to consider acceptance criteria such as the average response time of requests, 90th percentile and their relationship to the percentage of failed requests. The following criteria were established based on various research studies [19], [69], [71]. Criterion A: an acceptable limit for the average response times of the queries and the 90th percentile was 5,000 ms (5 s); criterion B: the acceptable limit for the percentage of failed requests was 5%. Based on these established limits for each test output, any value exceeding these limits was considered to show unacceptable performance.

In view of the above, it can be deduced from Table 1 (SPIDEPMS-T1-HB) that the tests with 50 users were acceptable, since they met the two defined criteria. However, it can be observed from the results in Table 2 (SPIDEPMS-T2-SQL) that the performance is not acceptable: although the results obtained from tests (EU and NA) under criterion A met the established limits, criterion B (percentage of failed requests) was not met, as it exceeded 5%. Since one of the two criteria was not met, the scenario is considered unacceptable.

Regarding the second finding, it is important to note that we observed that microservices created without taking into account the weaknesses of this technology mentioned above, for instance without considering the effects on performance of breaking down an application into multiple services, not contemplate the correct data segmentation (single instance) or not enable a cloud infrastructure for the operation of microservices [19], [72], [73]. Due to the increase in network traffic between microservices and HTTP resource APIs, there appears to be an inherent negative correlation between increased latency and an overall degradation in the performance of the application until collapse is seen.

Additionally, in the third finding, confirmed that the manually created microservices (unsupervised) based on single traditional SQL or software engineering principles do not always yield better performance compared to the monolithic implementation [19], as shown in Fig. 4. In conclusion, to guarantee the versatility and robustness of the architecture, three aspects are important to take into account: (i) scalability, microservices can be scaled individually when running a heavy workload just by replicating them on several containers without and not replicating the others underutilized (maximize the performance with minimal cost) [19], [74]; (ii) communication in a microservice, is important establishing simple communication protocol like http, http-rest ''request/response'' (synchronous protocol) or mqtt ''publish/subscribe'' (asynchronous protocol), depending on needs [67]; (iii) fine-grained microservices, it is fundamental to decompose each service focused on a specific function and of limited influence, according to the established requirements [75]; otherwise, this architecture suffers from a high level of abstraction and coordination among the teams [67], [76].

In terms of the fourth finding, both tests (SPIDEPMS-T1-HB and SPIDEPMS-T2-SQL) met the performance criteria for infrastructure (CPU and RAM) under demand from 50 users, since the CPU values were lower than 80% (15% $\sim$ 35%) and memory levels were less than 70% (10% $\sim$ 20%), as shown in Figs. 5 and 6. However, the network I/O value for SPIDEPMS-T2-SQL showed that there was saturation in the bandwidth traffic between the services and the traditional SQL instance (a single database for all services). Consequently, there was a high possibility of packet loss between services and data (communication). Due to this last factor, a high percentage of failed requests occurred, as shown in Fig. 6.

VI. CONCLUSIONS AND FUTURE WORK

This work has analysed, described and justified all the steps involved in the design, development and implementation of the e-health services that make up the SPIDEP project platform. This platform is based on a five-tier architecture using microservices. The objective of the platform is to create a framework based on the use of new ICT to support the early diagnosis of infectious diseases with the added benefits of increasing the level of service in medical care and reductions in cost.

The proposed e-health system is also suitable for distributed computing and for the use of big data and NoSQL structures that allow for the immediate application of machine learning and AI algorithms to discover hidden patterns in the health data of this population. Above all, however, the key innovation that brings telemonitoring system is the possibility of obtaining the medical information necessary to build analytical and predictive e-health models.

In addition, this proposal paves the way for future research for the design of services, based on the proposed workflow for automation, integration and continuous deployment of services, with the aim of achieving greater performance in terms of organisation and patient care.

The work done here is currently being expanded by exploring the possibility of further optimising the performance of SPIDEP by implementing microservices in other programming language environments (paradigms), or by integrating Kafka or another platform for event-driven management within Kubernetes, in order to allow for comparisons between versions. In addition, integration tests (SPIDEP-CDSS) have been planned with various practical studies (real data from a patient group) of the ninth service (Service I) since this work is currently at the experimental stage.

Another possible area of future research would be to incorporate the perspective of data mining by developing microservices adapted to machine learning (e.g., algorithms for early prediction of COVID-19 [77]), with the aim of identifying patterns in COVID-19 behavior; however, it would

be necessary to provide sufficient time-space series data to begin training the required algorithms [78]. For this reason, the research group is currently implementing a national geolocalized repository containing health and referral data of patients infected during the COVID-19 epidemic in Panama (proposed case study), funded by the National Secretariat of Science, Technology and Innovation of Panama (SENACYT-PANAMA).

In this sense, these data collected by the regional repository will provide various demographic variables that affect the rate of spread, such as: blocking variables, variability in infected populations, socioeconomic information and others [79], [80]. The georeferenced results of these analyses represent novel microservices for evaluating the development of the disease over time, considering demographic data and implementing effective measures to contain the spread of future epidemiological events such as social distancing and health fences, among others [80].

Further researches are needed to obtain sufficient results that can demonstrate the robustness of the proposed architecture in terms of the adaptations of the microservices to identify patterns of COVID-19 behavior or other epidemiological events.

ACKNOWLEDGEMENTS

The authors would like to extend special thanks to all the clinical and healthcare staff of the nursing homes for the elderly, the Francisco de Vitoria and Cisneros centres of the Department of Social Policies and Family of the Community of Madrid in Spain, and Ntra. Sra del Carmen and Carls George in the Dominican Republic, who generously worked with us to obtain the medical data of the residents.

REFERENCES

[1] D. Mendes, D. Jorge, G. Pires, R. Panda, R. António, P. Dias, and L. Oliveira, "VITASENIOR-MT: A distributed and scalable cloud-based telehealth solution," in *Proc. IEEE 5th World Forum Internet Things (WF-IoT)*, Limerick, Ireland, Apr. 2019, pp. 767–772, doi: 10.1109/WF-IoT.2019.8767184.

[2] V. Kontis, J. E. Bennett, C. D. Mathers, G. Li, K. Foreman, and M. Ezzati, "Future life expectancy in 35 industrialised countries: Projections with a Bayesian model ensemble," *Lancet*, vol. 389, no. 10076, pp. 1323–1335, Apr. 2017, doi: 10.1016/S0140-6736(16)32381-9.

[3] M. Solé-Casals, E. Chirveches-Pérez, E. Puigoriol-Juvanteny, N. Nubó-Puntí, C. Chabrera-Sanz, and M. Subirana-Casacuberta, "Perfil y resultados del paciente frágil valorado por la enfermera de Práctica avanzada en un servicio de urgencias," *Enfermería Clínica*, vol. 28, no. 6, pp. 365–374, Nov. 2018, doi: 10.1016/J.ENFCLI.2017.04.003.

[4] A. Baldominos, A. Puello, H. Ogul, T. Asuroglu, and R. Colomo-Palacios, "Predicting infections using computational intelligence—A systematic review," *IEEE Access*, vol. 8, pp. 31083–31102, 2020, doi: 10.1109/ACCESS.2020.2973006.

[5] J. Sanz-Moreno, J. Gómez-Pulido, A. Garcés, H. Calderón-Gómez, M. Vargas-Lombardo, J. L. Castillo-Sequera, M. L. P. Luque, R. Toro, and G. Sención-Martínez, "mHealth system for the early detection of infectious diseases using biomedical signals," in *Advances in Automation and Robotics Research*. Cham, Switzerland: Springer, 2020, pp. 203–213, doi: 10.1007/978-3-030-40309-6_20.

[6] A. Baldominos, H. Ogul, R. Colomo-Palacios, J. Sanz-Moreno, and J. M. Gómez-Pulido, "Infection prediction using physiological and social data in social environments," *Inf. Process. Manage.*, vol. 57, no. 3, May 2020, Art. no. 102213, doi: 10.1016/J.IPM.2020.102213.

[7] F. Raschilas, H. Blain, and C. Jeandel, "La infección en el paciente de edad avanzada," *EMC-Tratado Med.*, vol. 10, no. 2, pp. 1–11, Jan. 2006, doi: 10.1016/S1636-5410(06)70381-6.

[8] E. Á. Artero, A. C. Nuñez, M. G. Bravo, O. C. Calvo, M. B. Garcia, and J. P. Lledias, "Infección urinaria en el anciano," *Revista Clínica Española*, vol. 219, no. 4, pp. 189–193, May 2019, doi: 10.1016/J.RCE.2018.10.009.

[9] H. Calderón-Gómez, A. Garcés-Jiménez, M. Vargas-Lombardo, J. M. Gómez-Pulido, M.-L. Polo-Luque, J. L. Castillo, G. Sención, and J. S. Moreno, "Proposal using the cloud architecture in system for the early detection of infectious diseases in elderly people fed by biosensors records," in *Proc. 7th Int. Eng., Sci. Technol. Conf. (IESTEC)*, Panama City, Panama, 2019, pp. 631–634, doi: 10.1109/IESTEC46403.2019.00118.

[10] M. A. Martín Martínez, R. C. Alférez, E. Escortell Mayor, M. Rico Blázquez, and A. Sarría Santamera, "Factores asociados a reingresos hospitalarios en pacientes de edad avanzada," *Atención Primaria*, vol. 43, no. 3, pp. 117–124, Mar. 2011, doi: 10.1016/J.APRIM.2009.12.007.

[11] S. M. Q. Vera, D. M. Rojas Aguilar, D. A. Chavarro-Carvajal, and I. Riaño Forero, "Mortalidad en pacientes mayores de 65 años ingresados en cuidados intensivos del hospital universitario san ignacio en el 2014," *Acta Colombiana de Cuidado Intensivo*, vol. 19, no. 2, pp. 61–68, Apr. 2019, doi: 10.1016/J.ACCI.2018.11.002.

[12] B. Dinesen, B. Nonnecke, D. Lindeman, E. Toft, K. Kidholm, K. Jethwani, H. M. Young, H. Spindler, C. U. Oestergaard, J. A. Southard, M. Gutierrez, N. Anderson, N. M. Albert, J. J. Han, and T. Nesbitt, "Personalized telehealth in the future: A global research agenda," *J. Med. Internet Res.*, vol. 18, no. 3, p. e53, Mar. 2016, doi: 10.2196/jmir.5257.

[13] EU-CELAC. Fundación para la Investigación Biomédica del Hospital Universitario Príncipe de Asturias, (2017). *Design and Implementation of a Low Cost Smart System for Pre-Diagnosis and Telecare of Infectious Diseases in Elderly People | EU-CELAC*. Accessed: Feb. 11, 2020. [Online]. Available: https://www.eucelac-platform.eu/project/design-and-implementation-low-cost-smart-system-pre-diagnosis-and-telecare-infectious

[14] H. Calderón-Gómez *et al.*, "Development of ehealth applications-based on microservices in a cloud architecture," *RISTI Rev. Iber. Sist. Tecnol. Inf.*, vol. 2019, no. E23, pp. 81–93, Oct. 2019.

[15] D. C. Robinson, S. Mohanty, J. Young, G. Jones, and D. Wesemann, "Novel techniques for mapping infectious diseases using point of care diagnostic sensors," in *Proc. 2nd Int. Symp. Phys. Technol. Sensors (ISPTS)*, Pune, India, 2015, pp. 14–18, doi: 10.1109/ISPTS.2015.7220148.

[16] D. Guaman, L. Yaguachi, C. C. Samanta, J. H. Danilo, and F. Soto, "Performance evaluation in the migration process from a monolithic application to microservices," in *Proc. 13th Iberian Conf. Inf. Syst. Technol. (CISTI)*, Caceres, Jun. 2018, pp. 1–8, doi: 10.23919/CISTI.2018.8399148.

[17] S. Newman, *Building Microservices: Designing Fine-Grained Systems*, 1st ed. Newton, MA, USA: O'Reilly Media, 2015, p. 280.

[18] H. Jita and V. Pieterse, "A framework to apply the Internet of Things for medical care in a home environment," in *Proc. Int. Conf. Cloud Comput. Internet Things*, 2018, pp. 45–54, doi: 10.1145/3291064.3291065.

[19] M. Abdullah, W. Iqbal, and A. Erradi, "Unsupervised learning approach for Web application auto-decomposition into microservices," *J. Syst. Softw.*, vol. 151, pp. 243–257, May 2019, doi: 10.1016/J.JSS.2019.02.031.

[20] D. Namiot and M. Sneps-Sneppe, "On micro-services architecture," *Int. J. Open Inf. Technol.*, vol. 2, no. 9, pp. 24–27, 2014.

[21] P. Di Francesco, P. Lago, and I. Malavolta, "Architecting with microservices: A systematic mapping study," *J. Syst. Softw.*, vol. 150, pp. 77–97, Apr. 2019, doi: 10.1016/J.JSS.2019.01.001.

[22] I. Odun-Ayo, R. Goddy-Worlu, V. Geteloma, and E. Grant, "A systematic mapping study of cloud, fog, and edge/mobile devices management, hierarchy models and business models," *Adv. Sci., Technol. Eng. Syst. J.*, vol. 4, no. 2, pp. 91–101, 2019, doi: 10.25046/aj040212.

[23] A. Al-Qamash, I. Soliman, R. Abulibdeh, and M. Saleh, "Cloud, fog, and edge computing: A software engineering perspective," in *Proc. Int. Conf. Comput. Appl. (ICCA)*, Aug. 2018, pp. 276–284, doi: 10.1109/COMAPP.2018.8460443.

[24] A. Grguriš, M. Mošmondor, and D. Huljenić, "The SmartHabits: An intelligent privacy-aware home care assistance system," *Sensors*, vol. 19, no. 4, p. 907, Feb. 2019, doi: 10.3390/s19040907.

[25] S. Roca, J. Sancho, J. García, and Á. Alesanco, "Microservice chatbot architecture for chronic patient support," *J. Biomed. Informat.*, vol. 102, Feb. 2020, Art. no. 103305, doi: 10.1016/J.JBI.2019.103305.

[26] I. Semenov, R. Osenev, S. Gerasimov, G. Kopanitsa, D. Denisov, and Y. Andreychuk, "Experience in developing an FHIR medical data management platform to provide clinical decision support," *Int. J. Environ. Res. Public Health*, vol. 17, no. 1, p. 73, Dec. 2019, doi: 10.3390/ijerph17010073.

[27] J. M. Alvarez, J. A. Sanabria, and J. I. Garcia, "Microservices-based architecture for fault diagnosis in tele-rehabilitation equipment operated via Internet," in *Proc. IEEE Latin Amer. Test Symp. (LATS)*, Santiago, Chile, Mar. 2019, pp. 1–6, doi: 10.1109/LATW.2019.8704556.

[28] C. Andrikos, G. Rassias, P. Tsanakas, and I. Maglogiannis, "An enhanced device-transparent real-time teleconsultation environment for radiologists," *IEEE J. Biomed. Health Informat.*, vol. 23, no. 1, pp. 374–386, Jan. 2019, doi: 10.1109/JBHI.2018.2824312.

[29] P. E. Khoonsari *et al.*, "Interoperable and scalable data analysis with microservices: Applications in metabolomics," *Bioinformatics*, vol. 35, no. 19, pp. 3752–3760, Mar. 2019, doi: 10.1093/bioinformatics/btz160.

[30] M. Saadeh, A. Sleit, K. E. Sabri, and W. Almobaideen, "Hierarchical architecture and protocol for mobile object authentication in the context of IoT smart cities," *J. Netw. Comput. Appl.*, vol. 121, pp. 1–19, Nov. 2018, doi: 10.1016/J.JNCA.2018.07.009.

[31] L. Catarinucci, D. de Donno, L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, and L. Tarricone, "An IoT-aware architecture for smart healthcare systems," *IEEE Internet Things J.*, vol. 2, no. 6, pp. 515–526, Dec. 2015, doi: 10.1109/jiot.2015.2417684.

[32] A. Javed, J. Robert, K. Heljanko, and K. Främling, "IoTEF: A federated edge-cloud architecture for fault-tolerant IoT applications," *J. Grid Comput.*, vol. 18, no. 1, pp. 57–80, Mar. 2020, doi: 10.1007/s10723-019-09498-8.

[33] A. Sharma, M. Kumar, and S. Agarwal, "A complete survey on software architectural styles and patterns," *Procedia Comput. Sci.*, vol. 70, pp. 16–28, 2015, doi: 10.1016/j.procs.2015.10.019.

[34] D. Glushkova, P. Jovanovic, and A. Abelló, "Mapreduce performance model for Hadoop 2.X," *Inf. Syst.*, vol. 79, pp. 32–43, Jan. 2019, doi: 10.1016/J.IS.2017.11.006.

[35] R. Hernandez, Y. Becerra, J. Torres, and E. Ayguade, "Automatic query driven data modelling in cassandra," *Procedia Comput. Sci.*, vol. 51, pp. 2822–2826, Jan. 2015, doi: 10.1016/J.PROCS.2015.05.441.

[36] O. Bello and S. Zeadally, "Intelligent device-to-device communication in the Internet of Things," *IEEE Syst. J.*, vol. 10, no. 3, pp. 1172–1182, Sep. 2016, doi: 10.1109/JSYST.2014.2298837.

[37] X. Wan, X. Guan, T. Wang, G. Bai, and B.-Y. Choi, "Application deployment using microservice and docker containers: Framework and optimization," *J. Netw. Comput. Appl.*, vol. 119, pp. 97–109, Oct. 2018, doi: 10.1016/j.jnca.2018.07.003.

[38] S. Taherizadeh and M. Grobelnik, "Key influencing factors of the kubernetes auto-scaler for computing-intensive microservice-native cloud-based applications," *Adv. Eng. Softw.*, vol. 140, Feb. 2020, Art. no. 102734, doi: 10.1016/J.ADVENGSOFT.2019.102734.

[39] T. Kiss, P. Kacsuk, J. Kovacs, B. Rakoczi, A. Hajnal, A. Farkas, G. Gesmier, and G. Terstyanszky, "MiCADO—Microservice-based cloud application-level dynamic orchestrator," *Future Gener. Comput. Syst.*, vol. 94, pp. 937–946, May 2019, doi: 10.1016/J.FUTURE.2017.09.050.

[40] B. Götz, D. Schel, D. Bauer, C. Henkel, P. Einberger, and T. Bauernhansl, "Challenges of production microservices," *Procedia CIRP*, vol. 67, pp. 167–172, 2018, doi: 10.1016/J.PROCIR.2017.12.194.

[41] M. Krämer, S. Frese, and A. Kuijper, "Implementing secure applications in smart city clouds using microservices," *Future Gener. Comput. Syst.*, vol. 99, pp. 308–320, Oct. 2019, doi: 10.1016/J.FUTURE.2019.04.042.

[42] M. Gribaudo, M. Iacono, and D. Manini, "Performance evaluation of replication policies in microservice based architectures," *Electron. Notes Theor. Comput. Sci.*, vol. 337, pp. 45–65, May 2018, doi: 10.1016/J.ENTCS.2018.03.033.

[43] H. Suryotrisongko, D. P. Jayanto, and A. Tjahyanto, "Design and development of backend application for public complaint systems using microservice spring boot," *Procedia Comput. Sci.*, vol. 124, pp. 736–743, Jan. 2017, doi: 10.1016/J.PROCS.2017.12.212.

[44] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, L. Safina, "Microservices: Yesterday, today, and tomorrow," in *Present and Ulterior Software Engineering*. Cham, Switzerland: Springer, 2017, pp. 195–216.

[45] P. W. Handayani, A. N. Hidayanto, A. A. Pinem, I. C. Hapsari, P. I. Sandhyaduhita, and I. Budi, "Acceptance model of a hospital information system," *Int. J. Med. Informat.*, vol. 99, pp. 11–28, Mar. 2017, doi: 10.1016/J.IJMEDINF.2016.12.004.

[46] C. Catley and M. Frize, "Design of a health care architecture for medical data interoperability and application integration," in *Proc. 2nd Joint 24th Annu. Conf. Annu. Fall Meeting Biomed. Eng. Soc. Eng. Med. Biol.*, vol. 3, 2002, pp. 1952–1953, doi: 10.1109/IEMBS.2002.1053111.

[47] W. Hasselbring, "Microservices for scalability: Keynote talk abstract," in *Proc. 7th ACM/SPEC Int. Conf. Perform. Eng.*, New York, NY, USA, 2016, pp. 133–134, doi: 10.1145/2851553.2858659.

[48] A. R. Sampaio, J. Rubin, I. Beschastnikh, and N. S. Rosa, "Improving microservice-based applications with runtime placement adaptation," *J. Internet Services Appl.*, vol. 10, no. 1, p. 4, Dec. 2019, doi: 10.1186/s13174-019-0104-0.

[49] C. Chen, "With great abstraction comes great responsibility: Sealing the microservices attack surface," in *Proc. IEEE Cybersecur. Develop. (SecDev)*, Tysons Corner, VA, USA, Sep. 2019, p. 144, doi: 10.1109/SecDev.2019.00027.

[50] Y. Yu, J. Yang, C. Guo, H. Zheng, and J. He, "Joint optimization of service request routing and instance placement in the microservice system," *J. Netw. Comput. Appl.*, vol. 147, Dec. 2019, Art. no. 102441, doi: 10.1016/J.JNCA.2019.102441.

[51] R. Heinrich, A. V. Hoorn, H. Knoche, F. Li, L. E. Lwakatare, C. Pahl, S. Schulte, and J. Wettinger, "Performance engineering for microservices: Research challenges and directions," in *Proc. 8th ACM/SPEC Int. Conf. Perform. Eng. Companion*, New York, NY, USA, 2017, pp. 223–226, doi: 10.1145/3053600.3053653.

[52] C. Fan and S. Ma, "Migrating monolithic mobile application to microservice architecture: An experiment report," in *Proc. IEEE Int. Conf. AI Mobile Services (AIMS)*, Honolulu, HI, USA, Jun. 2017, pp. 109–112, doi: 10.1109/AIMS.2017.23.

[53] S.-P. Ma, C.-Y. Fan, Y. Chuang, I.-H. Liu, and C.-W. Lan, "Graph-based and scenario-driven microservice analysis, retrieval, and testing," *Future Gener. Comput. Syst.*, vol. 100, pp. 724–735, Nov. 2019, doi: 10.1016/J.FUTURE.2019.05.048.

[54] K. Jander, L. Braubach, and A. Pokahr, "Defense-in-depth and role authentication for microservice systems," *Procedia Comput. Sci.*, vol. 130, pp. 456–463, Jan. 2018, doi: 10.1016/J.PROCS.2018.04.047.

[55] P. Raković and B. Lutovac, "A cloud computing architecture with wireless body area network for professional athletes health monitoring in sports organizations—Case study of Montenegro," in *Proc. 4th Medit. Conf. Embedded Comput. (MECO)*, Budva, Montenegro, 2015, pp. 387–390, doi: 10.1109/MECO.2015.7181950.

[56] A. Domínguez and M. Vargas-Lombardo, "El estado del arte: Salud inteligente y el Internet de las cosas," *I+D Tecnológico*, vol. 14, no. 1, pp. 14–17, Jun. 2018, doi: 10.33412/idt.v14.1.1809.

[57] A. Fatima and R. Colomo-Palacios, "Security aspects in healthcare information systems: A systematic mapping," *Procedia Comput. Sci.*, vol. 138, pp. 12–19, Jan. 2018, doi: 10.1016/J.PROCS.2018.10.003.

[58] S. Li, H. Zhang, Z. Jia, Z. Li, C. Zhang, J. Li, Q. Gao, J. Ge, and Z. Shan, "A dataflow-driven approach to identifying microservices from monolithic applications," *J. Syst. Softw.*, vol. 157, Nov. 2019, Art. no. 110380, doi: 10.1016/J.JSS.2019.07.008.

[59] Y. Gan and C. Delimitrou, "The architectural implications of cloud microservices," *IEEE Comput. Archit. Lett.*, vol. 17, no. 2, pp. 155–158, Jul. 2018, doi: 10.1109/LCA.2018.2839189.

[60] Á. Brandón, M. Solé, A. Huélamo, D. Solans, M. S. Pérez, and V. Muntés-Mulero, "Graph-based root cause analysis for service-oriented and microservice architectures," *J. Syst. Softw.*, vol. 159, Jan. 2020, Art. no. 110432, doi: 10.1016/J.JSS.2019.110432.

[61] F. Rademacher, S. Sachweh, and A. Zündorf, "Aspect-oriented modeling of technology heterogeneity in microservice architecture," in *Proc. IEEE Int. Conf. Softw. Archit. (ICSA)*, Mar. 2019, pp. 21–30, doi: 10.1109/ICSA.2019.00011.

[62] F. H. Vera-Rivera, "Método de automatización del despliegue continuo en la nube para la implementación de microservicios," in *Proc. 21st Conf. Iberoamericana Ingeniería Softw. (CIBSE)*, Bogotá, Colombia, 2018, pp. 597–604.

[63] H. Vural, M. Koyuncu, and S. Guney, "A systematic literature review on microservices BT—Computational science and its applications," in *Proc. ICCSA*, Trieste, Italy, 2017, pp. 203–217.
[64] L. Bao, C. Wu, X. Bu, N. Ren, and M. Shen, "Performance modeling and workflow scheduling of microservice-based applications in clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 9, pp. 2114–2129, Sep. 2019, doi: 10.1109/TPDS.2019.2901467.
[65] S. N. Srirama, M. Adhikari, and S. Paul, "Application deployment using containers with auto-scaling for microservices in cloud environment," *J. Netw. Comput. Appl.*, vol. 160, Jun. 2020, Art. no. 102629, doi: 10.1016/J.JNCA.2020.102629.
[66] D. Morris, S. Voutsinas, N. C. Hambly, and R. G. Mann, "Use of docker for deployment and testing of astronomy software," *Astron. Comput.*, vol. 20, pp. 105–119, Jul. 2017, doi: 10.1016/j.ascom.2017.07.004.
[67] A. Benayache, A. Bilami, S. Barkat, P. Lorenz, and H. Taleb, "MsM: A microservice middleware for smart WSN-based IoT application," *J. Netw. Comput. Appl.*, vol. 144, pp. 138–154, Oct. 2019, doi: 10.1016/j.jnca.2019.06.015.
[68] F. Lonetti and E. Marchetti, "Emerging software testing technologies," *Adv. Comput.*, vol. 108, pp. 91–143, Jan. 2018, doi: 10.1016/BS.ADCOM.2017.11.003.
[69] R. Khan and M. Amjad, "Performance testing (load) of Web applications based on test case management," *Perspect. Sci.*, vol. 8, pp. 355–357, Sep. 2016, doi: 10.1016/J.PISC.2016.04.073.
[70] BlazeMeter. *Engine Health Report*. Accessed: Feb. 7, 2020. [Online]. Available: https://guide.blazemeter.com/hc/en-us/articles/206733869-Engine-Health-Report-Engine-Health-Report
[71] R. Ramakrishnan and A. Kaur, "Little's law based validation framework for load testing," *Inf. Softw. Technol.*, vol. 108, pp. 88–98, Apr. 2019, doi: 10.1016/J.INFSOF.2018.11.007.
[72] F. Alharbi, A. Atkins, C. Stanier, and H. A. Al-Buti, "Strategic value of cloud computing in healthcare organisations using the balanced scorecard approach: A case study from a Saudi hospital," *Procedia Comput. Sci.*, vol. 98, pp. 332–339, Jan. 2016, doi: 10.1016/J.PROCS.2016.09.050.
[73] N. M. Kumar and P. K. Mallick, "The Internet of Things: Insights into the building blocks, component interactions, and architecture layers," *Procedia Comput. Sci.*, vol. 132, pp. 109–117, Jan. 2018, doi: 10.1016/J.PROCS.2018.05.170.
[74] T. Kiss, J. DesLauriers, G. Gesmier, G. Terstyanszky, G. Pierantoni, O. A. Oun, S. J. E. Taylor, A. Anagnostou, and J. Kovacs, "A cloud-agnostic queuing system to support the implementation of deadline-based application execution policies," *Future Gener. Comput. Syst.*, vol. 101, pp. 99–111, Dec. 2019, doi: 10.1016/j.future.2019.05.062.
[75] Z. Yi, W. Meilin, C. RenYuan, W. YangShuai, and W. Jiao, "Research on application of SME manufacturing cloud platform based on micro service architecture," *Procedia CIRP*, vol. 83, pp. 596–600, Jan. 2019, doi: 10.1016/j.procir.2019.04.091.
[76] K. Malyuga, O. Perl, A. Slapoguzov, and I. Perl, "Fault tolerant central saga orchestrator in RESTful architecture," in *Proc. 26th Conf. Open Innov. Assoc. (FRUCT)*, Yaroslavl, Russia, Apr. 2020, pp. 278–283, doi: 10.23919/FRUCT48808.2020.9087389.
[77] L. Zhong, L. Mu, J. Li, J. Wang, Z. Yin, and D. Liu, "Early prediction of the 2019 novel coronavirus outbreak in the mainland China based on simple mathematical model," *IEEE Access*, vol. 8, pp. 51761–51769, 2020, doi: 10.1109/ACCESS.2020.2979599.
[78] S. J. Fong, G. Li, N. Dey, R. G. Crespo, and E. Herrera-Viedma, "Composite Monte Carlo decision making under high uncertainty of novel coronavirus epidemic using hybridized deep learning and fuzzy rule induction," *Appl. Soft Comput.*, vol. 93, Aug. 2020, Art. no. 106282, doi: 10.1016/j.asoc.2020.106282.
[79] D. Dong, Z. Tang, S. Wang, H. Hui, L. Gong, Y. Lu, Z. Xue, H. Liao, F. Chen, F. Yang, R. Jin, K. Wang, Z. Liu, J. Wei, W. Mu, H. Zhang, J. Jiang, J. Tian, and H. Li, "The role of imaging in the detection and management of COVID-19: A review," *IEEE Rev. Biomed. Eng.*, early access, Apr. 27, 2020, doi: 10.1109/RBME.2020.2990959.
[80] L. Li, Q. Zhang, X. Wang, J. Zhang, T. Wang, T.-L. Gao, W. Duan, K. K.-F. Tsoi, and F.-Y. Wang, "Characterizing the propagation of situational information in social media during COVID-19 epidemic: A case study on Weibo," *IEEE Trans. Comput. Social Syst.*, vol. 7, no. 2, pp. 556–562, Apr. 2020, doi: 10.1109/TCSS.2020.2980007.

HURIVIADES CALDERÓN-GÓMEZ was born in Panama, in 1993. He received the B.E. degree in software development from the Technological University of Panama, Coclé, Panama, in 2016, and the M.E. degree in higher education from University Santander, Panama City, Panama, in 2019. He is currently pursuing the Ph.D. degree in information and knowledge engineering with the University of Alcalá, Alcalá de Henares, Spain.

From 2014 to 2016, he was a Research Assistant with the Technological University of Panama. In 2017, he joined the Technological University of Panama as a Research Associate. Since March 2019, he has been with the Department of Computer Systems Engineering, Technological University of Panama, where he was an Associate Professor. His current research interests include e-health, cloud computing, and software engineering.

LUIS MENDOZA-PITTÍ was born in Panama, in 1992. He received the B.E. degree in engineering systems and computing from the Technological University of Panama, Panama, in 2015. He is currently pursuing the Ph.D. degree in information and knowledge engineering with the University of Alcalá, Alcalá de Henares, Spain.

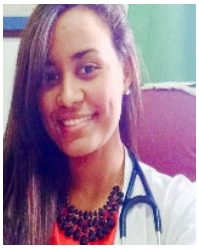
From 2013 to 2016, he was a Student Research Assistant with the Technological University of Panama. His current research interests include smart building, energy optimization, cloud computing, and software engineering.

MIGUEL VARGAS-LOMBARDO (Member, IEEE) received the Ph.D. degree from the Polytechnic University of Madrid. He is currently a member of the National Research System (SNI-SENACYT) of Panama. He is currently a Professor and a Researcher with the Technological University of Panama.

JOSÉ MANUEL GÓMEZ-PULIDO was born in Cáceres, Spain. He received the B.S. and M.S. degrees in telecommunications engineering from the University Polytechnic of Madrid, Spain, in 1988 and 1994, respectively, and the Ph.D. degree in telecommunication from the University of Alcalá, Madrid, in 2004. In 1992, he was an Assistant Researcher and a Professor with the University Polytechnic of Madrid. From 1993 to 2004, he was with the Signal Theory and Communications Department, University of Alcalá, where he is currently an Assistant Professor with the Computational Science Department. He has participated in more than 24 research projects several research projects with Spanish and European companies, related with radio propagation in mobile communications, signal processing in radar, ultrasound and laser applications, EMI–EMC design in on-board satellite instrumentation, radio propagation materials proprieties, numerical technics applied in telecommunication, and so on. His research interests include the areas of wireless sensors networks, smart grids, and new technologies in the data mining and artificial intelligence world applied to the ICT.

JOSÉ LUIS CASTILLO-SEQUERA received the degree in computing from the National University of San Marcos, Peru, the master's degree in computer project management and the master's degree in university teaching in Spain, and the Ph.D. degree in information systems, documentation, and knowledge from the University of Alcalá, Spain. He is currently a Full Professor with the Department of Computer Science, University of Alcalá. He has multiple publications at the level of high-level indexed JCR journals and is the author of books and book chapters related to the field of artificial intelligence. His research interests include information retrieval, knowledge extraction based on evolutionary learning and neural network, data mining and big data, documentation, learning, and teaching innovation with Web 2.0. He also leads projects of technological innovation oriented to the use of ICTs in university research groups. His research interests include business technologies and intelligent technologies for knowledge management with interdisciplinary technologies.

JOSÉ SANZ-MORENO was born in Spain, in 1956. He received the degree in internal medicine. He is currently the Chief of the Division of Infectious Diseases, Hospital Universitario Principe de Asturias, Alcalá de Henares, Madrid, Spain. He is also a Professor with the Department of Medicine and Medical Specialties, University of Alcalá, Spain.



GLORIA SENCIÓN received the degree in nephrologist from the Hospital Príncipe de Asturias, Alcalá de Henares, Spain, in 2015. She is currently a Professor with the Department of Medicine, Autonomous University of Santo Domingo, Dominican Republic.