Edge Computing Smart Healthcare Cooperative Architecture for COVID-19 Medical Facilities

Mateus C. Silva, Andrea G. C. Bianchi, Servio P. Ribeiro, Jorge Sá Silva, *Senior Member, IEEE* and Ricardo A. R. Oliveira

***Abstract*—Intelligent healthcare systems are a topic of interest in recent approaches due to novel possibilities created from edge hardware and software development. In 2020, the COVID-19 pandemic displayed the urge to speed up technological systems development to aid medical facilities. In this context, solutions must enhance the experience of both patients and healthcare professionals. Thus, we propose a novel cooperative architecture to improve healthcare facilities involved in pandemic control. On the one hand, this solution helps a faster recognition and link to the patients' data using reality augmentation resources. On the other hand, it helps monitor the conditions of medical professionals working in the facility and exposed to contamination danger.**

***Index Terms*—smart healthcare, cooperative systems, edge computing, IoT.**

I. Introduction

The usage of novel technologies for smart healthcare employs state-of-the-art edge computing, and IoT perspectives [1]. For instance, these systems can be employed to monitor patients [1] and detecting disease symptoms [2]. Furthermore, it is important to develop these cost-restrictive systems that are also quick to deploy, as in many times, the facilities are emergency field hospitals [3]. Hence, these tools can manage both workers and patients in an interconnected environment.

Kliger and Silberzweig [4] state that the COVID-19 is a disease caused by a novel coronavirus. These authors state that there is also an urge to mitigate in-hospital transmission. A relevant aspect in the hospital spaces [5], requiring the management of patients, personnel, space, and supplies. The healthcare professionals' exposure to contamination is also an essential factor in the control of the coronavirus pandemic [6].

Some researchers recommend the use of oximetry to monitor rapid patient deterioration. Shah et al. [7] evaluate patient-reported oxygen saturation (SpO2) using pulse oximetry as a home monitoring tool for patients with initially non-severe COVID-19 to identify the need for hospitalization. Oxygen levels best identify patients most at risk of poor outcomes. Therefore, applying an oximeter-based detector of the decrease in blood oxygen permits the creation of alert systems to act before a severe health problem. In this work, we propose using continuous monitoring of the healthcare workers groups in action.

Mateus C. Silva, Andrea G. C. Bianchi, and Ricardo A. R. Oliveira are with the Computing Department, Federal University of Ouro Preto, e-mail:mateuscoelho.ccom@gmail.com, andrea@ufop.edu.br, rrabelo@gmail.com.

Servio P. Ribeiro is with the Computing Department, Federal University of Ouro Preto, e-mail:spribeiro@gmail.com.

Jorge Sá Silva is with Department of Electrical and Computer Engineering, INESC, Universidade de Coimbra e-mail:sasilva@deec.uc.pt

In this work, we focus on the proposal and proof-of-concept of an environmental architecture for using mobile edge computing hardware and the IoT in the COVID-19 medical facilities. The issue in this proposal is to perform the monitoring of several patients using mobile edge computing hardware. This is suitable for applications with lower costs. For instance, this helps to solve this problem for field hospitals. This architecture employs a novel wearable solution for healthcare workers, communicating with a local server on a wireless network server, which also works as an access point. This server observes the conditions of the patients and workers, providing a management window with the conditions. This system also provides real-time information for the wearable edge devices, available using a Head-Up Display (HUD). Thus, the main contribution of this work is:

- A proposal and proof-of-concept of an environmental architecture for using mobile edge computing and IoT in medical facilities facing the COVID-19 pandemic.

The remainder of this text is organized as follows: In Section II, we present some theoretical references approaching works from the state-of-the-art publications related to this paper. We display the proposed architecture for an edge-based smart healthcare system in Section III. In Section V, we present the results of the proposed tests in the context of this work, providing the first insights and discussions. Finally, we present our conclusions and final discussions in Section VI.

II. Edge Computing and Healthcare

The Internet of Things provides a fertile environment for developing novel solutions towards healthcare. These solutions can gather data on the physical conditions of the users and provide this data using a web service [8]. The communication aspect provides fast information flow throughout the time, creating the possibility to propose, generate, and validate new appliances [9].

Intelligent healthcare systems integrate edge devices acquiring data from body-worn and body-proximal sensors. The acquired information generates valuable insights using data fusion, big data, and machine learning techniques [10]. Applications can employ cloud-based methods to process the acquired data [11]. Also, another option is processing the information in devices closer to the edge [12]. Thus, using innovative diagnostics and real-time monitoring can provide relevant improvement in medical facilities.

As much as smart healthcare systems are a topic of significant interest, most applications are designed to acquire data from people in the role of patients [13]. Nonetheless, cooperative edge systems can monitor the conditions of professionals exposed to stress and hazards [14]. This perspective enhances the development of an edge-based smart healthcare system for the COVID-19, aiming to aid not only the patient's monitoring but also the healthcare workers exposed to contamination dangers.

What has been previously presented regarding edge computing and COVID in this field are surveys [15], automated selection of services [16], a framework exploring potential DL algorithms to run on edge devices and cloudlets [17], other frameworks and algorithmic approaches [18], [19], or some fixed prototype solutions regarding another desired task [20]. None of them target the same issue presented here. The targeted problems are different, unrelated or even theoretical at some points.

III. Edge Computing - Architecture Proposal

As stated, the usage of edge devices for performing cooperative tasks is usually based on a wireless network environment, generally associated with the IoT concept [21]. The cooperation happens between edge devices and the edge server, as they gather patients' and workers' data, analyzing and providing insights for field and management workers. This solution targets smaller scale hospitals, as larger facilities can benefit from more powerful computing resources. Thus, the first decision on this system's architecture proposal is the networking environment. As many structures are adapted or quickly built and deployed, such as mobile field hospitals [3], a portable solution is ideal for quickly deploying a management structure. Finally, the network type of the solution is a Wireless Local Area Network (WLAN), given the ranges that the connectivity must reach [22]. Figure 1 displays the general proposed architecture.

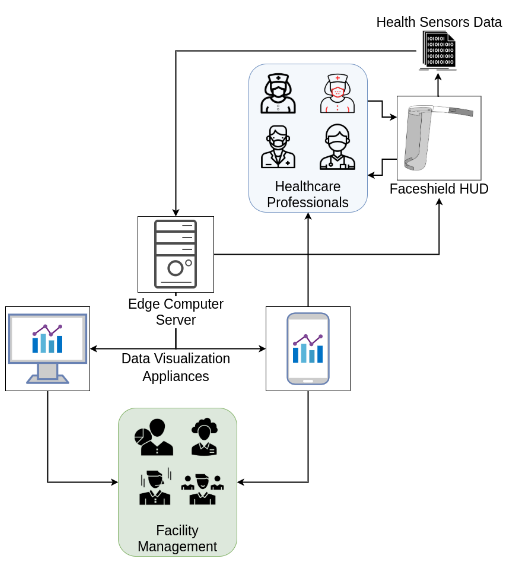


Fig. 1. Overview of the proposed architecture.

The WLAN-server is a local edge computing processing center that works as an access point for wearable devices and computer connections. This module gathers data from all the sensor nodes in the network, performing processing tasks to extract information and insights and providing a piece of high-level information as feedback in wearable, mobile, and computer terminal interfaces. This option enhances the capabilities of the Internet of Things appliances [23], enabling to reach higher-level insights from the acquired data.

In this context, health sensors are wearable devices worn by healthcare professionals. In our proposal, the sensor nodes are composed of sensors attached to single-core computer-on-modules, powered by power banks. They perform low processing tasks of data acquisition, pre-processing, and two-way communication to send the gathered information and receive feedback. Computer stations and mobile devices can also work as interfaces for medical professionals and facility managers.

A. Edge Computing Devices Description

Fig. 2. Face shield HUD Prototype. This device contains the camera sensor and the workers' health monitoring sensors.

For this work, we produced a wearable device, a prototype of the face shield HUD. We previously built and validated this prototype [24] over a mandatory safety device in many healthcare facilities and using a Raspberry Pi Zero W as its computer on module. Figure 2 displays the produced prototype.

This wearable device's objective is to monitor the user's health conditions and help retrieve information from a distance. With this device, we test some aspects of the proposed solution. The first relevant aspect of the solution is the computer module. Wearable appliances are usually power- and cost-restrictive. Thus, the first decision is to use a single-core ARM-based computer-on-module as a baseline to develop the solution. Another essential aspect of the computer-on-module is a WLAN-capable network board.

The next factor to consider is the sensor configuration. Traditionally, one of the most relevant features of wearable devices is increasing the user's awareness of the surrounding environment and his conditions [25]. This prototype has two different sensors. The first one is a pulse-oximeter and temperature sensor, which provides the data from its user's health conditions. The other sensor is a camera to acquire data from the environment. In the context of the healthcare facility, the camera may provide information from the patients' medical records using QR codes. Finally, the last aspect of it is the feedback interface. For this matter, we proposed a head-up display (HUD) to provide the user with high-level information.

B. Terminal and Mobile Interfaces

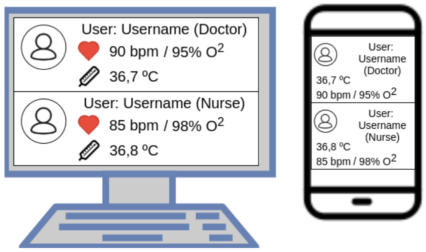


Fig. 3. Proposed Interfaces Illustration.

Another aspect of the proposed architecture is the management interfaces. These applications aid the facility managers in the decision process. It aids both monitoring and detecting early signs of contamination in healthcare professionals, [7]. In Figure 3, we display some examples of real-time monitoring of healthcare professionals' conditions.

C. Edge Computer Server Node

The network edge is composed of devices that handle computing tasks in edge computing. The edge device must be designed to handle such tasks efficiently and support requirements such as reliability. Integrating the IoT elements into the system happens through an edge computer server. This device performs two different tasks on this system. The first one acts as a gateway for receiving, storing, and broadcasting data from the wearable devices and the management terminal interfaces. The second task is submitting the acquired data to information extraction algorithms. To achieve this objective, we proposed using a portable device, called an edge computer server, with the hardware requirements.This hardware presents a Raspberry Pi4, with 4GB of RAM and 64GB storage, with WiFi support, with an attached battery. Our tests with this device achieved autonomy of around 12 hours of intensive usage. This device is suitable for more processing, as the computer on the wearable device has very limited resources.

D. System Integration

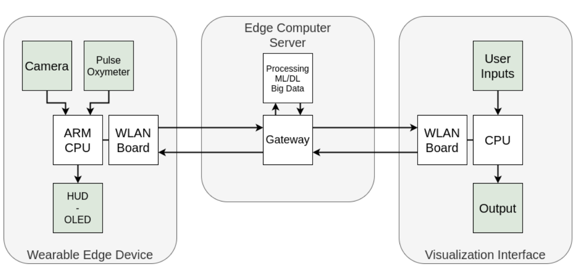


Fig. 4. Data Flow Diagram.

To understand the functioning of this system, we also provide an analysis of the data flow. Figure 4 provides a representation of the main components of this architecture and the visualization of the data flow. This representation helps to understand the expected behavior of each element according to its components. Also, it helps to examine the timing constraints from each part of the process.

There are three types of elements in the proposed architecture. The first element is the wearable edge computing device, represented by the face shield HUD presented in Subsection III-A. This element gathers data from the sensors, providing some pre-processing to turn them into information.

The central element is the Edge AI Computer/WLAN Server. This element is a processing center in an embedded edge computer, which also manages the network connections. As we provide the access point in a computer, it also can perform parallel data fusion and analysis tasks. This process provides high-level information for other elements from the architecture. Finally, the last element in this organization is the management interface. In Figure 4, this element is represented by the generic interface block. In both mobile devices or a computer terminal, the main elements are the user's inputs, communication with WLAN, processing, and output.

IV. Experimental Tests

A relevant aspect of distributed architectures is networking performance. This feature directly affects the quality of the provided services in IoT-based systems, having consequences in the system's real-time capabilities [26]. Thus, the experimental setup evaluates the timing constraints for the data flow and processing.

A. Real-Time as Quality-of-Service

To evaluate these aspects, we perform a QoS-based timing constraint test. The experiment was designed as a QoS formalization, presented on similar studies concerning IoT and Wireless Sensor Networks [27] to evaluate soft real-time constraints as network timing constraints.

At first, we divide the experiment time in discrete intervals, as the set $T = t_i, i \in \mathbb{N}$, where $t_{i+1} - t_i = \theta$, where θ is a constant sampling time. The soft real-time deadline will be represented by ϕ, where $\phi = k \times \theta, k \in \mathbb{N}^*$. From these primary statements, we establish the following definitions:

Definition 1. Let $D = d_i$ be the finite set of nodes consuming and producing data from the middleware node, where $i \in \mathbb{N}$;

Definition 2. Let $E = e_i$ be the finite set of events that each node performs, where $i \in \mathbb{N}$;

Definition 3. Let $L = l_{d,e}$ be the length of time interval that the node d takes to perform an event e, where $d \in D$ and $e \in E$;

Definition 4. Let $P = p_i$ be the set of patterns of events to be observed in the devices, where $p_i = E_i$, $E_i \subset E$ and $i \in \mathbb{N}$;

Definition 5. Let $O = o_i$ be the finite set of observations of a certain pattern $p_i \in P$ on the devices;

The equation that represents the elapsed time λ to observe a particular pattern $p_i \in P$ is:

$$\lambda_{o_i} = \sum l_{d,e_k} | \forall e_k \in o_i, o_i = O_{p_i} \quad (1)$$

In this case, each device in the network composition can have its single ϕ_i soft real-time deadline. Given this equation, let $\hat{O}$ be a subset of O, where $\lambda_{o_i} \leq \phi_i$, $\forall o_i \in \hat{O}$. Finally, given the sets O and $\hat{O}$:

Definition 6. Let N be the number of elements on the set O;

Definition 7. Let N_h be the number of elements on the subset $\hat{O}$;

The following equation will represent the quality factor Q_f:

$$Q_f = \frac{N_h}{N}(\times 100\%) \tag{2}$$

The nodes will try to gather or update data from the server node in parallel on each test. This result represents how often the nodes execute a pattern of events without violating the soft real-time constraints. This perspective allows experimenting on how increasing the number of devices producing and consuming data affects the network quality factor.

The proposed experiment is divided in 3 stages:

- **Stage 1:** Defining the soft real-time constraint;
- **Stage 2:** Evaluating the effect of stressing the system by increasing the number of edge devices;
- **Stage 3:** Evaluating the effect of stressing the system by increasing the number of client interfaces.

In **Stage 1**, we run a minimal version of the appliance, containing one element from each class represented in Figure 4. We calculate the minimal ϕ_i soft real-time constraint from this data to obtain a quality factor of 0.95 (95%). This constraint will be used as a reference to evaluate the system behavior when increasing the number of devices, with a block length of $\theta = 2$ ms. Also, we use this data to evaluate the internal timing constraints of each device for simulation purposes.

In **Stage 2**, we use the constraints from the first stage to evaluate the system behavior when increasing the number of edge devices. The behavior of multiple devices will be simulated in the same computer-on-module boards used to produce the prototype. The timing for each task on the simulated device comes from the evaluation from the previous stage.

Finally, in **Stage 3**, we evaluate the effect of increasing the number of terminal interfaces. For this matter, we instantiate several simulated terminals as individual processes in a computer connected to the network.

These stages work as validation for the proposed architecture and an overview of the constraints for increasing the number of devices and clients in this system. In this work, we focused on evaluating these constraints through simulations.

B. Server Behavior Evaluation

For the second experiment, we consider the influence of patients and medical professionals in the environment. Also, in this case, we evaluate the timing aspect. We evaluate how the overload of patients and medical devices in the network influence the server capability of answering requests. For this matter, we consider the dataflow presented in Figure 5.

For this matter, once again, we emulated multiple clients producing and consuming data from the edge server. We consider the patient appliance a temperature and pulse-oximeter sensor, with the same timing constraints presented in the previous stage.

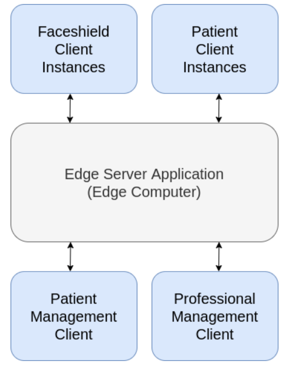


Fig. 5. Data Flow for the Experimental Setup on the Second Test.

In this perspective, five categories of elements are present in the solution. The first and central element is the edge computing server. It runs an application that accepts connections from multiple clients, storing their data and making it available for requests for management applications in the wireless local area network. The second and third elements are management clients. The patient management client inquiries the data from a particular patient using his name as an identifier. The key worker management client retrieves the data from all professionals working in the facility. Finally, the fourth and fifth elements are the instances for healthcare device clients. They work similarly, as they both require readings from the same sensors.

To evaluate the performance of this system in the network, we analyzed two different scenarios. Initially, we increased the number of instances from the face shield device from 5 to 50, using a constant number of 20 patient instances. Then, we tested a second scenario, where we increased the number of patient device instances from 5 to 50, with a constant number of 20 face shield device instances. We executed one instance of the patient management client in both scenarios and another using a key workers' management client. The choice for this test was to evaluate the effect of scaling each of the client types, with a fixed number of clients for the other type. Thus, we selected this methodology over other options for stressing the system and testing its scalability. These numbers were selected based on the number of patients at some field hospitals [28].

The parameter for this response is the server answer time, measured during the whole execution. For this matter, we evaluated the average time for answering the requests and the moving mean during the program execution. We executed both tests for around 600s, changing the payload on the variable that stresses the test every 60s.

V. Results

A. Devices Internal Constraints Evaluation

Initially, we evaluate the device's internal timing constraints. This stage is the first step in creating a validation environment where the conditions are similar to those faced in the field. For this matter, we describe both the applications from the server and the terminal and wearable clients. Also, we experimentally determine the timing constraints in the prototype to create valid simulation environments.

The first element to describe is the server application. The server keeps the updated data from all the clients in the memory for the experimental set, sending the complete information back to the inquiring terminal clients. Also, the server keeps track of the timing constraints for experimental purposes.

To support the connection of multiple clients, we developed a multi-threaded server. Each thread is responsible for handling a single client. The application employs two critical sessions: Registering the latest health signals from each wearable client and writing to the log file's handler. In Figure 6, we display the algorithm performed by each thread handling a single client connection, identifying the critical sessions in red blocks.

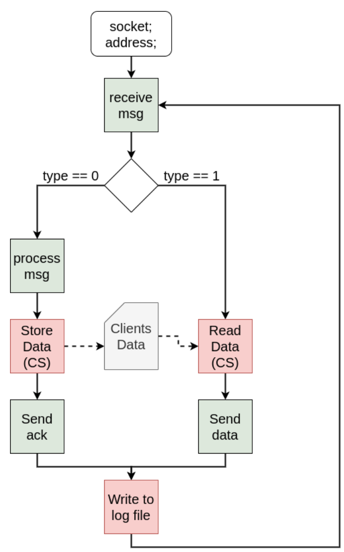


Fig. 6. Edge Server Node Algorithm.

The processes start with the handlers for the connection creators and information. The client address and port information also work as indexing for the client data stored in the memory. This appliance has two types of clients: terminal clients and wearable clients. Messages of terminal requests are identified by "type == 1", and messages from the wearable clients are identified by "type == 0".

The threads store the clients' information in the same data structure. Therefore, reading and writing in this structure is the first critical section. The data coming from the clients is interpreted (stage: "process msg") and stored (stage: "Store Data"), after which the thread sends an acknowledge message (stage: "Send ack"). Requests from terminals receive data from all the clients available (stage: "Read Data"), sending it afterwards (stage: "Send data"). Finally, the threads share the same log file. Thus, it is also a second critical section (stage: "Write to log file").

The first client type is the terminal client. This client has two main stages: (i) acquire the data from the server, and (ii) process data and display a screen frame using a background template and the gathered information. Figure 7 displays the background and an example of a card generated with simulated information. Finally, the threads share the same log file. Thus, it is also a second critical section.

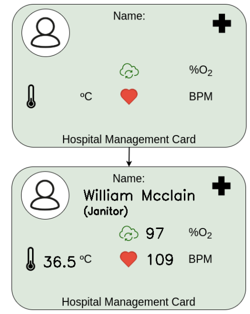


Fig. 7. Visualization Prototype Application Example.

The wearable client has three different tasks: (i) reading the sensors data, (ii) sending data and receiving the answer from the server, and (iii) processing the data and displaying a frame in the OLED display. While tasks (i) and (iii) are internal, task (ii) is network-dependant. To determine the timing requirements to run this application, we performed tasks (i) and (iii) for 120 seconds on the prototype displayed in Figure 2, measuring the times to perform both internal issues. On average, the results obtained indicate that:

- Task (i) takes an average time of 2.25 ± 0.10 ms;
- Task (iii) takes an average time of 11.8 ± 14.3 ms.

With this data, we created an application to emulate the client's behavior for the stress tests (Stages 2 and 3). The prototype also uses the application to determine the real-time constraint in Stage 1.

B. Stage 1: Real-Time Constraints Definition

To define the real-time constraints, we evaluate the timing requirements of the tasks performed by each device in the context of this test. For this matter, we apply the definition of quality factor presented in Section IV. We evaluate the minimal configuration in this stage, with only one element of each class running on the targeted hardware. The server runs in a Raspberry Pi 3 Model B, the interface runs in a desktop computer, and the wearable application runs in an embedded raspberry pi zero w, mounted on the face shield prototype.

In this first stage, we want to establish the soft real-time constraint ϕ. We divide the experiment time into discrete-time blocks of $\theta = 2$ ms. For this matter, we establish a target quality factor of $Qf = 0.95$. Then, we establish the minimum number of time blocks to obtain the desired constraint as a factor of the number of blocks k to obtain the targeted objective. As each class of device has its unique set of tasks, we establish an individual constraint for each type.

TABLE I
REAL TIME CONSTRAINT DEFINITION RESULTS

	Average time (ms)	Requirement (k)
WEARABLE	27.6 ± 27.4	37 blocks
INTERFACE	27.7 ± 33.3	33 blocks
SERVER	54.4 ± 50.4	65 blocks

In the following stages, we use the constraints presented in Table I to evaluate the effect of stressing the system with more clients and more interfaces. The server still runs in the Raspberry Pi 3 Model B for the stress tests. One computer runs the interface appliance, and another executes multiple simulated client applications instances.

C. Stage 2: Stressing the System with more Wearable Edge Devices

The first test's objective is to understand the effect of increasing the number of wearable devices on network performance. We ran from 5 up to 50 instances of the application that emulates a wearable device's behavior on the network for this test. During the whole period of the test, we executed a single interface application.

At first, we determined the overall values for the quality factor from the non-variable elements. From our tests, during the whole period, the results indicate that:

- The quality factor for the interface was $Q_f = 0.989$;
- The quality factor for the server was $Q_f = 0.937$;

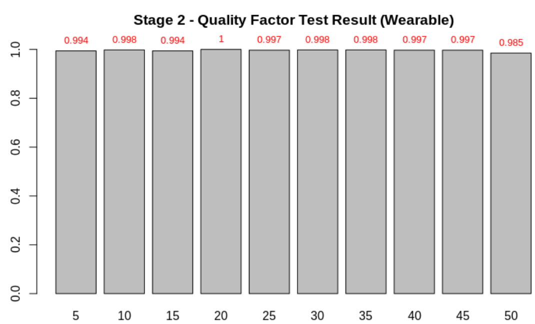


Fig. 8. Quality Factor Test Results.

Figure 8 displays the results for the quality factor test. The loss of quality when increasing the number of instances in the system was around 1%. This effect indicates that this system architecture can gather and manage data from various devices without compromising the soft real-time constraint. Also, the other constraints faced a low compromising, enforcing this preliminary conclusion.

D. Stage 3: Stressing the System with more Terminal Devices

In this test, we want to understand how the applications that demand a higher amount of data respond to concurrency stress. In this case, we varied the number of emulated terminal devices. The terminal devices receive the data from all connected clients for the management applications. Again, we executed from 5 up to 50 instances of the application. During the period of the test, we emulated 20 wearable clients.

At first, we determined the overall values for the quality factor from the non-variable elements. From our tests, during the whole period, the results indicate that:

- The quality factor for the wearable devices was $Q_f = 0.986 \pm 0.001$;
- The quality factor for the server was $Q_f = 0.885$;

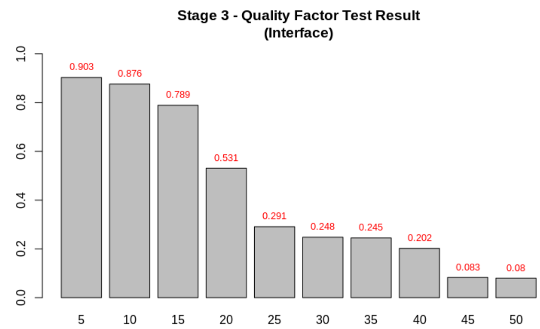


Fig. 9. Quality Factor Test Results.

Figure 9 displays the results for the quality factor test. In this case, increasing the number of instances significantly jeopardizes the real-time feature of this application. The decrease in the quality factor on the server enforces this trend. This result indicates that, as much as possible, it is interesting to execute most of the data processing in the edge server, as broadcasting large amounts of data can decrease the reliability of the architecture itself.

E. Server Behavior Evaluation

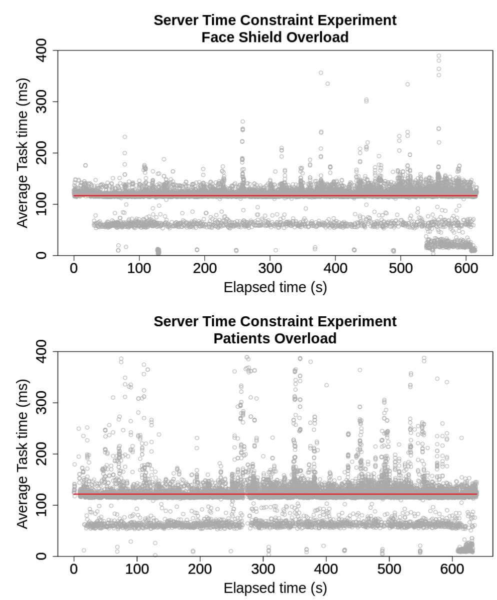


Fig. 10. Performance Test Results.

As presented in Section IV, this last test evaluates the effect of clients overload in the system behavior. Figure 10 displays the results obtained for the tests. We display local results for the moving average during the test in gray. The red line indicates the system's average time to process a request and send the result.

For both tests, we took a moving average from 100 subsequent samples. In the first test, the average time required to answer a single sample was 116.9 ± 19.11 ms. The moving average displays that the average time does not vary much during the execution, with some outlier values. This first result indicates the system stability, even when stressed with more devices. From the second test, we extract that the average time constraint was 121.8 ± 73.28 ms. Although the moving average also mainly displays that the expected behavior is similar to the first test, i.e., stable around the global average, more outlier values are detected, reflecting a more significant standard deviation. Figure 11 displays the density plot for both experiments. They confirm that the majority of the results are centered around the average value indicated by the red line.

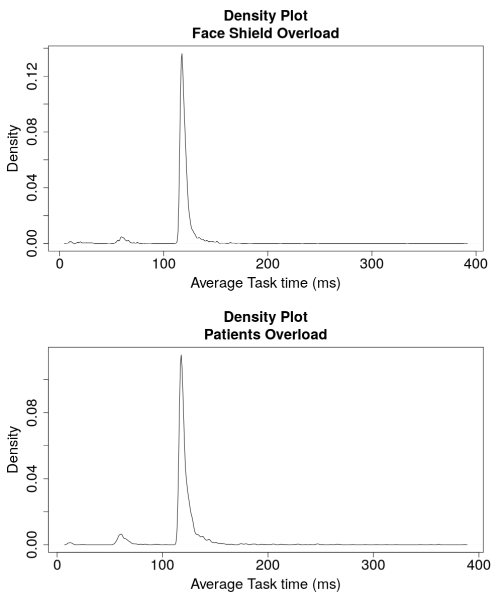


Fig. 11. Density Plot for the Performance Test Results.

VI. Conclusions and Discussion

In this work, we proposed an edge computing cooperative architecture for medical facilities, primarily directed to aid the COVID-19 exposed professionals. The architecture is an intelligent healthcare wearable-based system with portability and ease to deploy. To validate this architecture, we presented a complete prototype environment to establish and test the devices' real-time capabilities. Finally, we also evaluate how stressing the system's variables affect real-time performance.

Smart healthcare systems based on the Internet of Things and Wearable Technologies are a topic of keen interest, as displayed in the first sections. There is a particular urge to attend the medical community with low-cost and secure deployment solutions. In this context, we propose a novel wearable device created over the protective face shield. This design is widely employed in the field medical facilities to avoid contamination. This device has a HUD to present useful high-level information for the user. The edge computer server is an embedded computer-on-module solution with WLAN support and capable of data-processing tasks. Although not every hospital environment supports wi-fi, we assess some in which it is possible to perform this deployment.

We performed real-time constraint evaluations for systems with networking features to validate the architecture capabilities. This formalization considers both internal processing and network-related events to establish the real-time constraints. From the test results, we understand that the addition of wearable edge devices concurrently producing and feeding the server with data does not jeopardize the quality of the proposed architecture. Nevertheless, as the interface consumes more bandwidth, requiring more data to run its application, the addition of further elements deteriorates the quality of the provided service. The server behavior tests also indicate that the system bears the usage by many parallel clients, with some stability guarantee. In the test appliance, the system was capable of processing around eight requests per second. This last test is also an indicator of this application's feasibility. Future work involves confirming these trends by performing in-field tests with the proposed devices. This solution is nonetheless limited to small-sized facilities. To develop a system in scale for larger spaces, more computational resources and investment are required.

Acknowledgements

The authors would like to thank FAPEMIG, CAPES, CNPq, the University of Coimbra, and the Federal University of Ouro Preto for supporting this work. This work was partially funded by CAPES (Finance Code 001), Pró-Reitoria de Pesquisa e Inovação (23109.000928/2020-33), and CNPq (306572/2019-2 and 308219/2020-1)

References

[1] V. Vippalapalli and S. Ananthula, "Internet of things (iot) based smart health care system," in *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES)*, pp. 1229–1233, IEEE, 2016.

[2] M. Alhussein and G. Muhammad, "Automatic voice pathology monitoring using parallel deep models for smart healthcare," *IEEE Access*, vol. 7, pp. 46474–46479, 2019.

[3] Z. Chen, S. He, F. Li, J. Yin, and X. Chen, "Mobile field hospitals, an effective way of dealing with covid-19 in china: sharing our experience," *BioScience Trends*, 2020.

[4] A. S. Kliger and J. Silberzweig, "Mitigating risk of covid-19 in dialysis facilities," *Clinical Journal of the American Society of Nephrology*, vol. 15, no. 5, pp. 707–709, 2020.

[5] J. Wong, Q. Y. Goh, Z. Tan, S. A. Lie, Y. C. Tay, S. Y. Ng, and C. R. Soh, "Preparing for a covid-19 pandemic: a review of operating room outbreak response measures in a large tertiary hospital in singapore," *Canadian Journal of Anesthesia/Journal canadien d'anesthésie*, pp. 1–14, 2020.

[6] V. N. Prachand, R. Milner, P. Angelos, M. C. Posner, J. J. Fung, N. Agrawal, V. Jeevanandam, and J. B. Matthews, "Medically-necessary, time-sensitive procedures: A scoring system to ethically and efficiently manage resource scarcity and provider risk during the covid-19 pandemic," *Journal of the American College of Surgeons*, 2020.

[7] S. Shah, K. Majmudar, A. Stein, N. Gupta, S. Suppes, M. Karamanis, J. Capannari, S. Sethi, and C. Patte, "Novel use of home pulse oximetry monitoring in covid-19 patients discharged from the emergency department identifies need for hospitalization," *Academic Emergency Medicine*, vol. n/a, no. n/a.

[8] S. B. Baker, W. Xiang, and I. Atkinson, "Internet of things for smart healthcare: Technologies, challenges, and opportunities," *IEEE Access*, vol. 5, pp. 26521–26544, 2017.

[9] A. Rizwan, A. Zoha, R. Zhang, W. Ahmad, K. Arshad, N. A. Ali, A. Alomainy, M. A. Imran, and Q. H. Abbasi, "A review on the role of nano-communication in future healthcare systems: A big data analytics perspective," *IEEE Access*, vol. 6, pp. 41903–41920, 2018.

[10] G. Manogaran, R. Varatharajan, D. Lopez, P. M. Kumar, R. Sundarasekar, and C. Thota, "A new architecture of internet of things and big data ecosystem for secured smart healthcare monitoring and alerting system," *Future Generation Computer Systems*, vol. 82, pp. 375–387, 2018.

[11] M. S. Hossain, G. Muhammad, and A. Alamri, "Smart healthcare monitoring: A voice pathology detection paradigm for smart cities," *Multimedia Systems*, vol. 25, no. 5, pp. 565–575, 2019.

[12] M. A. Salahuddin, A. Al-Fuqaha, M. Guizani, K. Shuaib, and F. Sallabi, "Softwarization of internet of things infrastructure for secure and smart healthcare," *arXiv preprint arXiv:1805.11011*, 2018.

[13] S. Tuli, N. Basumatary, S. S. Gill, M. Kahani, R. C. Arya, G. S. Wander, and R. Buyya, "Healthfog: An ensemble deep learning based smart healthcare system for automatic diagnosis of heart diseases in integrated iot and fog computing environments," *Future Generation Computer Systems*, vol. 104, pp. 187–200, 2020.

[14] M. C. Silva, V. J. Amorim, S. P. Ribeiro, and R. A. Oliveira, "Field research cooperative wearable systems: Challenges in requirements, design and validation," *Sensors*, vol. 19, no. 20, p. 4417, 2019.

[15] A. Sufian, A. Ghosh, A. S. Sadiq, and F. Smarandache, "A survey on deep transfer learning to edge computing for mitigating the covid-19 pandemic," *Journal of Systems Architecture*, vol. 108, p. 101830, 2020.

[16] Y. Abdulsalam and M. S. Hossain, "Covid-19 networking demand: an auction-based mechanism for automated selection of edge computing services," *IEEE Transactions on Network Science and Engineering*, 2020.

[17] M. A. Rahman and M. S. Hossain, "An internet-of-medical-things-enabled edge computing framework for tackling covid-19," *IEEE Internet of Things Journal*, vol. 8, no. 21, pp. 15847–15854, 2021.

[18] H. Ghayvat, M. Awais, P. Gope, S. Pandya, and S. Majumdar, "Recognizing suspect and predicting the spread of contagion based on mobile phone location data (counteract): a system of identifying covid-19 infectious and hazardous sites, detecting disease outbreaks based on the internet of things, edge computing, and artificial intelligence," *Sustainable Cities and Society*, vol. 69, p. 102798, 2021.

[19] E. Laxmi Lydia, C. Anupama, A. Beno, M. Elhoseny, M. D. Alshehri, and M. M. Selim, "Cognitive computing-based covid-19 detection on internet of things-enabled edge computing environment," *Soft Computing*, pp. 1–12, 2021.

[20] C. H. Loke, M. S. Adam, R. Nordin, N. F. Abdullah, and A. Abu-Samah, "Physical distancing device with edge computing for covid-19 (paddie-c19)," *Sensors*, vol. 22, no. 1, p. 279, 2021.

[21] C. Long, Y. Cao, T. Jiang, and Q. Zhang, "Edge computing framework for cooperative video processing in multimedia iot systems," *IEEE Transactions on Multimedia*, vol. 20, no. 5, pp. 1126–1139, 2017.

[22] M. S. Mahmoud and A. A. Mohamad, "A study of efficient power consumption wireless communication techniques/modules for internet of things (iot) applications," 2016.

[23] C. Chang, S. N. Srirama, and R. Buyya, "Indie fog: An efficient fog-computing infrastructure for the internet of things," *Computer*, vol. 50, no. 9, pp. 92–98, 2017.

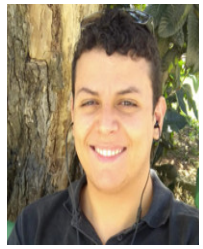
[24] M. Silva., R. Oliveira., T. D'Angelo., C. Garrocho., and V. Amorim., "Faceshield hud: Extended usage of wearable computing on the covid-19 frontline," in *Proceedings of the 23rd International Conference on Enterprise Information Systems - Volume 1: ICEIS,*, pp. 893–900, INSTICC, SciTePress, 2021.

[25] S. Majumder, T. Mondal, and M. J. Deen, "Wearable sensors for remote health monitoring," *Sensors*, vol. 17, no. 1, p. 130, 2017.

[26] K. Cao, G. Xu, J. Zhou, T. Wei, M. Chen, and S. Hu, "Qos-adaptive approximate real-time computation for mobility-aware iot lifetime optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 10, pp. 1799–1810, 2018.

[27] M. Silva and R. Oliveira, "Analyzing the effect of increased distribution on a wearable appliance," in *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2, pp. 13–18, IEEE, 2019.

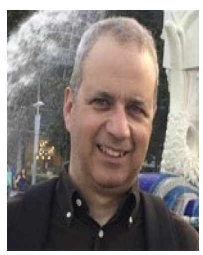
[28] M. D. des Déserts, Q. Mathais, A. Luft, J. Escarment, and P. Pasquier, "Conception and deployment of a 30-bed field military intensive care hospital in eastern france during the 2020 covid-19 pandemic," *Anaesthesia, Critical Care & Pain Medicine*, vol. 39, no. 3, p. 361, 2020.



Mateus Coelho Silva is currently a Ph.D. student in Computer Science at the Federal University of Ouro Preto. His current research interests include Cyber-Physical Systems, IoT, Wearable Devices and Robotics. Contact him at mateuscoelho.ccom@gmail.com.

Andrea G.C. Bianchi is an Associate Professor at the Computing Department at the Federal University of Ouro Preto (UFOP). She obtained her Ph.D. in Computational Physics at São Paulo University (USP), Brazil, and acted as a postdoc at Lawrence National Berkeley Laboratory in Data Analytics and Visualization Group. She has experience in computer vision and machine learning systems, building tools capable of extracting knowledge of images. Her research interests are image processing and analysis, pattern recognition, machine learning. Contact her at andrea@ufop.edu.br.

Servio P. Ribeiro is currently a Full Professor in the Biology Department at the Federal University of Ouro Preto (UFOP). He received his Ph.D. Degree in Ecology from Imperial College at Silwood Park, University of London (1998). He has experience in Ecology and acts on various subjects, including Plant-Insect Interaction and Evolutive Ecology in Forest Canopies, Bioindication of degraded areas, Ecohealth, and Evolutive Ecology of Parasites and Viruses. Contact him at spribeiro@gmail.com.



Jorge Sá Silva received his PhD in Informatics Engineering in 2001 from the University of Coimbra, where is Associate Professor with Habilitation at the Department of Electrical and Computer Engineering of the Faculty of Sciences and Technology of the University of Coimbra and a Senior Researcher of Laboratory of Communication and Telematics of Centre of Informatics Engineering of University of Coimbra, Portugal. His main research interests are Internet of Things, Network Protocols, Machine to Machine, and Wireless Sensor Networks. He is a senior member of IEEE, and he is a licensed Professional Engineer. Contact him at sasilva@deec.uc.pt.

Ricardo Augusto Rabelo Oliveira received his Ph.D. degree in Computer Science from the Federal University of Minas Gerais in 2008. Nowadays he is an Associate Professor in the Computing Department at the Federal University of Ouro Preto. Has experience in Computer Science, acting on the following subjects: Wavelets, Neural Networks, 5G, VANT, and Wearables. Contact him at rrabelo@gmail.com.