



Digital health in smart cities: Rethinking the remote health monitoring architecture on combining edge, fog, and cloud

Vinicius Facco Rodrigues¹ · Rodrigo da Rosa Righi¹ · Cristiano André da Costa¹ · Felipe André Zeiser² · Bjoern Eskofier² · Andreas Maier³ · Daeyoung Kim

Received: 22 February 2023 / Accepted: 6 April 2023 / Published online: 27 April 2023

© The Author(s) under exclusive licence to International Union for Physical and Engineering Sciences in Medicine (IUPESM) 2023

Abstract

Purpose Smart cities that support the execution of health services are more and more in evidence today. Here, it is mainstream to use IoT-based vital sign data to serve a multi-tier architecture. The state-of-the-art proposes the combination of edge, fog, and cloud computing to support critical health applications efficiently. However, to the best of our knowledge, initiatives typically present the architectures, not bringing adaptation and execution optimizations to address health demands fully.

Methods This article introduces the VitalSense model, which provides a hierarchical multi-tier remote health monitoring architecture in smart cities by combining edge, fog, and cloud computing.

Results Although using a traditional composition, our contributions appear in handling each infrastructure level. We explore adaptive data compression and homomorphic encryption at the edge, a multi-tier notification mechanism, low latency health traceability with data sharding, a Serverless execution engine to support multiple fog layers, and an offloading mechanism based on service and person computing priorities.

Conclusions This article details the rationale behind these topics, describing VitalSense use cases for disruptive healthcare services and preliminary insights regarding prototype evaluation.

Keywords Digital health · Smart city · Remote health monitoring · Internet of things · Fog computing · Edge computing

✉ Rodrigo da Rosa Righi
rrighi@unisinis.br

Vinicius Facco Rodrigues
viniciusf@unisinis.br

Cristiano André da Costa
cac@unisinis.br

Felipe André Zeiser
felipez@unisinis.br

Bjoern Eskofier
bjoern.eskofier@fau.de

Andreas Maier
andreas.maier@fau.de

Daeyoung Kim
kimd@kaist.ac.kr

¹ Applied Computing Graduate Program, Universidade do Vale do Rio dos Sinos (UNISINOS), São Leopoldo, Brazil

² Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Erlangen, Germany

³ Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea

1 Introduction

Over the last few years, the health sector has understood that the Internet can be an essential support instrument in searching for a better quality of life and better conditions for patient care [1, 2]. Among other advantages associated with using the Internet in the health field, the analysis and processing of data in real-time through remote servers have been highlighted. An efficient model capable of providing storage and processing of applications over the Internet is the concept of cloud computing [2]. This model can be described as a service provided by large data centers that offer part of their infrastructure - hardware and software - to third parties (companies and individuals) and public organizations. Once these customers purchase this type of service, they will have computing resources with increasing capacity without the need for significant financial capital investments to acquire, maintain and manage such resources.

Cloud computing is the primary support to enable the Internet of Things (IoT) applications. IoT environments are composed of hundreds or thousands of devices that

constantly generate requests for collected data to be later analyzed. In the context of Healthcare 4.0 [3], the Internet of Health Things (IoHT) represents the spread of IoT devices applied to e-health applications [4]. This process naturally generates heavy requests sent to a central processing server, flooding that server’s network and requiring computational power that a single computer would often not be able to supply [5]. Here, cloud computing can be used as a processing medium for IoT scenarios to leverage its scalability and pay-as-you-go model. However, sending requests from an IoT device to a cloud server adds network latency overhead to the communication that cannot be accepted in some cases. For example, we can cite some e-health scenarios, such as those addressing remote electrocardiogram (ECG), where data collecting and processing times are critical to the correct system functioning. We often cannot wait for a message to be sent, processed in the cloud, and returned, as the time involved in these procedures can influence essential aspects of time-critical applications [5, 6]. Furthermore, even with a highly scalable cloud computing environment, scaling it to serve many requests would result in additional power consumption.

To allow better scalability of IoT systems, it is necessary to design new architectures and solutions that allow handling many devices and requests simultaneously, maintaining the Quality of Service (QoS) [7]. Aligned with this sentence, fog

computing expands the services offered by the traditional cloud model to be closer to the data generators [8, 9]. Also, edge computing enters here to enable some processing and decision support precisely on the network’s border, i.e., close to the IoT device itself. Computing in fog or edge has as its main characteristics low latency, better support to collect the geographic distribution of data, and mobility over a large number of nodes in the network. Thus, with predominantly wireless access, we have the execution of applications in real-time and more significant support for device heterogeneity. Data read by the sensors is collected, processed, and stored in a temporary database instead of delivered to the cloud, avoiding round-trip delays in network traffic.

A combination of cloud, fog, and the edge is especially pertinent to provide an architecture to answer healthcare and pandemic research, such as the case of Coronavirus disease (COVID-19) [10]. More significantly, we are entering a period where long-COVID-19 research is mainstream. The purpose is to continuously monitor the vital signs of those contaminated by the virus beforehand [11]. In remote health monitoring, most vital sign monitoring systems follow a generalized three-tier architecture composed of sensing devices, fog gateways, and the cloud [12] (see Fig. 1(a)). By analyzing the current initiatives in the literature, they do not address all issues concomitantly as follows: (i) person’s

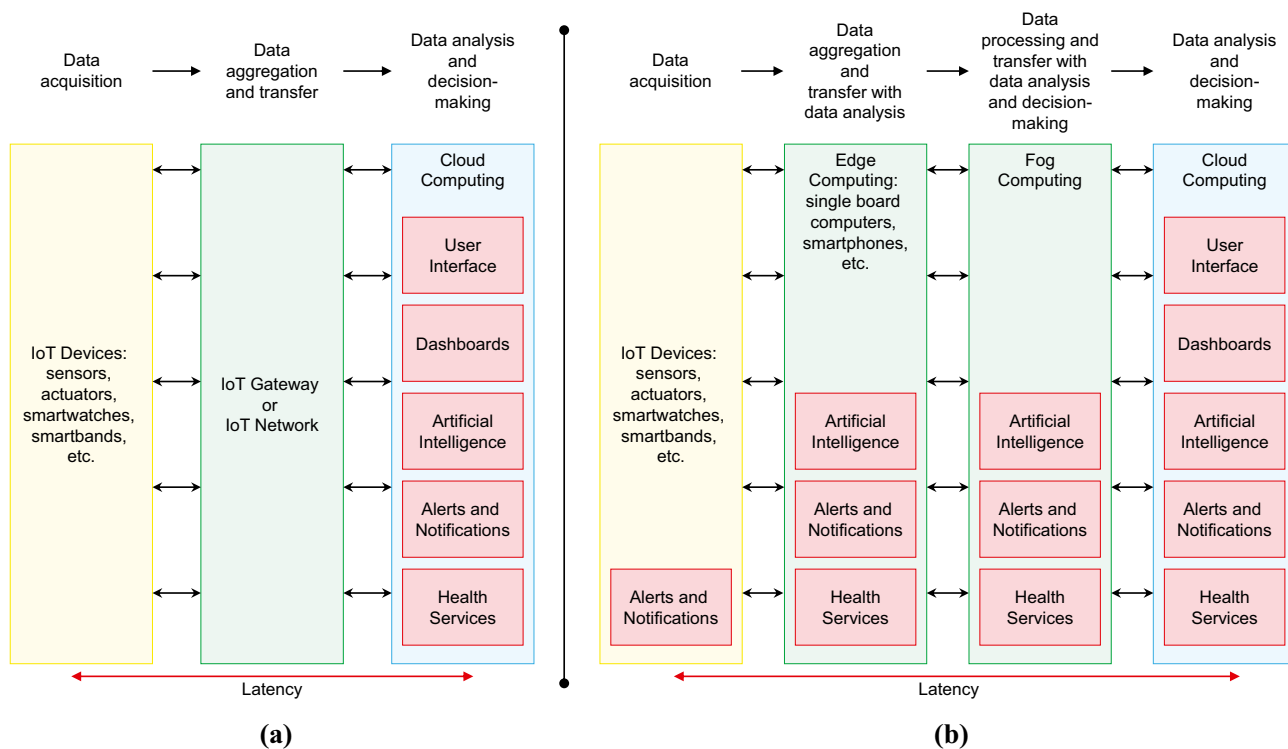


Fig. 1 Comparison between traditional (a) strategies for health monitoring and VitalSense (b). Applications can access health services from anywhere

traceability and privacy, both in terms of historical view of vital signs or places visited in a smart city; (ii) artificial intelligence to execute health services proactively, generating value for end-users, in addition to hospitals and public sector; (iii) state-of-the-art mechanisms to address QoS, elastic processing capability and an efficient and scalable message notification system.

In this context, this article introduces VitalSense as a novel smart city architecture that combines edge, fog, and cloud to manage historical health data of citizens, acting primarily in the fight against COVID-19. Figure 1(b) demonstrates the VitalSense approach, where we rely on a hierarchical fog topology and a distributed computing engine distributed along with the three involved tiers. Hospitals, individuals through wearables, homes, and certain places in cities would rely on sensors and readers so that vital signs and geolocation data are collected to serve as input to health services. The outcome brings benefits both to the public sector and to own citizens of the smart city. In VitalSense, for example, we envisage that the system can proactively call an ambulance to a particular person's home, taking into account the AI-based processing of historical vital sign data. Also, real-time historical data allows us to generate dashboards to verify the degree of spread of the SARS-CoV-2 (severe acute respiratory syndrome coronavirus 2) virus through the places that a particular person has visited and analyze the effectiveness of quarantine actions. Although using a traditional composition of edge, fog, and cloud, our contributions appear in how we handle each infrastructure level. In detail, our addition to the literature is a remote health monitoring architecture that explores:

- Adaptive data compression and homomorphic encryption in the edge;
- A multi-tier notification mechanism;
- Low latency health traceability with data sharding;
- Serverless execution elasticity engine involving multiple fog layers with an offloading mechanism based on service and person computing priorities;
- Live geographical-based dashboards for data visualization.

The remaining of this article first introduces the related work in Section 2. Here, we provide literature discussion, highlighting research opportunities not covered up to this moment. Second, Section 3 presents the VitalSense Model, describing its architecture, processes, and algorithms. Third, Section 4 is in charge of discussing health services and use cases. This section shows health services for the future, and how they match with the edge, fog, and cloud architecture proposed. Finally, Section 5 concludes the article, detaching our achievements to improve citizens' quality of life worldwide.

2 Related work

This section introduces the main articles in the literature on the remote health monitoring scope, describing their main characteristics (Table 1). Following, Section 2.1 describes in detail each article, and Section 2.2 discusses the main findings.

2.1 State of the art

The current state-of-the-art presents several articles focusing on remotely monitoring patients' vital signs. In general, cloud computing is commonly employed in such solutions combined with edge or fog [12–21]. Nevertheless, some explore different strategies without employing cloud [22, 23]. Sicari et al. [22] propose a quarantine home monitoring system composed of IoT sensors and Global Positioning System (GPS) tracking. The first goal is to collect and analyze vital signs according to predefined thresholds generating alerts. In addition, the system performs GPS monitoring to track if the patient moves outside its quarantine area. The proposal does not focus on any QoS parameter and relies only on the Internet and Global System for Mobile communication (GSM) networks. Li et al. [14] propose a remote monitoring system that tracks respiratory rate deterioration for patients with COVID-19. The authors introduce a non-invasive respiratory rate estimation technique using WiFi. It transmits data to the cloud for patient monitoring and triggering alerts.

Rahman et al. [13] propose a framework to collect and analyze physiological data from patients remotely. The system employs wearable sensors and a smartphone running the developed application to collect, process, and post data to the cloud. In addition, the system uses the location of each patient according to their address. The patient must manually initiate the collection and analysis process according to a defined protocol. Al Bassam et al. [15] propose a wrist wearable device equipped with temperature, oxygen saturation (SpO₂), heart rate, and sound sensors providing remote data to monitor and manage potential COVID-19 infected patients. The strategy uses a microcontroller to send data to a cloud back-end. According to the readings, the system can generate Email and Short Message/Messaging Service (SMS) alarms. In addition, a mobile application allows data visualization from tracked patients.

Paganelli et al. [16] present a conceptual architecture for remote monitoring of patients with COVID-19. The architecture considers data acquisition from users at home and in medical wards. The authors define a three-tier architecture in which mobile and static gateways connect the IoT layer with a data distribution layer through a message broker

Table 1 Comparison among studies proposing remote monitoring systems to track patients parameters: body temperature (BT), blood pressure (BP), blood oxygen saturation (SpO2), heart rate (HR), respiratory rate (RR), heart rate variability (HRV), and electrocardiogram (ECG)

Ref.	Year	Focus	Vital Signs	Level	Technique	QoS
[22]	2022	Patient with COVID-19 remote monitoring	BT, BP, SpO2, HR, RR	IoT, Edge	Threshold monitoring, dashboard visualization, SMS alerts, and GPS tracking.	-
[13]	2022	Remote vital sign monitoring	BT, SpO2, HRV	IoT, Edge, Cloud	Wearable sensors, data scanner, and smartphone running pre-trained ML algorithms to check possible diseases and share results remotely.	latency
[12]	2022	Remote vital sign monitoring	BT, SpO2, HR, ECG, RR, BP	IoT, Edge, Cloud	Traditional three-tier architecture: sensors, gateway, and cloud; MCU unit collects data from multiple sensors and transmits it to a smartphone application for processing, analysis, and alarm triggering; The smartphone acts as a gateway to send data to the cloud.	-
[14]	2021	Patient with COVID-19 remote monitoring	RR	Edge, Cloud	Respiratory rate estimation using WiFi, transmission to the cloud, visualization and alerts.	-
[15]	2021	Patient with COVID-19 remote monitoring	BT, SpO2, HR, cough count	IoT, Edge, Cloud	Wearable sensors providing health parameters and GPS information; A microcontroller acquires and sends data to cloud where Email/SMS alerts can be triggered; Android app for data visualization.	-
[16]	2021	Patient with COVID-19 remote monitoring	BT, HR, SpO2, RR, BP, and electrocardiogram (ECG)	IoT, Edge, Cloud	Three layer architecture in which a micro-controller collects and processes data from IoT sensors and transmit messages and alarms to gateways; Gateways publish messages in a remote message queue; Applications consume data from the message queue for complex processing.	-
[17]	2021	Remote vital sign monitoring	BP, BT, pulse rate, SpO2	IoT, Edge, Fog, Cloud	Smartphone collects data from health sensors through Bluetooth and transmits data to the fog; fog nodes process data and notify patient if abnormalities occur, also sending notification to the cloud; cloud aggregate results and send alerts to the patient smartphone.	delay, energy
[18]	2021	Health sensor data communication framework	-	IoT, Fog, Cloud	Load balance strategy that chooses between fog or cloud to process data; fog provides filtering, fusion, compression, aggregation, intermediate data analytic and local storage for reducing the load on the cloud.	latency, loss, throughput
[19]	2020	Remote vital sign monitoring	HR, RR, SpO2, BP	IoT, Edge, Cloud	Smartphone collects data from health sensors through Bluetooth and transmits data to the cloud; cloud services process data and generate alarms with a severity level; Depending on the level the system dispatches a drone to deliver medication.	-

Table 1 (continued)

Ref.	Year	Focus	Vital Signs	Level	Technique	QoS
[20]	2020	Patient with COVID-19 remote monitoring	SpO2, HR	IoT, Edge, Cloud	Android application on smartphone acting as a gateway for Bluetooth Low Energy (BLE) sensors; Remote server employing Logstash, Elasticsearch, and Kibana.	-
[23]	2018	Remote vital sign monitoring	HR, RR, SpO2, BP, BT, ECG, environment parameters	IoT, Edge	Distributed gateways close to wireless sensor networks acquiring data, providing computing services, and sending data to the cloud; Services provided: data filtering, data fusion, data analysis, compression, storage, security, and notification.	latency
[21]	2016	Remote vital sign monitoring	BP, and BT	IoT, Cloud	Threshold monitoring, dashboard visualization, SMS alerts, and GPS tracking.	-

network. The architecture comprises an IoT kit composed of a micro-controller and radio transceiver to connect to the closest gateway.

Rahmani et al. [23] propose a smart gateway at the network’s edge to collect, process, and transmit data to the cloud. It provides several services, including compression, filtering, security, data analysis, and fusion. The main goal is to generate notifications, prioritize critical data, and decrease the data volume, thus reducing latency when sending data to the cloud. Moving patients change the base gateway according to the coverage. Sahu et al. [12] propose a three-tier architecture for vital sign monitoring. The authors employ a microcontroller unit (MCU) wired to several body sensors. It extracts data from them and transmits it to a smartphone application through Bluetooth. The application can provide data analysis and alarms for sensor measurements. The system can also send data to a cloud server using an Internet connection; however, the main focus of the proposal is the local processing and alarms.

Sangeetha et al. [19] propose a remote health monitoring system for rural areas comprising medicine delivery using drones. The strategy consists of a smartphone acquiring data from health sensors and uploading it to a cloud server through the Internet. Data analytics services process data in the cloud and generate alerts when it identifies abnormalities. Depending on the severity, caregivers, doctors, or ambulance services can be alerted. Mukherjee et al. [17] propose a edge-fog-cloud framework for IoHT that predicts patients conditions and generates alerts. They developed a workflow that collects data from a body area network (BAN) using an edge device and forwards data to fog nodes. Notifications and further processing take place only when fog processing identifies abnormalities.

Chudhary and Sharma [18] propose a fog-cloud framework for IoHT comprising four layers: data collection, fog, cloud, and application. Fog nodes process data locally to provide faster response time while re-transmitting data to the cloud when local processing is not required. In turn, the application layer provides an interface for users to access processed data from both the cloud and fog layers. Aziz et al. [21] propose a real-time monitoring system to track blood pressure and body temperature from remote patients. The solution relies on GSM connectivity to send alerts to a remote server when the sensors’ readings violate pre-defined thresholds. Then, the back-end server relays the messages to the Doctor’s mobile phone. In addition, the system attaches GPS data to the alerts so that the medical team can send an ambulance if it is the case. Priambodo and Kadarina [20] detail a COVID-19 patient remote monitoring to track the patient’s SpO2, heart rate, and location. A smartphone acts as a gateway to collect data from sensors, attach identification and GPS information, and then transmit it to a remote

server. In turn, the remote server employs Logstash, Elasticsearch, and Kibana for data processing so doctors can analyze patients' data.

2.2 Discussion

As Table 1 shows, most articles focus on vital sign monitoring in general [12, 13, 17, 19, 21, 23] or patient with COVID-19 monitoring [14–16, 20, 22]. Commonly, they aim at monitoring vital signs such as: (i) body temperature (BT); (ii) blood pressure (BP); (iii) blood oxygen saturation (SpO₂); (iv) heart rate (HR); (v) respiratory rate (RR); (vi) heart rate variability (HRV); and (vii) electrocardiogram (ECG). IoT plays a critical role in these strategies as they rely on small, mobile sensor devices. In addition, most authors employ different computing levels in their solutions, combining edge or cloud technologies with IoT devices.

Although beneficial, fog computing appears in only two articles [17, 18] as a solution to improve the performance of such systems. Solutions can benefit from closer data processing units the fog provides, improving response time. Critical applications from the healthcare domain require fast processing, and excessive delays can compromise their purpose. Current literature lacks a proposal exploring fog in such scenarios to improve response time for critical situations. That leads to the current research that combines fog with IoT, edge, and cloud to improve response time.

3 The VitalSense model

VitalSense aims to provide a large-scale real-time health monitoring system for smart cities. Its main characteristic relies on employing fog and edge to deliver close data processing for patients and health administrators. It also employs the cloud for data aggregation and analysis. As data processing is at the fog and edge, users benefit from a fast response time that is critical for medical applications. The use of remote health monitoring can significantly improve the delivery of health services to the population. In traditional health care, patients visit a medical facility for periodic health checks or to treat health issues. Such a scenario shows the reactivity of the public healthcare system that requires a patient's action of a sudden problem to deliver health services. VitalSense offers a proactive approach to delivering health services, regardless of the patient's actions.

Aside from vital sign monitoring, VitalSense also comprises people's motion tracking. Real-time location systems (RTLS) [24] can integrate the architecture by injecting people tracking information at the edge of the architecture. RTLS improves the understanding of movement patterns and exposure to risk locations by providing indoor and outdoor positioning information. Currently, there are several

RTLS technologies for different objectives depending on the required precision, including Radio-Frequency Identification (RFID), Ultrawideband (UWB), and GPS. VitalSense relies on RFID city portals that can track when a person accesses or leaves a particular region. Data acquired from portals generate information that can support several services. For instance, crowd monitoring is crucial for pandemic monitoring and hazard control. Tracking people's movements from region to region helps monitor their exposure to potential harm. Therefore, the public administration can take action beforehand when critical situations are prone to happen.

With concepts such as the IoT, Big Data, and Industry 4.0 (I4.0) [25] becoming ubiquitous in industries, including e-health ones, the exponential growth of temporal data generated by IoT sensors is likely to continue. This data is vital for the companies; analyzing it and acting upon the results can optimize production quality, save energy, and improve the quality of life for end-user citizens. However, when analyzing such large amounts of data, we have a trend of stressing the computer and the network. In this context, by selecting more efficient transfer protocols, an IoT-based application can significantly increase performance.

As the article exposes, when designing data traffic in a smart city, we can deal with excessive network traffic that we could easily avoid if the middleware only sends the number of data points needed. In summary, we perceived in the scientific literature that today there is no existing solution capable of optimizing IoT data traffic effortlessly while achieving a lossless data representation. Usually, approaches compress data on the server side while decompressing it on the client side. This idea suffers from calibrating the overhead regarding compression and decompression activities and the possible gains in transmission time, in addition to modifications mainly on the client side.

Our proposal for data compression resides at the edge. The main idea is to send vital sign data to the fog in accordance with the person's status and the subscribed services. For example, we can send data to the fog more frequently for babies or the elderly. Also, if at the edge we detect some health complication (such as the data collected is outside predefined thresholds for a particular vital sign type) or if a specific person has a chronic disease or recovering from medical treatment, we can maintain a high frequency on edge-fog data transmission. However, healthy and mid-age people can work with higher transmission periods without compromising the detection of health problems. It is important to note that the transmission frequency is adapted per person, being dynamic along with the functioning of the architecture.

3.1 Architecture

I4.0 recently brought several IoT sensors to the real world that hospitals can now employ in health services, called

Healthcare 4.0. VitalSense relies on those concepts to build an architecture capable of collecting patients’ data regardless of location. Figure 2 presents VitalSense architecture composed of three layers: edge, fog, and cloud. Individuals at the network’s edge wear off-the-shelf health sensors to monitor their vital signs during daily activities. VitalSense introduces Edge Controllers to capture data from sensors providing data processing, transmission, and feedback. Edge Controllers are devices placed close to sensors capable of reaching them to extract data. For instance, an Edge Controller can be either a smartphone or a single board computer installed at the user’s house.

At the fog layer, geographically distributed fog nodes offer a near data processing infrastructure. Fog nodes provide several health services providing notifications for Edge Controllers and producing reports to public administrators. In contrast to Edge Controllers, which do not communicate with one another, fog nodes can exchange data and offload data processing [26]. In addition, fog nodes form a hierarchy

tree in which parent nodes aggregate information from child nodes to produce region knowledge.

Finally, the cloud layer provides a central platform that aggregates data from the entire deployment. Cloud services employ a more robust infrastructure for the deep analysis of data. In addition, health administrators have a broader view of several regions at the cloud layer. That allows solutions capable of identifying risk zones and outbreaks of diseases at their earliest. VitalSense can scale its services using cloud infrastructure for resource elasticity. Therefore, it supports a growing architecture where more fog nodes can be integrated into the solution.

When deploying the VitalSense architecture, some technologies can guarantee data security and QoS. Dedicated network links can provide private networks to allow end-to-end communication between the architecture modules. Besides, private networks provide more reliable communication channels improving QoS. The use of Multi-Protocol Label Switching (MPLS) [27] and Software-Defined Wide

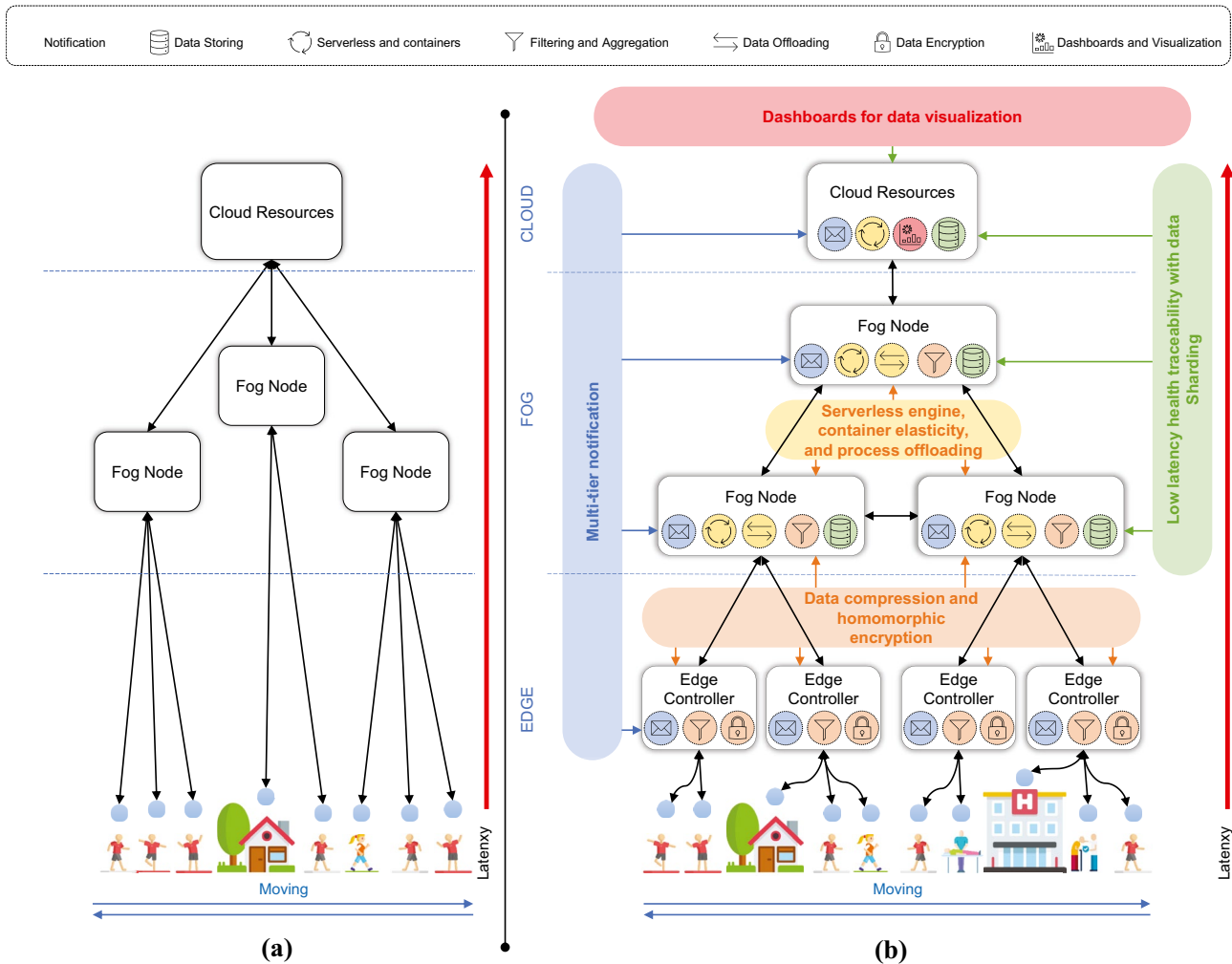


Fig. 2 Traditional fog architecture a versus VitalSense’s architecture design b employing edge, fog, and cloud

Area (SD-WAN) [28] network can support QoS when working on a shared network environment. Fog providers can employ such technologies at the network level to provide dynamic QoS strategies according to needs.

3.1.1 Edge layer

The edge layer comprises several sensors collecting data from people and the environment. These sensors can monitor vital signs and transmit them to the VitalSense. Individuals can wear a sensor to monitor several health parameters during daily activities. Wireless technology allows data extraction and collection without requiring the individual to offload the data manually. VitalSense places the Edge Controller module in this layer responsible for data collection. Such a module also provides data pre-processing before transmitting it to the upper layers, allowing close service provisioning to the users through generating alerts for both users and the upper layers.

Managing personal data is always challenging regarding privacy in the IoT world. Currently, the General Data Protection Regulation¹ (GDPR) defines several criteria to enhance individuals' control and rights over their data. VitalSense addresses that by proposing a robust network topology that places the Edge Collectors as entry points directly connected to the sensors. Thus, it can provide data encryption services at the network's border protecting the users' data. In particular, this layer employs asymmetric cryptography (RSA 2048) to transfer data to the fog. In addition, VitalSense does not collect identifying information and only transmits anonymized data into the system. Upon registering to use the system, VitalSense requires the subject to consent to processing their data. VitalSense generates a unique hash code for each individual to label the data without transmitting identification information along with it.

Figure 3 details the edge layer operation including the Edge Controller architecture. At the bottom, the Edge Controller provides a sensor middleware able to connect to sensors via three different strategies: (i) subscribe to data streams; (ii) collect data using sensors APIs; (iii) or pull data from vendor applications. The middleware provides raw data from sensors to the data processing module. Data coming from sensors flow into two different parallel phases. First, Edge Controllers must prepare the data before it can be transmitted to the fog. In this phase, the module filters, aggregates, compresses and encrypts several data samples. On the one hand, filtering, aggregation, and compression reduce the amount of data, improving network performance. On the other hand, encryption ensures data protection.

During the second phase, the Edge Controller generates notifications employing lightweight services. It analyses raw data samples employing prediction, correlation, and classification algorithms. The notification module generates notifications based on the output of the services. Depending on the Edge Controller's hardware, it can perform notification actions such as turning an LED on, popping up a message on the screen, or sending a message to a network service.

The Edge Controller connection with the fog layer occurs in a two-way publish-subscribe model. First, the Edge Controller publishes data to the fog through an Message Queuing Telemetry Transport (MQTT) server to which fog nodes subscribe. Second, the Edge Controller subscribes to data streams from the fog to listen for notifications. The connection protocol between these layers occurs through a handshake process. Edge Controllers do not know which fog node they should connect. Fog nodes firstly reach Edge Controllers to inform a new connection. The Fog Connector module handles the new connection and connects to the notification MQTT server from this particular fog node. Therefore, once a fog node connects to read data, the Edge Controller also listens for notifications from this new fog node.

3.1.2 Fog layer

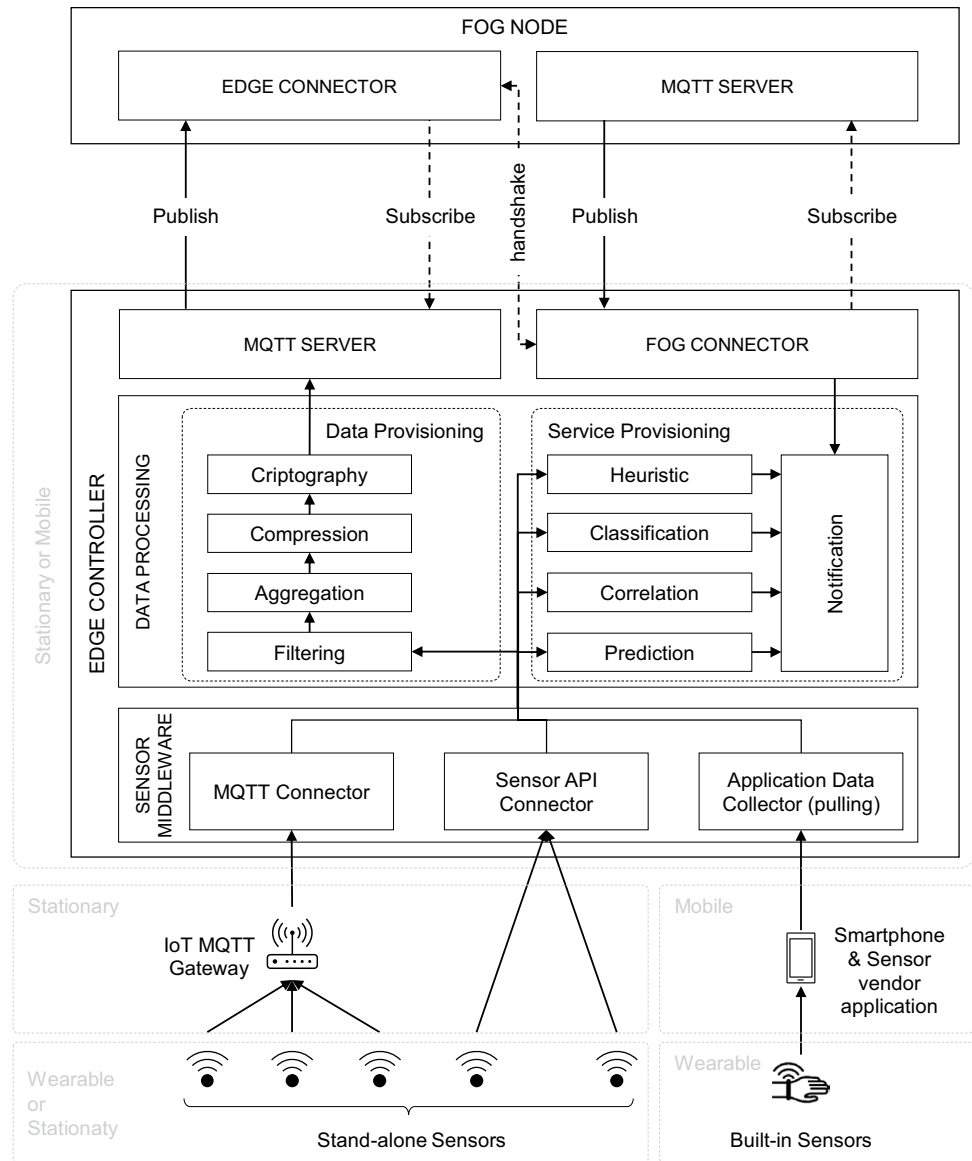
The fog layer comprises several geographically distributed nodes (fog nodes) responsible for processing data from the edge and providing services for both edge and cloud layers. Fog nodes form the core of the architecture since they process data from sensors by applying different algorithms for data analysis. Results from such analysis can generate notifications to either edge or cloud components. In addition, fog nodes reside near the edge of the network, which reduces communication delay improving response time. Figure 4 depicts a fog node architecture and the modules that compose it.

Fog nodes provide two different platforms for data processing: (i) serverless functions; and (ii) container pool. Different from the cloud, the resources are more limited in the fog. Therefore, resource sharing is challenging in such a solution, given that a fog node processes data from multiple users simultaneously. Serverless provides a quick way of executing computing tasks in shared resources. However, some tasks might require more computing time than others and suffer from timeout limitations of serverless platforms. VitalSense proposes the combination of both serverless and a container pool to execute tasks. The container pool can support data processing for tasks requiring increased time. The Data Orchestrator is responsible for choosing each solution according to resource consumption and availability.

VitalSense proposes a network topology where fog nodes are organized in a hierarchical structure. Each fog node can subscribe to one parent and several child nodes to receive three types of messages: (i) notifications; (ii) service data;

¹ <https://gdpr.eu/>

Fig. 3 Edge layer architecture. The sensor middleware collects data from sensors supporting different interfaces. Data is then processed and transmitted to the fog, while local processing can generate notifications



and (iii) data offload. A given fog node can subscribe to notifications from its child nodes and generate new alert messages aggregating data from multiple sources. In turn, service data regards messages containing resulting computation from particular services. The Data Analysis module comprises several services which produce an output that is published in the MQTT server. Both parent and child nodes can subscribe to receive service data and use the information in their local processing.

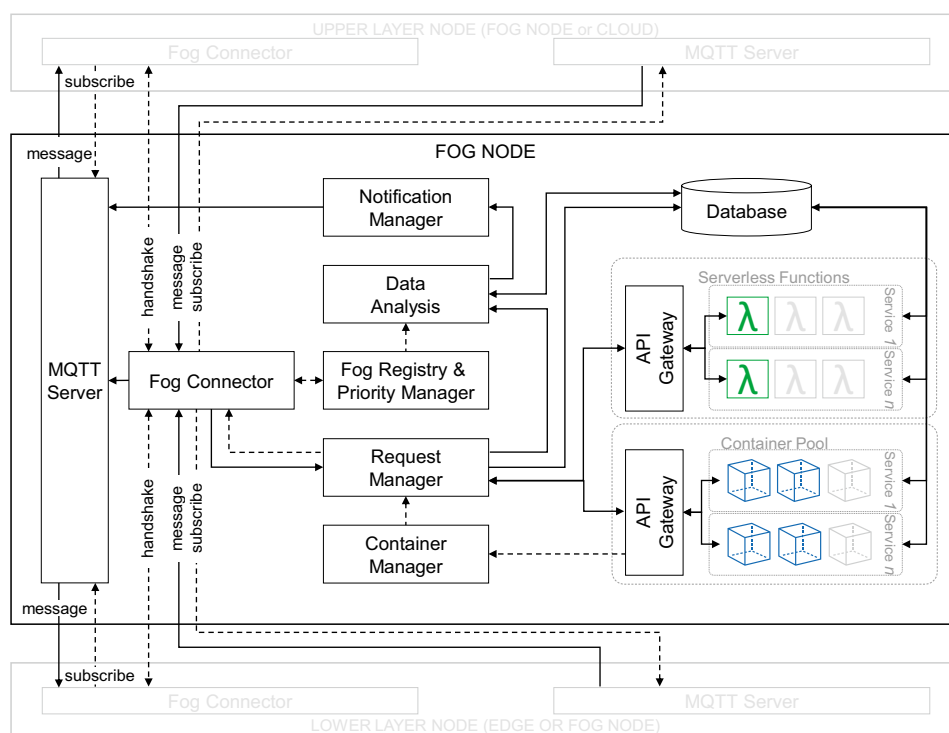
A particular fog node has limited resources which may lead to an overloaded situation according to the number of users. In such cases, the Fog Connector can forward data for processing to a parent node. The Request Manager decides whether resources are available, informing the Fog Connector in which direction requests should flow. In case of offload

needs, the Fog Connector publishes data to the MQTT server, which the parent node consumes data for processing.

Besides the parent/child topology, fog nodes can also connect to brother nodes. Brother nodes are the physically closer nodes from a particular fog node. The communication channel is important to migrate a user session when the user is moving from one region to another. Therefore, brother nodes can perform a handoff strategy so the system can maintain the user track and history.

Privacy and security is also an important issue when handling patients’ data. Fog nodes generate an RSA 2048 key pair to enable data encryption at the edge layer. Edge Controllers use the public key to encrypt data, while fog nodes use the private key to decrypt it. In addition, VitalSense employs homomorphic encryption [29] algorithms at this

Fig. 4 Fog layer architecture. Data is received from many Edge Collectors to be analyzed and processed by different services. Fog nodes can generate notifications according to the service and analysis processing



layer which guarantees that all data flowing in the system is secure. Raw data can be transferred to third-party services and public clouds, which can perform arithmetic computations with the data without accessing them. These services cannot see the actual data and need to request a fog node to decrypt the result of their computation.

3.1.3 Cloud layer

The cloud layer is the data aggregator from the entire architecture. It connects to the fog nodes on the top of the hierarchy to receive notifications and aggregated data. The cloud combines data from all sources providing a broader overview of the architecture. Global analysis is possible at this layer through data prediction and classification algorithms. Applications can use cloud features to generate critical insights for public administrators. Dashboards, views, and graphs are examples of real-time applications that can monitor regions. Besides, data reports can provide a valuable understanding of patterns and risk regions based on data processed at the fog.

Figure 5 demonstrates a data visualization dashboard at cloud layer. The dashboard can show regions concentrating abnormalities on measurements of a particular vital sign. Figure 5 demonstrates a broader view of the city of Porto Alegre, showing red circles in regions where several individuals present fever. In turn, Fig. 5b shows the same dashboard but zooming in on a particular region where the concentration is high. The public administrator can visualize

the number of individuals and their position in that particular region. Such a dashboard can help decision-making by the public administrator that can proactively act to mitigate the risks for the population.

3.1.4 Network organization

Figure 6 depicts VitalSense's network organization. The interconnection between sensors, Edge Controllers, fog nodes, and cloud forms a hybrid star-mesh topology. At the border, Personal Area Networks (PAN) comprise the patient's sensors connected to an Edge Controller to produce data. Alternatively, sensors can form a short-range network in which a master node collects its data to transmit to the Edge Controller. Wireless technology, such as Bluetooth and WiFi, are predominant at this layer. However, an Ethernet connection between the master node and the Edge Controller is also possible.

As an entry point to the system, the Edge Controllers live close enough to PANs while connecting to fog nodes in Local Area Networks (LAN). Each Edge Controller connects to a single fog node, and they can change according to the Edge Controller's mobility. Mobile Edge Controllers change their fog node over time according to network proximity. Mobility is only possible when employing WiFi and 5 G technology to connect the fog. When employing Ethernet technology, Edge Controllers are fixed at houses, hospitals, stores, and others. Fiber at this layer can provide a secure and private connection to the fog, supporting high rate and low latency data transmission.

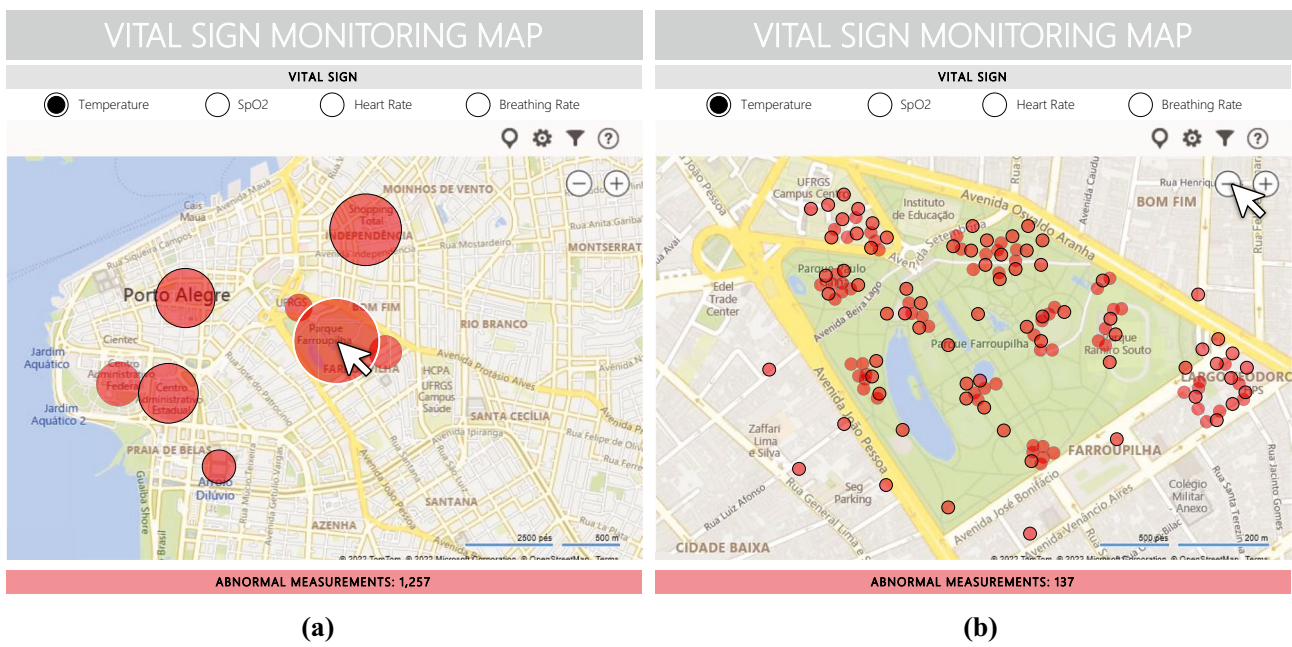
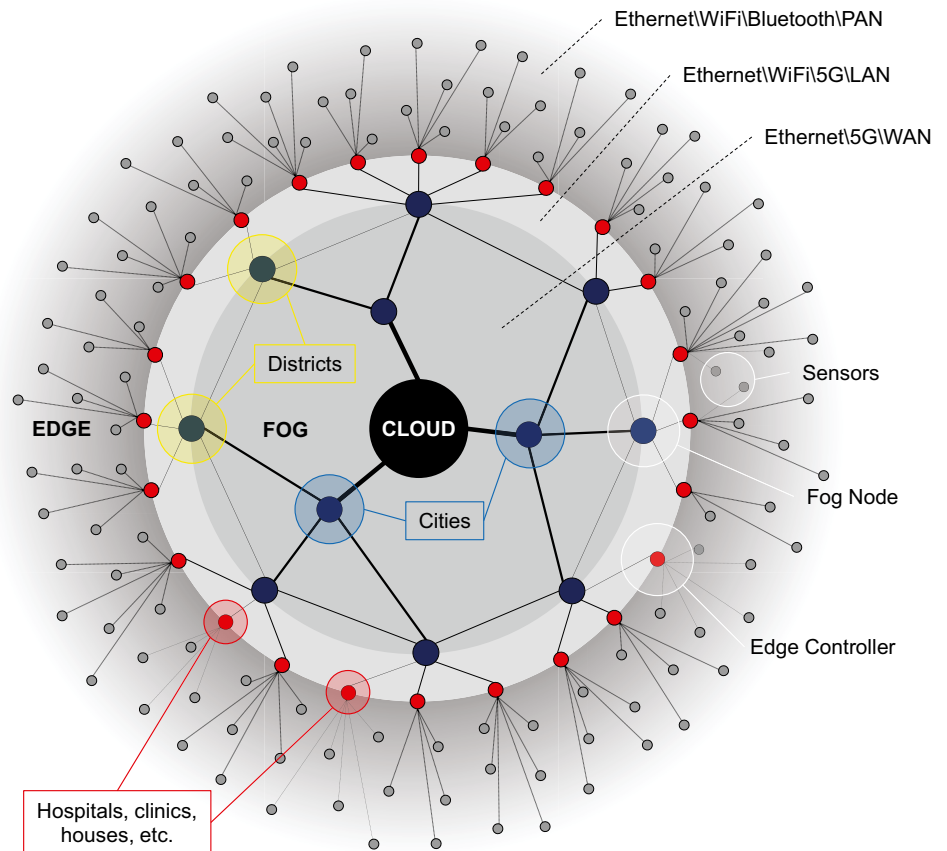


Fig. 5 Dashboard for map visualization and navigation according to vital sign measurements out of normality: **a** shows a broader view of a city, while **b** zooms in on a particular region of interest

At the fog, a hierarchical mesh network topology builds the path to the cloud providing low latency services to the edge. Fog nodes closer to the edge connect to Edge Controllers to

consume data and provide services. These fog nodes can also connect to several neighboring fog nodes according to physical and network proximity. Moving Edge Controllers can only

Fig. 6 Network topology



change to one of the neighbors, and the neighboring allows handoff between the fog nodes involved in the process. Additionally, fog nodes can connect either to a parent node or the cloud. Parent fog nodes allow data processing offloading if a particular fog node is overloaded. Further, parent fog nodes comprise physical regions providing specific services for a particular region they monitor.

The physical distribution of fog nodes and their network topology should be guided by urban segmentation (districts, cities, and others). The granularity of this segmentation is flexible according to the needs. For instance, it is possible for a particular case distributing one fog node per district connected to a parent fog node of a city. Districts with higher populational density can have multiple fog nodes or even dedicated fog nodes for hospitals. Regardless of the granularity, the topology still respects VitalSense’s network topology.

3.2 Information naming service and data distribution

Data management in the IoT world is one of the many challenges fog computing solutions face [30]. Cloud-like systems traditionally employ centralized data storage solutions that require devices to transmit a high volume of data through the Internet (Fig. 7a). IoT devices can produce a massive amount of data sending it to the cloud at the cost of high latency. Health services that request this data also suffer from the same latency penalties. Fog computing addresses this drawback by allowing local data storing in fog nodes that are closer to the devices. Such a strategy comprises a distributed data storing solution in which data is scattered among several fog nodes (Fig. 7b). That makes data management complex since a particular device can move and change fog nodes on-the-fly. In such cases, fetching data from this particular device requires the system to lookup up its data in many locations.

Fig. 7 Data distribution strategies: **a** traditional cloud-like systems in which the cloud centralizes the data; and **b** fog strategy scattering the data among several nodes

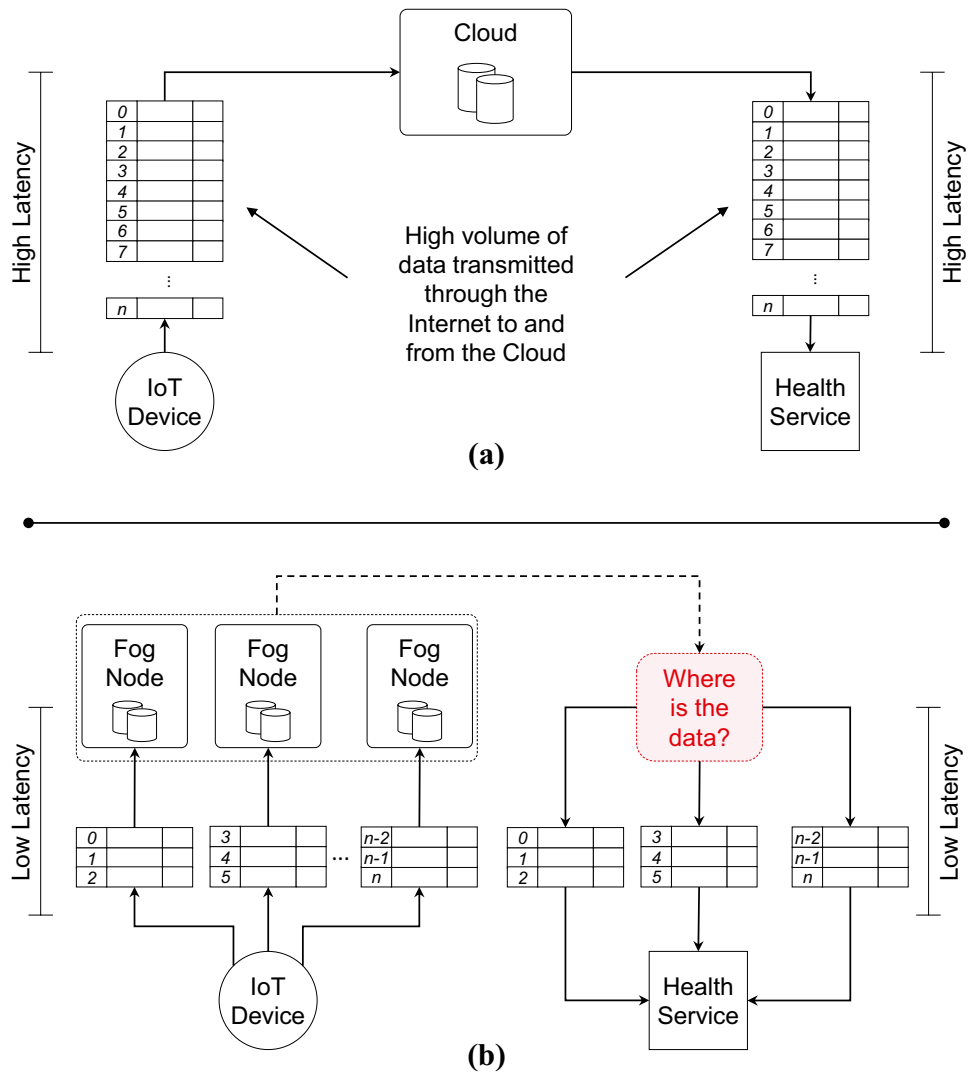
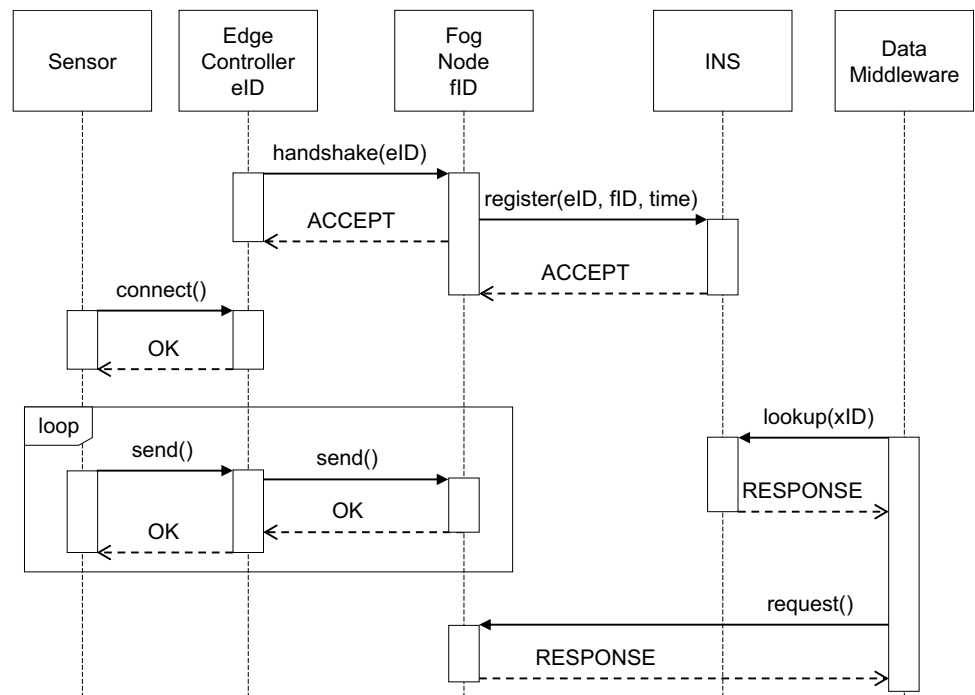


Fig. 8 Data storing and accessing sequence diagram using the INS to locate the data



VitalSense employs a data traceability strategy that offers two main features: user location tracking and faster data lookup. As users can move and connect to different fog nodes, their data can be scattered across several servers, imposing challenges to retrieve data from a given period. Therefore, we propose a new component called Information Naming Service (INS) placed in the cloud to maintain data track. The INS registers all handshake events edge-fog and handoff events fog-fog components perform. Therefore, it can track changes in the network topology and determine in which particular nodes data from a given device is stored. Each device has an Edge Controller registered in the INS. Edge Controllers have unique IDs which they inform fog nodes in the handshake process. Therefore, at each handshake and handoff process, fog nodes register in the INS the operation. The INS implements a naming system similar to the Internet DNS allowing queries given a particular ID. Figure 8 depicts a sequence diagram showing the operations that occur between Edge Controllers, fog nodes, and the INS.

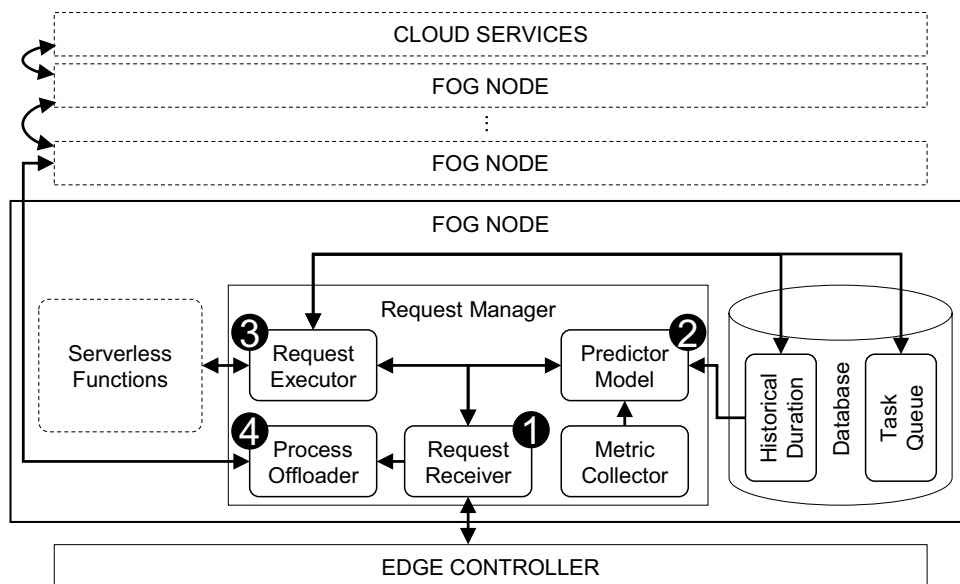
To access data from a device, the VitalSense queries the device ID in the INS. It can provide a specific time window to filter a target period. The INS provides the entries describing the device’s path and the time assigned for each fog node. Therefore, HealthStack can directly access the data in the fog nodes for further processing. For instance, we can employ sharding [31] techniques so HealthStack can access the node with the highest data volume or the closest node to fetches the data. When deploying a distributed data storage system, administrators should take several aspects into account [30]. Data sharding addresses several aspects, such

as fault tolerance, scalability, and data distribution. However, there are others one should also consider: security, privacy, data dissemination, and data replication.

3.3 Distributed service processing

VitalSense supports several services at the fog layer distributed among several nodes. The same service runs at different levels, enabling process offloading from one node to another. In other words, a particular fog node may offload a data sample to a different node to process the request. Data offloading is essential when a particular node is overwhelmed by too many requests that degrade its performance. Figure 9 presents the load distribution model performed by the Request Manager component of the fog nodes. The Request Receiver (1) decides where to process incoming requests according to user and service priorities and the prediction computed by a machine learning model in the Predictor Model (2). This particular module predicts how long the service execution will take to complete. Then, it delegates requests to the current node (3) in case it would not exceed the timeout or to the upper node (4) in case it would exceed. While the machine learning model is still learning services behavior, the prediction may not be completely accurate, especially in the first few executions. That may execute the request on the node but exceed the timeout for the first requests. Therefore, as we aim to ensure reliability if the execution exceeds the timeout, execution is re-executed on the upper node to benefit from unlimited processing capabilities. It is essential to mention that fog

Fig. 9 Request Manager responsible for processing and offloading requests: (1) decides where to process incoming requests based on predictions (2), and then it delegates requests to the current node (3) or the upper node (4)



nodes are heterogeneous. Each has its model, enabling them to learn their particular behavior.

After executing the request, the Request Executor stores its duration so the Predictor Model can use this information to predict how long future executions will take to finish. Recent duration samples have a higher impact on predictions. Request parameters and inputs should also be stored in the database, as duration may vary depending on the input. In case of resources on the node are overloaded, and connection with the upper node is inaccessible, the Request Executor adds incoming requests to a queue. The task queue ensures no request will be discarded and reliability for time-sensitive services. If local resources become available again, the Request Executor executes queued requests locally. When the upper node is accessible again, but local resources are still overloaded, the Process Offloader vertically offloads requests to it. Depending on the workload of upper nodes, all nodes may be overloaded, and the Request Managers recursively offload them until it reaches the cloud. In this case, the request is processed without a time limit, executing with unlimited processing capabilities provided by cloud vendors.

There are two types of priorities that the Request Receiver needs to consider when deciding which requests are the most important: (i) user priority; and (ii) service priority. Priorities impact the decision of the layer (closest fog node or parent fog nodes of the hierarchy) where the system should execute requests. The primary goal is to execute requests in the closest fog node whenever possible, resulting in low-latency responses because of the physical proximity between the edge devices and the node. All requests can be executed on the closest fog node when it has many computing resources available, regardless of the

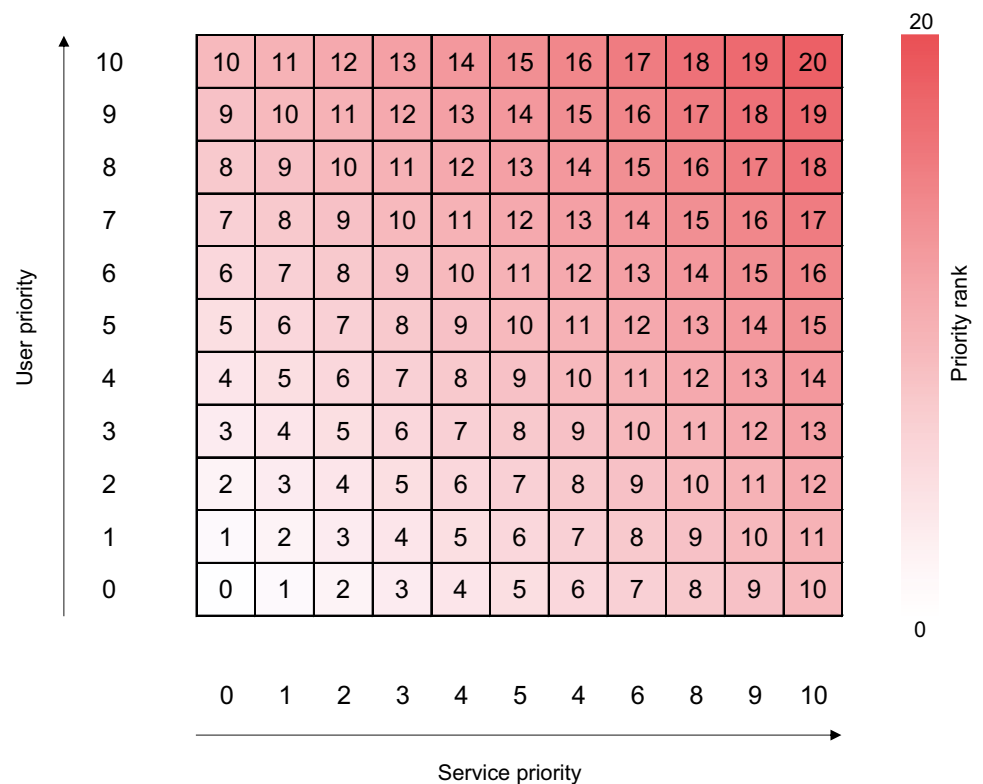
priority. On the other hand, low-priority requests should be offloaded to parent fog nodes when the platform is experiencing high usage peaks and when the closest fog node is overloaded.

User priority is essential when specific users should receive more attention than others, such as elderly people or groups in risky situations in the health field, that should have their requests executed with the lowest possible latency. The latter is related to the priority of the service itself, as specific services may be more critical than others. It is important to highlight that user priority is dynamic and is received on the request payload, as a person with health problems may become healthier after some time, and further requests should indicate lower priorities. The service priority is typically static and does not change over time unless the service is re-deployed or the priority is globally re-configured and, for this reason, is fetched from the database. The Request Receiver module is responsible for combining both information on a single numeric value. It combines both values and computes a rank value, as detailed in Fig. 10. Critical services with high user priority can be the first to run on the closest fog node. In contrast, requests for low-priority services with low-priority users are more feasible to execute on parent fog nodes like the cloud. In summary, the rule to combine both values is to sum both priorities, while the ranking is this final result.

4 Deployment and use cases

This section explores application use cases that can leverage VitalSense capabilities. In addition, we demonstrate how to deploy the architecture from a bottom-up point of view.

Fig. 10 Ranking combining both service and user priorities, with intense colors representing the most important requests



4.1 Architecture deployment instance

Figure 11 presents each architecture module’s deployment, including infrastructure aspects. The deployment base regards the IoT devices at the individual’s house. The target person wears a wrist device, such as a smartwatch, that monitors several vital signs. The Edge Collector can be deployed either in a single-board computer at the individual’s home or on his smartphone. Therefore, it can reach the smartwatch to extract data using either WiFi or Bluetooth. An Internet connection using a Modem/Router enables communication with the fog network from edge to fog. Virtual Private Network (VPN) and MPLS technologies can guarantee this connection’s privacy and performance, ensuring QoS in data transmission.

The network comprises several links among fog nodes distributed over the city at the fog. For instance, one can deploy a fog node server in public utility poles, network racks, shops, markets, hospitals, and others. The main requirement is that the network infrastructure for these fog nodes provides performance guarantees, using VPN and MPLS as an example. This infrastructure is the same when connecting the fog with the cloud. Therefore, the end of the infrastructure is the cloud data centers, which also connect to the Internet providing its endpoints to fog nodes. In addition, user applications can connect to the cloud Services using their public Internet connection likewise standard services and websites.

VitalSense supports the development of several services to run at all levels of the architecture. Different data types

are available depending on the level from which applications benefit. Applications may vary from dashboards to prediction algorithms to generate alerts. The following sections describe different use cases.

4.2 Risk exposure control

Figure 12 represents a risk exposure management system where the user can monitor the time of exposure to risk regions in a time frame. It consists of path monitoring that crosses location data with risk region classifications. Each zone can be classified depending on health parameters from users in each region. In addition, the public health system and the patients themselves can report to the system their current status. Machine learning techniques can classify each region by combining different data sources. Depending on the case, a high-risk region can mean a different thing. For instance, an infectious disease monitoring system can report exposure time to the disease and the potential for contraction. Another example is a hazard exposure system that can track and alert users to avoid crossing a specific area due to a gas leak or fire.

Besides providing alerts directly to users, such a system also provides valuable information to the public administrators. For instance, dashboards built on top of the information show the real-time status of several regions. In regions where risks are increasing, administrators can take proactive actions to mitigate potential threats to the population. In

Fig. 11 Example of a Vital-Sense’s architecture deployment in a smart city

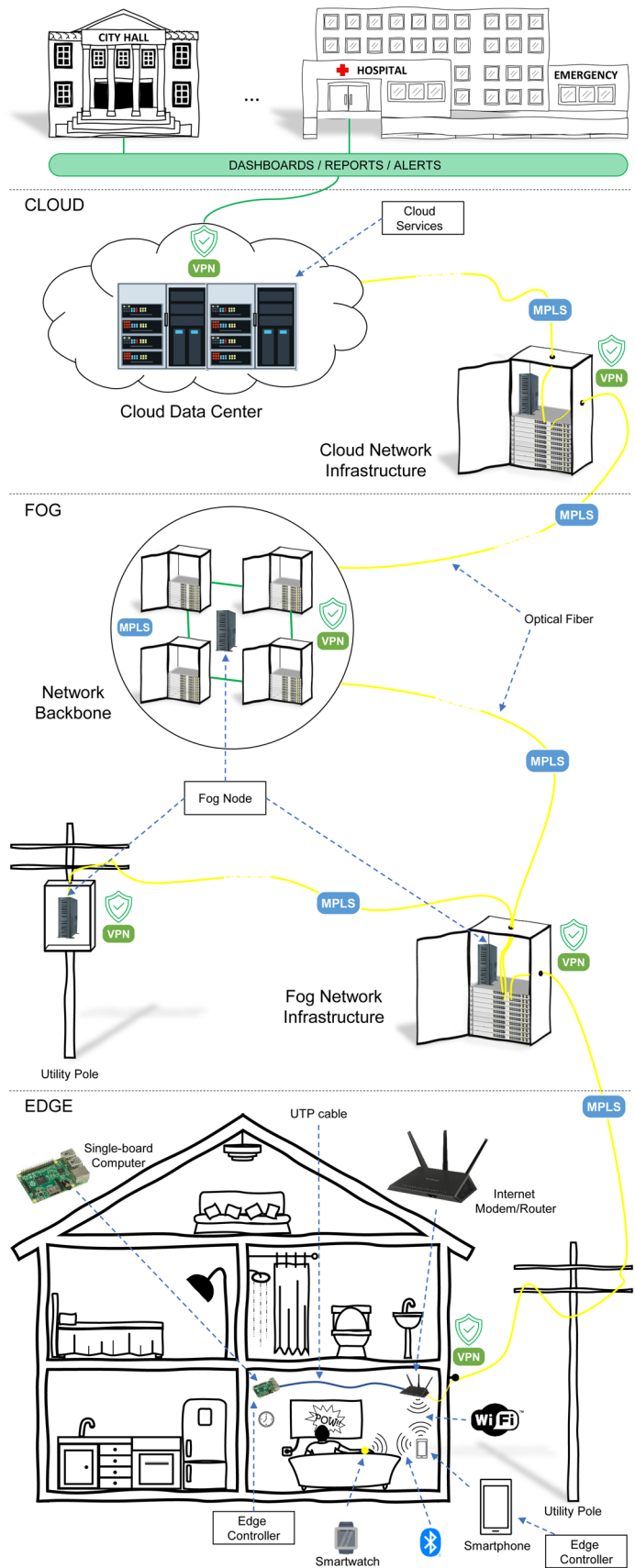
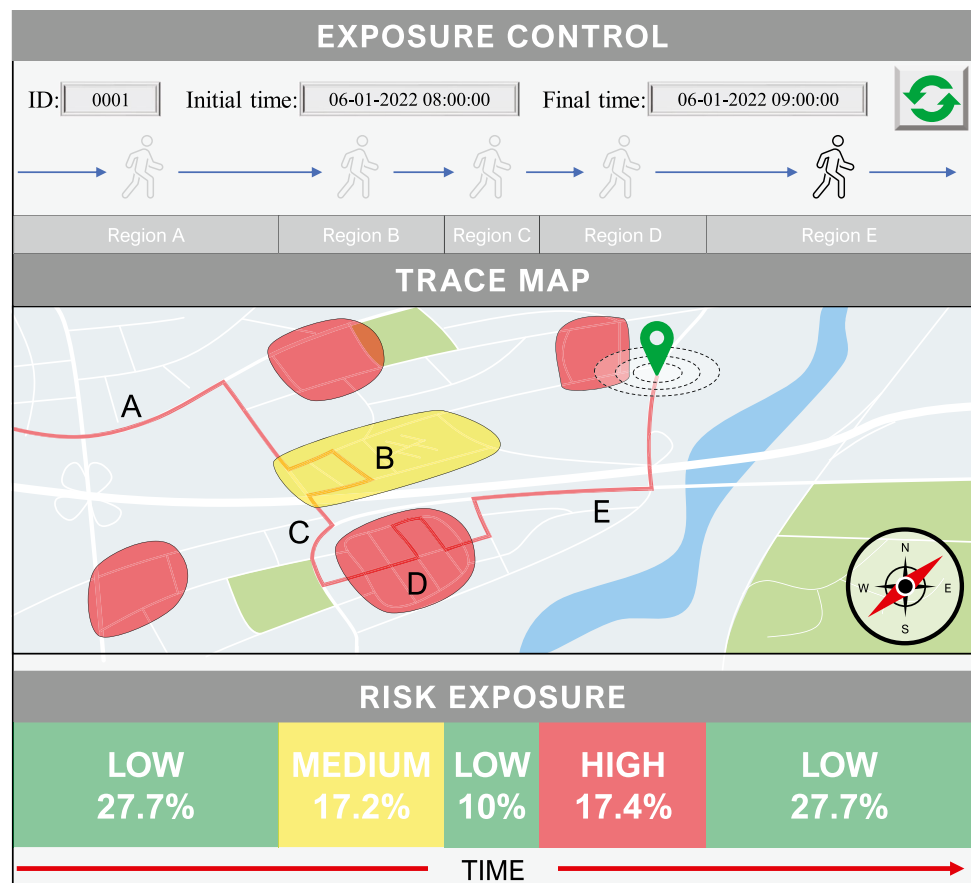


Fig. 12 Risk exposure control system

addition, they can generate public alerts to the population according to how the situation in specific regions degrades. When the system detects a leak of toxic solutions or airborne diseases, it can automatically isolate the area. At the same time, the public emergency department gets alerts asking for proactive aid. Being proactive in such scenarios can significantly improve health assistance services.

4.3 Proactive service delivery

Figure 13 demonstrates a proactive service delivery in which a hospital dispatches an ambulance when receiving a proactive alert from the fog. In this scenario, a user wears a heart monitoring sensor providing data to the system. The Edge Controller collects the data in real-time and forwards samples to the fog. A real-time analysis service processes the data performing predictions to identify possible cardiac failures. The fog notifies the public health system of an upcoming emergency. Thus, the closest available ambulance can answer the call and proactively attend to the patient in its own home.

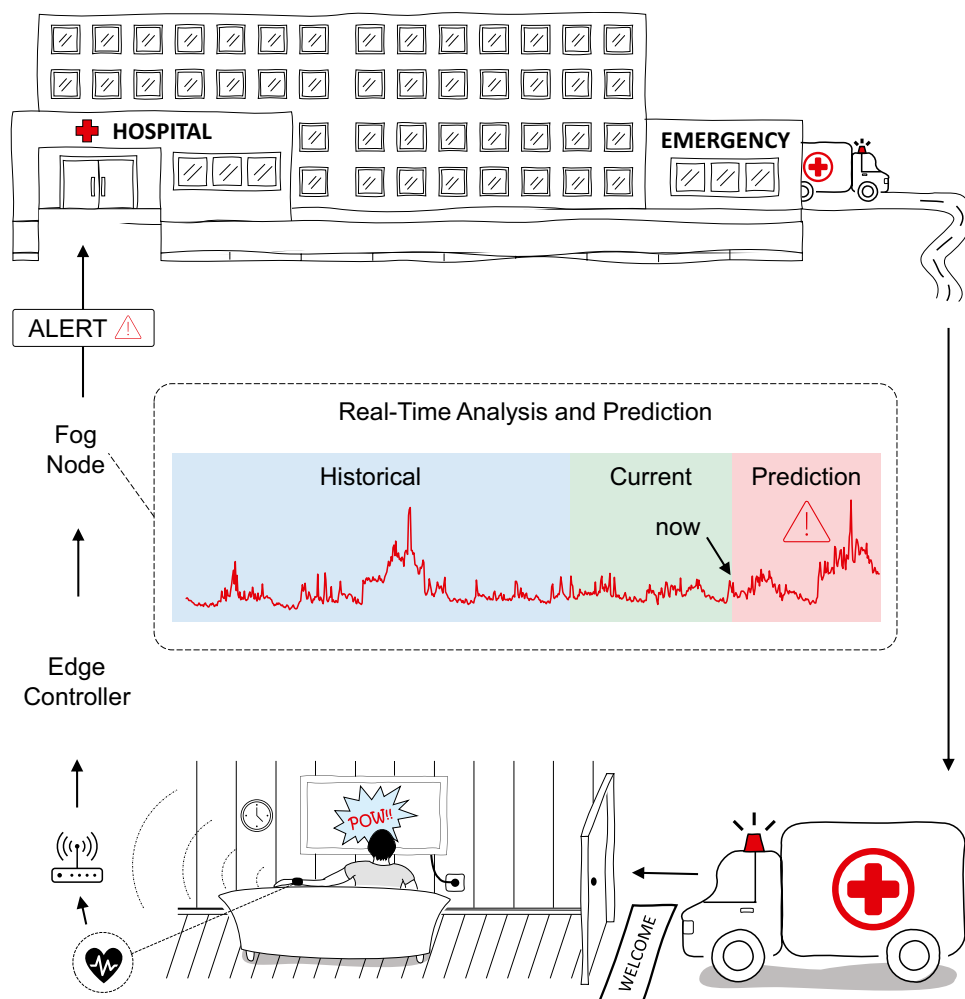
Although the figure presents a specific case for heart monitoring, such a service works for several data types. For instance, the same service works for fall detection of the elderly or leak detection of toxic substances. It can generate complex predictions for entire regions when combining data

from numerous sensors, including multiple locations. Multiple users reporting increasing fever and cough in a particular region can indicate a disease outbreak. The use of predictions in health services can mitigate public health crises or even prevent people from exposing themselves to dangerous situations. In addition to providing alerts to the public health system, the user can also receive notifications about hazards or health crises. Therefore, users can also proactively seek medical assistance in advance. Combined with the previous service, the user can receive real-time alerts when entering predicted risk regions.

4.4 Public health control dashboard

The presentation of clinical data summarized in a single panel or dashboard is increasingly in evidence. They are tools that use visual representations of data to allow a lower load of cognitive processing. Therefore, they increase the speed and accuracy of decision-making. Visual representations facilitate the identification of trends, patterns, or abnormalities in a given database compared to data analysis without a structure [32]. Figure 14 demonstrates what a dashboard would look like condensing several patient data into a single panel. Collecting sensor data makes it possible to monitor entire regions, including detecting the most

Fig. 13 Proactive ambulance attendance



critical regions and problems. Also, it is possible to present the current status of the region's vital signs and zoom in on micro-regions according to a specific parameter. Although the panel presents data from several individuals, it is possible to navigate data from specific patients, for example. Such dashboards may be available at different levels, including for the health provider or the government with data aggregation.

The design used for the visual resources in a dashboard must seek a balance between visual complexity and the usefulness of the information. Essentially, dashboards should be tailored to the intended purpose and the end-user, considering their analytical skills and level of knowledge. The patient's perspective needs to be repeatedly emphasized. The presentation of data in graphs should be the choice when comparing, grouping, or recognizing patterns in the information [33]. In other situations, tables should be the choice when there is a need to evaluate specific values [33]. Due to the complexity, one should avoid the presentation of information in formulas and fractions. Other details, such as the presentation of specific values with the same number of decimal places and the same scale on their axes of the graphs,

must also be observed to avoid biases in the interpretation of the information displayed.

Using color schemes helps draw attention to results that indicate an area of concern, such as increased body temperature or decreased blood oxygenation. The colors green or blue, yellow or orange, and red should preferably be chosen to sequentially indicate the risk or severity of a physiological abnormality [34]. Gray color alerts can indicate system problems during the recovery of elements that make up the information to be visualized. Noteworthy, clinical indicators of a health condition must be based on scientific evidence and be recognized in national guidelines [35]. In addition to the different indicative colors, the system can notify the user with alert services to indicate that certain data is outside predetermined limits. Alert services can be designed to generate links to help the user make a decision, such as providing the option to call an emergency service or a specialized service, send a message or email to the responsible doctor with details of the condition, or even to seek information about why the alert was issued.

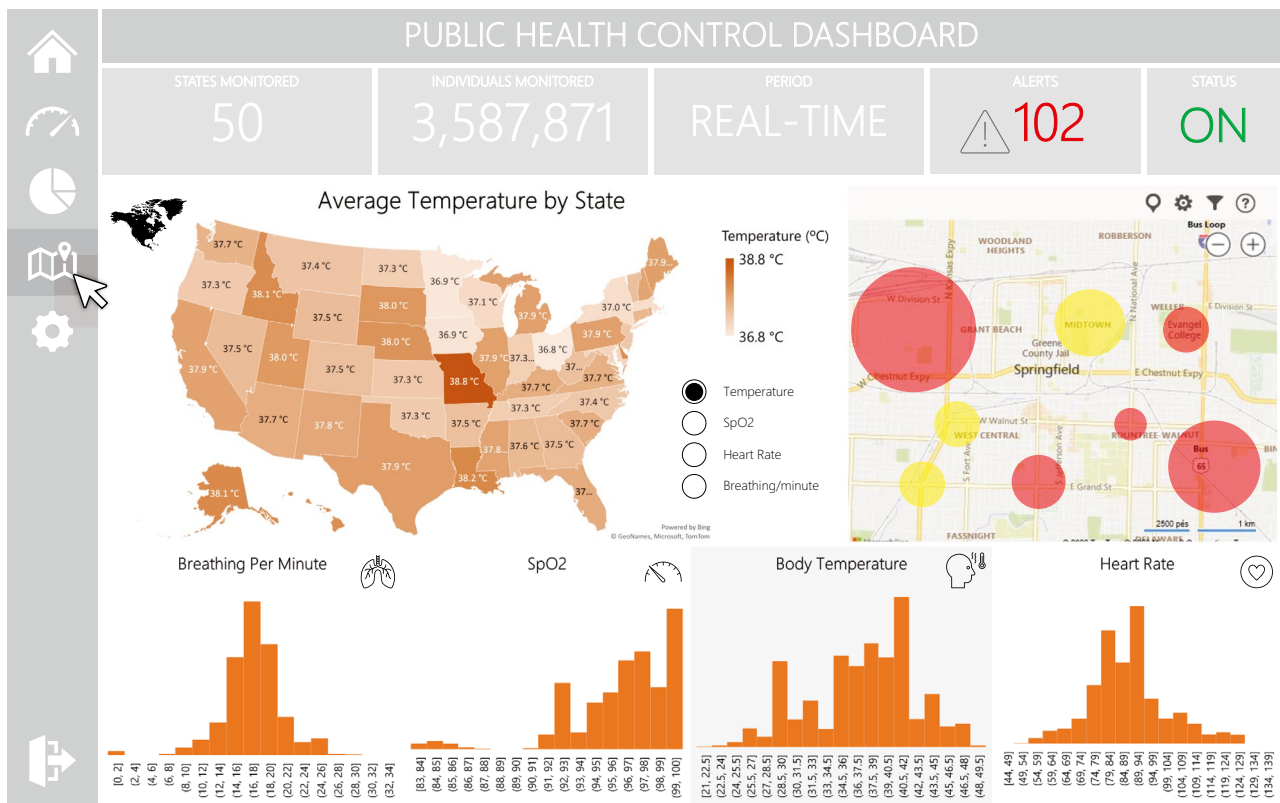


Fig. 14 Dashboard condensing several patient data into a single panel by region in real-time. The map allows zoom in and out of specific regions

5 Discussion and preliminary experiments

VitalSense is a solution developed as part of two research projects: MinhaHistoriaDigital (MyDigitalHistory) and MinhaSaudeDigital (MyDigitalHealth), which focus on public health issues in Brazil. The main goal of the current research is to develop and deploy the infrastructure in the city of Porto Alegre, Rio Grande do Sul, Brazil. This section introduces the initial experiments and considerations regarding the development and deployment of VitalSense. Section 5.1 presents preliminary experiments using Sharding to test the performance of the data distribution strategy. Following, Section 5.2 introduces the primary IoT devices available in the market to monitor vital signs.

5.1 Data distribution efficiency using sharding

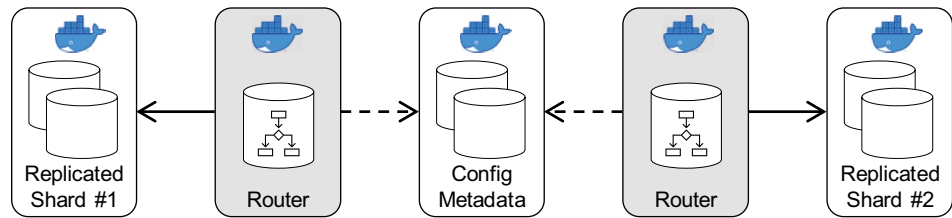
We performed preliminary experiments to validate the model's data distribution efficiency using Sharding in a Docker-based infrastructure to individualize processing on a single computer. The host configuration consists of an Intel Core i7 Hexa-core 2.6 GHz, 16GB of DDR4 Dual-Channel 2667 MHz RAM, and an SSD with a read and write rate of 3200 MB/s and 2200 MB/sec, respectively. We employed MongoDB as a database that implements Sharding

straightforwardly and efficiently. As shown in Fig. 15, as it is a requirement to use sharding in MongoDB, we added a configuration server to the model that stores the metadata of the location of records in the structure.

We established two preliminary scenarios to validate the model's efficiency using Sharding (with two nodes) compared to a centralized database. The first scenario performs a search using a filter by the server in the query. In the second, the query was performed without filtering, returning the entire database. Both scenarios consider a database with over 2 million records from which we retrieve a sample of data in 10 different executions. Figure 16 presents the results for both scenarios.

Preliminary results show that employing Sharding leads to an improvement of approximately 20% in the query response time. The proximity factor with the data is not applicable in the experiments since the instances run in the same machine; otherwise, it would bring even more advantages to Sharding in a real distributed scenario. With Sharding, results show an average response time of 1.36 and 1.38 s (with a standard deviation of 0.1350 and 0.0919 s) in the first and second scenarios. Without Sharding, the response time is 1.66 and 1.78 s (with a standard deviation of 0.1430 and 0.1751 s) for the first and second scenarios. In the experiments with Sharding, the search time of all records in

Fig. 15 Evaluation methodology using two Sharding instances



both nodes or a single node is practically the same. Due to the data distribution among several servers, the query runs concurrently, distributing the workload.

The preliminary experiments show the advantages of distributing data among several fog nodes instead of centralizing them in the cloud. In addition to improving response time when querying data, it also enhances the performance of the whole network by not forwarding data from all sensors up to the cloud. Furthermore, applications can request data from the Fog infrastructure and benefit from faster data transmission since the Fog is closer to the application than the cloud.

5.2 Directions on vital sign monitoring

Currently, we are witnessing a rise in IoT solutions for self-health monitoring. Each day new devices are available in the market so everyone can buy and monitor their vital signs using smartphone applications. IoT devices constitute the basis of vital sign remote monitoring systems, providing health data from individuals in real-time. The lack of this information can compromise the feasibility of the solutions since they depend on patient health data extraction. Nowadays, several solutions propose specific monitoring devices that are slowly becoming off-the-shelf products available for everyone. Although numerous devices are available, they are not entirely tailored to support a massive monitoring system. Their primary focus is to provide information to whom is wearing the device through proprietary smartphone applications. In addition, most devices equip only a few sensors,

which may require patients to wear more than one device for remote health monitoring.

Wearing several devices is acceptable when the individual is already a patient who can stay at home for remote monitoring after already identifying a medical condition. However, when focusing on proactive health monitoring, it is not feasible for individuals to wear several devices during daily activities. Daily health monitoring should be seamless from the patient's point of view. It should not impose discomfort and restrictions on individuals' activities. That is only possible through devices equipped with several sensors at once. Therefore, individuals can only wear a unique device, such as a smartwatch or smart band. Thus, this device can support several communication protocols to provide data from several sensors in a standard format.

To assess the availability of such devices, we performed research in the market looking for candidate personal health monitoring devices to integrate the VitalSense model. We evaluated 149 devices, including smartwatches, smart bands, and smart rings, analyzing the sensors they equip, security features, API availability, and communication protocols. We ruled out all devices with only one or two sensors since we focus on devices that support a broad set of sensors. Table 2 compares 40 resulting smart IoT devices that equip several vital sign sensors. The table shows that heart rate and SpO2 sensors are the most common among the devices. Besides, only seven devices provide all vital signs sensors, representing 4.7% of the entire set of 149 devices. Although several devices are available, just a few support a complete set of sensors, demonstrating a lack of devices for vital sign remote monitoring solutions. Future research should focus intensely on this topic since it can limit the potential of vital sign remote health monitoring solutions.

VitalSense works with edge, fog, and cloud computing, bringing an idea of districts where fog nodes operate. In other words, our architecture was first planned to cover urban areas. However, we can extend this coverage area to rural areas by adopting technological requirements. In our understanding, power supply and Internet communication are the most expressive problems in agricultural areas. Some strategies could be taken to aggregate them, such as: (i) we can use battery-based IoT wearables devices to collect vital signs; (ii) we can use long-range radio frequency technologies. For both cases, LoRa and LoRaWAN appear as one of the best initiatives to connect rural areas. LoRa and LoRaWAN

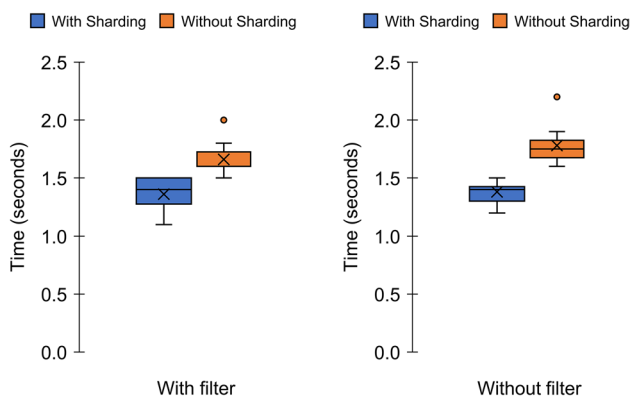


Fig. 16 Response time when querying data from the database with and without Sharding

Table 2 Devices details: Respiratory Rate (RR); Heart Rate (HR); Heart Rate Variability (HRV); Body temperature (BT); and Oxygen Saturation (SpO2)

Device	Vital Signs					Security		API	Communication
	RR	HR	HRV	BT	SpO2	Crypt.	Auth.		
Amazfit GTR 2e		✓		✓	✓		✓	✓	BLE, Bluetooth 5.0
Amazfit GTR 3 Pro	✓	✓		✓	✓		✓	✓	Bluetooth 5.0, WiFi
Amazfit GTS 2e		✓		✓	✓		✓	✓	BLE, Bluetooth 5.0
Amazfit GTS 3		✓		✓	✓		✓	✓	BLE, Bluetooth 5.0
Amazfit TRex Pro		✓	✓		✓		✓	✓	BLE, Bluetooth 5.0
Biostrap EVO Wristband	✓	✓	✓		✓		✓		Bluetooth
Colmi Land 2		✓		✓	✓				Bluetooth
Colmi Land 2 S		✓		✓	✓				Bluetooth
Colmi P8		✓		✓	✓				Bluetooth
Colmi V23 Pro		✓		✓	✓				Bluetooth
Empatica E4 wristband		✓	✓	✓		✓	✓	✓	BLE, USB 2.0
Empatica Embrace PLUS	✓	✓	✓	✓	✓	✓	✓	✓	BLE, USB 2.0
Fitbit Inspire 2	✓	✓	✓	✓	✓	✓	✓	✓	BLE, NFC
Fitbit Ionic	✓	✓	✓	✓	✓	✓	✓	✓	BLE
Fitbit Sense	✓	✓	✓			✓	✓	✓	BLE
Fitbit Versa 2	✓	✓	✓	✓	✓	✓	✓	✓	BLE, NFC, WiFi
Fitbit Versa 3	✓	✓	✓	✓	✓	✓	✓	✓	BLE, NFC, WiFi
Fitbit Charge 5	✓	✓	✓	✓	✓	✓	✓	✓	BLE, NFC, WiFi
Fitbit Luxe	✓	✓	✓	✓	✓	✓	✓	✓	BLE, NFC, WiFi
Garmin fênix 6	✓	✓	✓	✓	✓	✓	✓	✓	ANT+, Bluetooth, WiFi
Garmin fênix 6 Pro	✓	✓	✓	✓	✓	✓	✓	✓	ANT+, Bluetooth
Garmin fênix 6 S	✓	✓	✓	✓	✓	✓	✓	✓	ANT+, Bluetooth, WiFi
Garmin fênix 6 S Pro	✓	✓	✓	✓	✓	✓	✓	✓	ANT+, Bluetooth
Garmin fênix 6X Pro	✓	✓	✓	✓	✓	✓	✓	✓	ANT+, Bluetooth, WiFi
Garmin Forerunner 245	✓	✓	✓		✓	✓	✓	✓	ANT+, Bluetooth, WiFi
Garmin Forerunner 55		✓	✓		✓	✓	✓	✓	ANT+, Bluetooth
Garmin Forerunner 745	✓	✓	✓			✓	✓	✓	ANT+, Bluetooth
Garmin Forerunner 945	✓	✓	✓		✓	✓	✓	✓	ANT+, Bluetooth, WiFi
Garmin Forerunner 945 LTE	✓	✓	✓		✓	✓	✓	✓	ANT+, Bluetooth, WiFi
Garmin Instinct	✓	✓	✓		✓	✓	✓	✓	ANT+, Bluetooth, WiFi, LTE
Garmin Instinct Solar		✓	✓			✓	✓	✓	ANT+, Bluetooth
Garmin Venu 2		✓	✓		✓	✓	✓	✓	ANT+, Bluetooth
Garmin Venu Sq	✓	✓	✓		✓	✓	✓	✓	ANT+, Bluetooth, WiFi
Garmin Vivoactive 4	✓	✓	✓		✓	✓	✓	✓	ANT+, Bluetooth
Huawei Watch 3		✓		✓	✓			✓	BLE, Bluetooth 5.2, LTE, NFC, WiFi
Huawei Watch 3 Pro		✓		✓	✓			✓	BLE, Bluetooth 5.2, LTE, NFC, WiFi
Oura Oura Ring	✓	✓	✓	✓		✓	✓	✓	BLE
Xunjia Tech E66 Smart Bracelet		✓		✓	✓				BLE 4.0
Xunjia Tech HW12 Smartwatch		✓		✓	✓				Bluetooth 5.2
Xunjia Tech W26 Smartwatch		✓		✓	✓				Bluetooth

define a Low Power Wide Area (LPWA) networking protocol designed to wirelessly connect battery-operated devices to the Internet in regional, national, or global networks. Moreover, this duet targets critical IoT requirements such as bi-directional communication, end-to-end security, mobility, and localization services. Considering the device scope, we

have a battery duration of up to 10 years today. Considering the network viewpoint, the LoRaWAN data rate ranges from 0.3 kbit/s to 50 kbit/s per channel. This bandwidth is entirely acceptable for transferring vital sign data of a family. For Example, Fig. 17 illustrates JSON data with a person’s vital signs where the whole message has less than 200 bytes.

Fig. 17 Example of a JSON data packet representing vital sign collection of a person

```
{
  "service_name": "heart-failure-predictor",
  "user_priority": 5,
  "start_timestamp": 1663527788128,
  "vital_sign": "{\"heartbeat\": 97,\"spo2\": 99,\"temperature\": 40}"
}
```

6 Final remarks

We envisage that AI, edge/fog, geolocation-based sharding, resource elasticity, and data compression will support the new vision of smart health cities. VitalSense aims to provide benefits for the following group of users: (i) citizens, who will have their data monitored and will receive alerts regarding possible problems in their vital signs, as well as alerts regarding the spread of diseases in places in the city and, in particular, those frequently visited by the individual in question; (ii) public managers, who will have a real-time macro view of specific and historical population data, being able to direct public policies for certain groups of people or districts of the city; (iii) hospital managers with access to municipal data regarding the management and prediction of the spread of COVID-19 can allocate teams and request public funds in advance to increase the chances that human lives are saved. The model introduces several contributions spread throughout the document:

- Employment of data compression and homomorphic encryption at the edge (Sections 3.1.1 and 3.1.2);
- A multi-tier notification mechanism (Sections 3.1.1, 3.1.2, and 3.1.3);
- Low latency health traceability with data sharding (Section 3.2);
- Serverless engine, container elasticity, and process off-loading mechanism (Section 3.3);
- Live geographical-based dashboards for data visualization (Section 3.1.3).

In addition to the aforementioned technical contributions, it is essential to outline some social contributions and how the authors see the healthcare future. We envisage a collection of AI-based latency-oriented services seemingly incorporated into the citizens' daily lives, bringing a novel set of notifications, dashboards, and integration among people, communities, hospitals, and the public sector. In this scope, VitalSense is a deployment possibility to enable digital healthcare monitoring truly. Below we present some healthcare scenarios in which we can live in the short and mid-term:

- Analyzing a person's vital signs, a hospital or clinic can send a vehicle to the house of a possible patient, as it is predicted that they are feeling unwell and may die soon;

- I want to go to the supermarket, and I am trying to decide which one to go to and which would be safer. I can see the citizens tracking information, people who have/had COVID-19, and if they have frequented that establishment;
- Analyzing vital signs data from different city neighborhoods and seeing that the situation is worsening with event prediction, a hospital (or more than one) can prepare human resources and physical infrastructure beforehand to receive new patients;
- We can analyze which type of disease occurs more frequently in a given scenario. In detail, we can explore trends and disease seasonality to get insights like this: the older the age group, the higher the incidence of this particular disease;
- It is also possible to apply public health policies personalized by districts and neighborhoods, given that there is a higher incidence of a certain disease.
- Using people's tracking data, the public sector can check whether the lockdown works;
- By getting data on a smart city scale, we can identify areas with worsening mental health of post-COVID-19 patients, such as anxiety and depression crises due to the loss of family members. Thus, the government can assist them by indicating therapeutic follow-up/medical consultation;
- Citizens can be alerted about personal or family health problems and receive information about their community.

VitalSense is a solution in development as part of two projects: MinhaHistoriaDigital (MyDigitalHistory) and MinhaSaúdeDigital (MyDigitalHealth). They focus on the public health area in Brazil and are supported by development agencies in that country. Privacy is a primary concern in such a scenario, so the VitalSense hash strategy is based on the GDPR. We are basing the model's decisions on the regulation and are currently working on this front. Future work comprises the end-to-end deployment of the three levels. Although we have preliminary results, to the best of our knowledge, we are the first to explore internal novelties in the layers and think about interconnections between them.

Acknowledgements The authors would like to thank the following Brazilian Agencies for supporting this work: the Research Support Foundation of the Rio Grande do Sul State (FAPERGS, processes 19/2551-0001340-0 and 21/2551-0000118-6); Coordination for the

Improvement of Higher Education Personnel (CAPES, financial code 001); National Council for Scientific and Technological Development (CNPq, grant numbers 309537/2020–7 and 305263/2021–8).

Author contributions All authors contributed to the study conception and design. In particular, Rodrigo Righi and Vinicius Facco worked in the VitalSense technologies. Cristiano da Costa and Bjoern Eskofier focused on reviewing the Related Work section, also detailing the open gaps in the literature. Daeyoung Kim revised the modules composition of the Edge, Fog and Cloud architectures. Andreas Mayer focused on analyzing information in Section 4, where we present the Deployment and Use Cases. All authors contributed to final modifications in the article, including more details in

Funding This work was supported by the following Brazilian agencies: the Research Support Foundation of the Rio Grande do Sul State (FAPERGS, processes 19/2551–0001340-0 and 21/2551–0000118-6); Coordination for the Improvement of Higher Education Personnel (CAPES, financial code 001); National Council for Scientific and Technological Development (CNPq, grant numbers 309537/2020–7 and 305263/2021–8).

Data availability Not Applicable.

Code availability Not Applicable.

Declarations

Competing interests Not Applicable.

References

- Philip NY, Rodrigues JJPC, Wang H, Fong SJ, Chen J. Internet of things for in-home health monitoring systems: Current advances, challenges and future directions. *IEEE J Sel Areas Commun.* 2021;39(2):300–10. <https://doi.org/10.1109/JSAC.2020.3042421>.
- Rekha G, Yashaswini J. In: Tyagi, A.K., Abraham, A., Kaklauskas, A. (eds.) *Industry 4.0: A Revolution in Healthcare Sector via Cloud, Fog Technologies*; 2022. pp. 321–335. Springer, Singapore. https://doi.org/10.1007/978-981-16-6542-4_16.
- Tortorella GL, Fogliatto FS, Mac Cawley Vergara A, Vassolo R, Sawhney R. *Healthcare 4.0: trends, challenges and research directions.* *Prod Plan Control.* 2020;31(15), 1245–1260.
- da Costa CA, Pasluosta CF, Eskofier B, da Silva DB, da Rosa Righi R. Internet of health things: Toward intelligent vital signs monitoring in hospital wards. *Artif Intell Med.* 2018;89:61–9. <https://doi.org/10.1016/j.artmed.2018.05.005>.
- Shukla S, Hassan MF, Tran DC, Akbar R, Papatungan IV, Khan MK. Improving latency in Internet-of-Things and cloud computing for real-time data transmission: a systematic literature review (SLR). *Clust Comput.* 2021;3. <https://doi.org/10.1007/s10586-021-03279-3>.
- Buschmann P, Shorim MHM, Helm M, Bröring A, Carle G. Task allocation in industrial edge networks with particle swarm optimization and deep reinforcement learning. In: *Proceedings of the 12th International Conference on the Internet of Things. IoT 2022*, pp. 239–247. Association for Computing Machinery, New York, NY, USA (2023). <https://doi.org/10.1145/3567445.3571114>.
- Rodrigues VF, Righi RdR, da Costa CA, Antunes RS. Smart hospitals and iot sensors: Why is qos essential here? *J Sens Actuator Netw.* 2022;11(3). <https://doi.org/10.3390/jsan11030033>.
- Aazam M, Zeadally S, Harras KA. Fog computing architecture, evaluation, and future research directions. *IEEE Commun Mag.* 2018;56(5):46–52. <https://doi.org/10.1109/MCOM.2018.1700707>.
- Tuli S. Ai and co-simulation driven resource management in fog computing environments. *SIGMETRICS Perform Eval Rev.* 2023;50(3):16–9. <https://doi.org/10.1145/3579342.3579347>.
- Parimala Devi M, Raja GB, Gowrishankar V, Sathya T. In: Chakraborty C, Banerjee A, Garg L, Rodrigues JJPC. (eds.) *IoT-Based Smart Diagnostic/Therapeutic Kit for Pandemic Patients*; 2020. pp. 141–165. Springer, Singapore. https://doi.org/10.1007/978-981-15-8097-0_6.
- Dong Y, Yao Y-D. Iot platform for covid-19 prevention and control: A survey. *IEEE Access.* 2021;9:49929–41. <https://doi.org/10.1109/ACCESS.2021.3068276>.
- Sahu ML, Atulkar M, Ahirwal MK, Ahamad A. Vital sign monitoring system for healthcare through iot based personal service application. *Wirel Pers Commun.* 2022;122(1):129–56. <https://doi.org/10.1007/s11277-021-08892-4>.
- Rahman MJ, Morshed BI, Harmon B, Rahman M. A pilot study towards a smart-health framework to collect and analyze biomarkers with low-cost and flexible wearables. *Smart Health.* 2022;23:100249. <https://doi.org/10.1016/j.smhl.2021.100249>.
- Li F, Valero M, Shahriar H, Khan RA, Ahamed SI. Wi-covid: A covid-19 symptom detection and patient monitoring framework using wifi. *Smart Health.* 2021;19:100147. <https://doi.org/10.1016/j.smhl.2020.100147>.
- Al Bassam N, Hussain SA, Al Qaraghuli A, Khan J, Sumesh EP, Lavanya V. Iot based wearable device to monitor the signs of quarantined remote patients of covid-19. *Informatics in Medicine Unlocked.* 2021;24: 100588. <https://doi.org/10.1016/j.imu.2021.100588>.
- Paganelli AI, Velmovitsky PE, Miranda P, Branco A, Alencar P, Cowan D, Endler M, Morita PP. A conceptual iot-based early-warning architecture for remote monitoring of covid-19 patients in wards and at home. *Internet of Things.* 2021;100399. <https://doi.org/10.1016/j.iot.2021.100399>.
- Mukherjee A, Ghosh S, Behere A, Ghosh SK, Buyya R. Internet of health things (ioht) for personalized health care using integrated edge-fog-cloud network. *J Ambient Intell Humaniz Comput.* 2021;12:943–59.
- Chudhary R, Sharma S. Fog-cloud assisted framework for heterogeneous internet of healthcare things. *Procedia Computer Science.* 2021;184, 194–201. <https://doi.org/10.1016/j.procs.2021.03.030>. The 12th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 4th International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops.
- Sangeetha D, Rathnam MV, Vignesh R, Chaitanya JS, Vaidehi V. In: Vijayakumar, V., Neelanarayanan, V., Rao, P., Light, J. (eds.) *MEDIDRONE—A Predictive Analytics-Based Smart Healthcare System*; 2020. pp. 19–33. Springer, Singapore. https://doi.org/10.1007/978-981-32-9889-7_2.
- Priambodo R, Kadarina TM. Monitoring self-isolation patient of covid-19 with internet of things. In: *2020 IEEE International Conference on Communication, Networks and Satellite (Comnetsat)*; 2020. pp. 87–91. <https://doi.org/10.1109/Comnetsat50391.2020.9328953>.
- Aziz K, Tarapiah S, Ismail SH, Atalla S. Smart real-time healthcare monitoring and tracking system using gsm/gps technologies. In: *2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC)*; 2016. pp. 1–7. <https://doi.org/10.1109/ICBDSC.2016.7460394>.
- Sicari S, Rizzardi A, Coen-Portisini A. Home quarantine patient monitoring in the era of covid-19 disease. *Smart Health.* 2022;23:100222. <https://doi.org/10.1016/j.smhl.2021.100222>.
- Rahmani AM, Gia TN, Negash B, Anzanpour A, Azimi I, Jiang M, Liljeberg P. Exploiting smart e-health gateways at the edge of healthcare internet-of-things: A fog computing approach. *Futur Gener Comput Syst.* 2018;78:641–58. <https://doi.org/10.1016/j.future.2017.02.014>.
- Kamel Boulos MN, Berry G. Real-time locating systems (rtls) in healthcare: a condensed primer. *Int J Health Geogr.* 2012;11(1):1–8.

25. Xu X, Lu Y, Vogel-Heuser B, Wang L. Industry 4.0 and industry 5.0-inception, conception and perception. *J Manuf Syst.* 2021; 61:530–5.
26. Chabi Sika Boni AK, Hablatou Y, Hassan H, Drira K. Distributed deep reinforcement learning architecture for task offloading in autonomous iot systems. In: *Proceedings of the 12th International Conference on the Internet of Things. IoT 2022; 2023.* pp. 112–118. Association for Computing Machinery, New York, NY, USA. <https://doi.org/10.1145/3567445.3567454>.
27. Xiao X, Hannan A, Bailey B, Ni LM. Traffic engineering with mpls in the internet. *IEEE Netw.* 2000;14(2):28–33. <https://doi.org/10.1109/65.826369>.
28. Yang Z, Cui Y, Li B, Liu Y, Xu Y. Software-defined wide area network (sd-wan): Architecture, advances and opportunities. In: *2019 28th International Conference on Computer Communication and Networks (ICCCN); 2019.* pp. 1–9. <https://doi.org/10.1109/ICCCN.2019.8847124>.
29. Alloghani M, Alani MM, Al-Jumeily D, Baker T, Mustafina J, Hussain A, Aljaaf AJ. A systematic review on the status and progress of homomorphic encryption technologies. *J Inf Secur Appl.* 2019;48:102362. <https://doi.org/10.1016/j.jisa.2019.102362>.
30. Moysiadis V, Sarigiannidis P, Moscholios I. Towards distributed data management in fog computing. *Wirel Commun Mob Comput.* 2018;2018.
31. Abdelhafiz BM, Elhadef M. Sharding database for fault tolerance and scalability of data. In: *2021 2nd International Conference on Computation, Automation and Knowledge Management (ICCAKM); 2021.* pp. 17–24. <https://doi.org/10.1109/ICCAKM50778.2021.9357711>.
32. Rouhani S, Zamenian S. An architectural framework for health-care dashboards design. *J Healthc Eng.* 2021;2021, 1964054. <https://doi.org/10.1155/2021/1964054>.
33. Simpson SH. Creating a data analysis plan: What to consider when choosing statistics for a study. *Can J Hosp Pharm.* 2015;68(4):311–7. <https://doi.org/10.4212/cjhp.v68i4.1471>.
34. Caroline Gonçalves P, Pinto Júnior D, de Oliveira Salgado P, Machado Chianca TC. Relationship between risk stratification, mortality and length of stay in a emergency hospital. *Invest Educ Enferm.* 2015;33(3):424–431. <https://doi.org/10.17533/udea.iee.v33n3a05>.
35. Ghazisaeidi M, Safdari R, Torabi M, Mirzaee M, Farzi J, Goodini A. Development of performance dashboards in healthcare sector: Key practical issues. *Acta informatica medica : AIM : journal of the Society for Medical Informatics of Bosnia & Herzegovina : casopis Društva za medicinsku informatiku BiH.* 2015;23(5):317–21. <https://doi.org/10.5455/aim.2015.23.317-321>.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.