PARTNER: Development Platform for Self-Adaptive IoHT Application Microservices

Evilasio Costa Junior[1,*], Francisco Victor da Silva Pinheiro[1], Rhenara Alves Oliveira[1], Ismayle de Sousa Santos[2] and Rossana Maria de Castro Andrade[1]

[1]*Federal University of Ceará, Fortaleza, Brazil*
[2]*State University of Ceará, Fortaleza, Brazil*

Abstract

Monitoring people's health through sensor data collection is a topic that has gained prominence in recent years with the advent of the Internet of Health Things (IoHT) and the growing perception of population aging in many countries. IoHT brings several challenges for application development and some of them are common between different IoHT applications, such as the great dynamicity of IoHT environments, and others are specific for each domain, such as the need to relate sensor data with Quality of Life (QOL) aspects in QOL monitoring applications. So, we used two concepts to deal with these challenges: self-adaptive systems that are designed to automatically adapt to changes in context, which makes them viable for addressing the dynamism of IoHT systems; and microservices architecture that enables processing parts of the application as services, which can be reused by different applications, including those specific to certain domains. Thus, in this paper, we present PARTNER, a development platform for self-adaptive IoHT applications based on microservices. Moreover, we developed a proof of concept (PoC) application called FRIEND to evaluate the mobile module of the PARTNER platform. An experiment was also conducted to evaluate the acceptance of the PoC application with real users.

Keywords

Internet of Health Things, Self-Adaptive Systems, Microservices Architecture

1. Introduction

Many computer science studies in recent years have focused on software solutions that apply the Internet of Things (IoT) to healthcare [1], called Internet of Health Things (IoHT) [2]. IoHT discusses how IoT devices connected to the Internet of Things can help monitor, treat and even improve the health of patients, both inside and outside of hospitals [3].

Research on IoHT focuses on applying the concepts and benefits of IoT to healthcare [4]. The number of studies on IoHT has increased recently, focusing mainly on healthcare monitoring solutions, health risk situations and improving people's quality of life [5]. Non-invasive glucose sensing, oxygen saturation monitoring, medication management, fall detection in the elderly, and an ingestible sensor to measure medication adherence are examples of IoHT applications [1].

This field has encountered challenges in standardizing IoT devices and platforms tailored for healthcare applications development, quality assurance, data security, and privacy [6]. Addressing these issues requires evolving software engineering methods to enhance the development of reliable IoT systems [7].

Although we have many studies showing this growing concern about the demand for IoHT systems, these studies are mainly based on the view of researchers or through data mining, discussions of developers on Question and Answer sites, and development forums, such as Stack Overflow [8]. They often do not comprehensively address the overall aspects of IoHT software development.

ICAIW 2024: Workshops at the 7th International Conference on Applied Informatics 2024, October 24–26, 2024, Viña del Mar, Chile
*Corresponding author.
evilasiojunior@ufc.br (E. C. Junior); victor.pinheiro.ce@alu.ufc.br (F. V. da Silva Pinheiro); rhenaraalves21@alu.ufc.br (R. Alves Oliveira); ismayle.santos@uece.br (I. de Sousa Santos); rossana@ufc.br (R. M. de Castro Andrade)
0000-0002-0281-2964 (E. C. Junior); 0000-0002-5693-0499 (F. V. da Silva Pinheiro); 0009-0002-2501-9800 (R. Alves Oliveira); 0000-0001-5580-643X (I. de Sousa Santos); 0000-0002-0186-2994 (R. M. de Castro Andrade)

Many of these challenges of IoT and IoHT applications are directly related to the constantly change of context in which these applications are introduced [9] [10]. In this sense, there are self-adaptive systems (SAS), which are designed to self-adapt at runtime to changes in the environment [11]. SAS is ideal for execution in dynamic systems, such as IoHT systems, providing a way to mitigate problems arising from the great dynamism of these environments through self-adaptation strategies [10, 12].Then, it is possible to observe the growing number of IoT applications developed as self-adaptive systems designed to automatically adapt their behavior in response to changes in the context [13].

Considering the challenges of building self-adaptive IoHT applications, this work proposes the PARTNER platform that helps develop this type of application. This platform has artifacts to support the developer to building self-adaptive IoHT applications. The mobile module of this platform is composed of microservices and a framework developed using the React Native programming language. It incorporates the self-adaptation steps of the MAPE-K adaptation loop proposed by IBM [14] as internal components of the framework. These components are responsible for the functioning of the internal logic application and communicate with each other through the observer software pattern. It includes components that facilitate the collection of data from native smartphone sensors and also communicate with Health Application APIs, enabling the collection of data from wearable devices such as smartbands and smartwatches.

We evaluated the functionality of our platform by developing a proof of concept (PoC) application, using only smartphone sensors and data from the Google Fit API. The results demonstrate that the mobile module of the platform, including its services and framework, effectively supports the development of self-adaptive IoHT applications. We also evaluated the perception of usefulness and ease of use of the PoC, throut a feedback survey genereted based on Technology Acceptance Model (TAM) [15], that was answer for a group formed for six older adults that experimented to the PoC for two weeks.

The remainder of this article is organized as follows: Section 2 provides a detailed review of the related work, highlighting the current state of research in IoHT and identifying gaps in the existing literature. Section 3 describes the methodology used in our study, including the design and implementation of our plataform and the proof-of-concept (PoC) application. Section 4 presents the PARTNER platform while Section 5 describes the results of our evaluation, demonstrating the functionality and effectiveness of our platform. In section 6, the experiment carried out to evaluate the perception of usefulness and ease of use of the PoC is presented. Finally, Section 7 concludes the article by summarizing the key contributions and suggesting directions for future research.

2. Related Work

Some recent studies have proposed software artifacts to assist the development of IoHT applications [12, 16, 17]. These artifacts present different approaches to mitigate the difficulties inherent in developing this type of system.

The study [12] proposes a framework developed in Kotlin, called KREATION, to facilitate the development of self-adaptive Internet of Health Things (IoHT) applications on Android mobile devices. Given the increasing complexity and challenges posed by dynamic context changes in IoHT applications - such as energy consumption, network communication type, and latency - KREATION utilizes the Model-View-Control architecture and incorporates the MAPE-K adaptation loop.

Hence, this framework provides mechanisms for collecting data from Android smartphone sensors and the Google Fit API, integrating data from devices such as smart bands and smartwatches. The functionality and effectiveness of KREATION were demonstrated through the development of two proofs of concept, highlighting its capability to assist in developing self-adaptive IoHT applications.

The study [16] presents an innovative framework focused on pervasive and ubiquitous monitoring of the elderly, aiming to promote their well-being while living independently. The research investigates the integration of technologies such as Microsoft's Kinect, smartwatches, and video cameras to capture relevant data on daily activities and health status of the elderly. Emphasizing non-intrusiveness, the framework utilizes passive sensors like PIR to detect movement without compromising privacy. The results highlight the effectiveness of internally developed algorithms, enabling precise assessment of fall risks, postural stability, and sleep quality. Furthermore, the flexibility of the framework allows adaptation to different contexts and integration of new sensors and algorithms as needed.

This article [17] introduces a framework designed to enhance healthcare in assisted living environments by integrating Internet of Medical Things (IoMT) technologies and advanced big data analytics techniques. The primary goal is to enhance patients' quality of life by utilizing IoMT devices for continuous health data collection. These data are processed and analyzed using big data techniques, enabling valuable insights such as early diagnoses and personalized treatments. The proposed framework promotes an integrated and efficient approach to remote monitoring and healthcare management, standing out as a promising solution for advancing healthcare in assisted living settings. This approach facilitates timely medical intervention and holds significant potential to enhance healthcare service efficiency by adapting more precisely and proactively to individual patient needs.

In contrast to these frameworks, PARTNER proposes a development platform for self-adaptive IoHT applications based on microservices. This approach allows applications to dynamically adapt to changes in the IoHT environment, addressing common challenges such as dynamism and specific domain requirements like QoL monitoring. The proof of concept application FRIEND validates the PARTNER's mobile module, highlighting its potential for real-world applications and user acceptance.

While each framework contributes uniquely to the field of IoHT and elderly care, the adoption of plataforms to develop self-adaptive sytems like PARTNER plataform represents a promising direction. The systems developed using our plataform can effectively manage the complexities of IoHT environments and meet the specific requirements of different application domains, enhancing healthcare delivery and improving the quality of life for diverse populations.

3. Method

We employ Design Science Research (DSR) [18] to research and develop our solution. DSR delineates five stages: *Problem Awareness*, where research seeks to comprehend the rationale and identify the specific problem under study and its existing approaches; *Suggestion*, where a proposed solution to the identified problem is suggested; *Development*, where the proposed solution is developed and implemented; *Evaluation*, where the effectiveness of the solution is assessed; and *Conclusion*, where the results determine whether the proposed solution adequately resolves the problem, either fully or partially.

Figure 1 presents an overview of the methodology used; on the left are the DSR methodology stages, and on the right, how we developed each stage throughout our research. First, to contextualize the problem, we conducted a literature review that follows part of the formalism of the systematic literature review method proposed in [19]. This literature review sought to identify studies on developing IoHT solutions and the related challenges.

Based on the studies analyzed, we identified a gap for software artifacts that would help develop this type of solution; in particular, we chose to work with self-adaptive IoHT systems, as they are advantageous to address the challenge related to the dynamism of IoHT environments. In this sense, part of our work was based on studying the framework proposed in [12]. Alongside self-adaptive systems, we opted to employ microservices architecture due to its ability to easily integrate new functionalities as services and replace existing ones with minimal disruption to running IoHT applications. With this in mind, we propose the PARTNER, a platform for self-adaptive Internet of Health Things applications based on microservices architecture. PARTNER is a platform that aims to offer reuse artifacts to assist the development of self-adaptive IoHT applications based on microservices architecture. It was designed to function as a Mobile module, offering tools for developing mobile applications and services, and a Server module, where web services that may require greater processing can be accessed by mobile applications developed using PARTNER's mobile tools through an API.

With this proposal, our initial focus was on developing the mobile module of the PARTNER platform. The development of the Mobile module of the PARTNER platform and the proof of concept developed

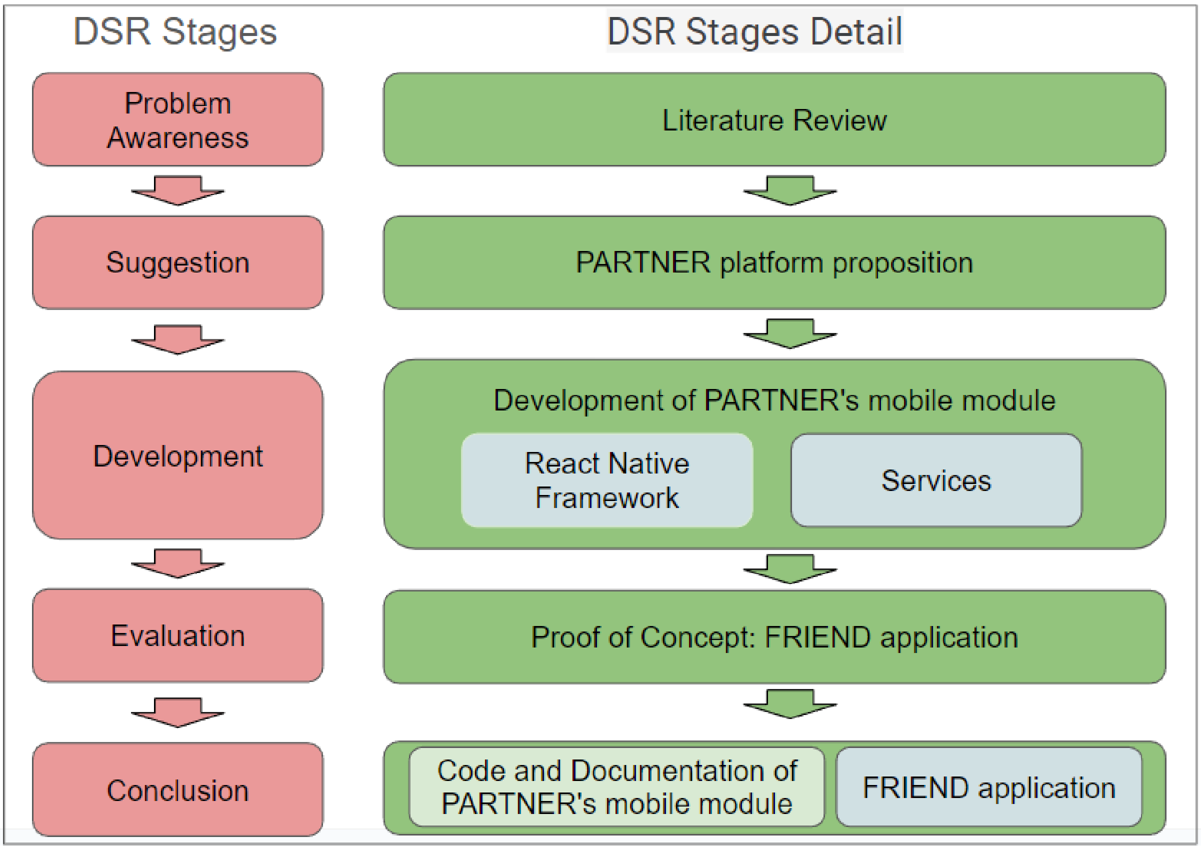


Figure 1: DSR Methodology Overview

followed the SCRUM agile method [20].

Furthermore, we used the echnology Acceptance Model (TAM) [15] as a basis to evaluate the PoC developed in relation to the usefulness and ease of use, through an experiment with six elderly people, who used the PoC application for two weeks.

4. PARTNER Platform

The Platform for self-adaptive service based internet of health things applications (PARTNER) is a development platform that seeks to assist developers in building IoHT applications.

PARTNER was designed to have two modules: the Mobile module and the Server Module. The mobile module contains a framework and a set of services to assist in developing mobile applications. The server module includes services that require greater processing power, such as smart algorithms processing services, machine learning and generative AI, storage, management services for large volumes of data, and data traffic security services. These Server module services can be accessed through a Web API by mobile or web applications. In this study, we present the mobile module of the PARTNER platform.

4.1. PARTNER's mobile module

Figure 2 presents the architecture of the mobile module of the PARTNER Platform. This mobile module comprises a mobile framework developed in React to support the development of Android e IOS Self-Adaptive Internet of Health Things applications and mobile services. The API and services of the server module are compressed in the lower left part of Figure 2 to demonstrate the connection of the mobile module with the server module.

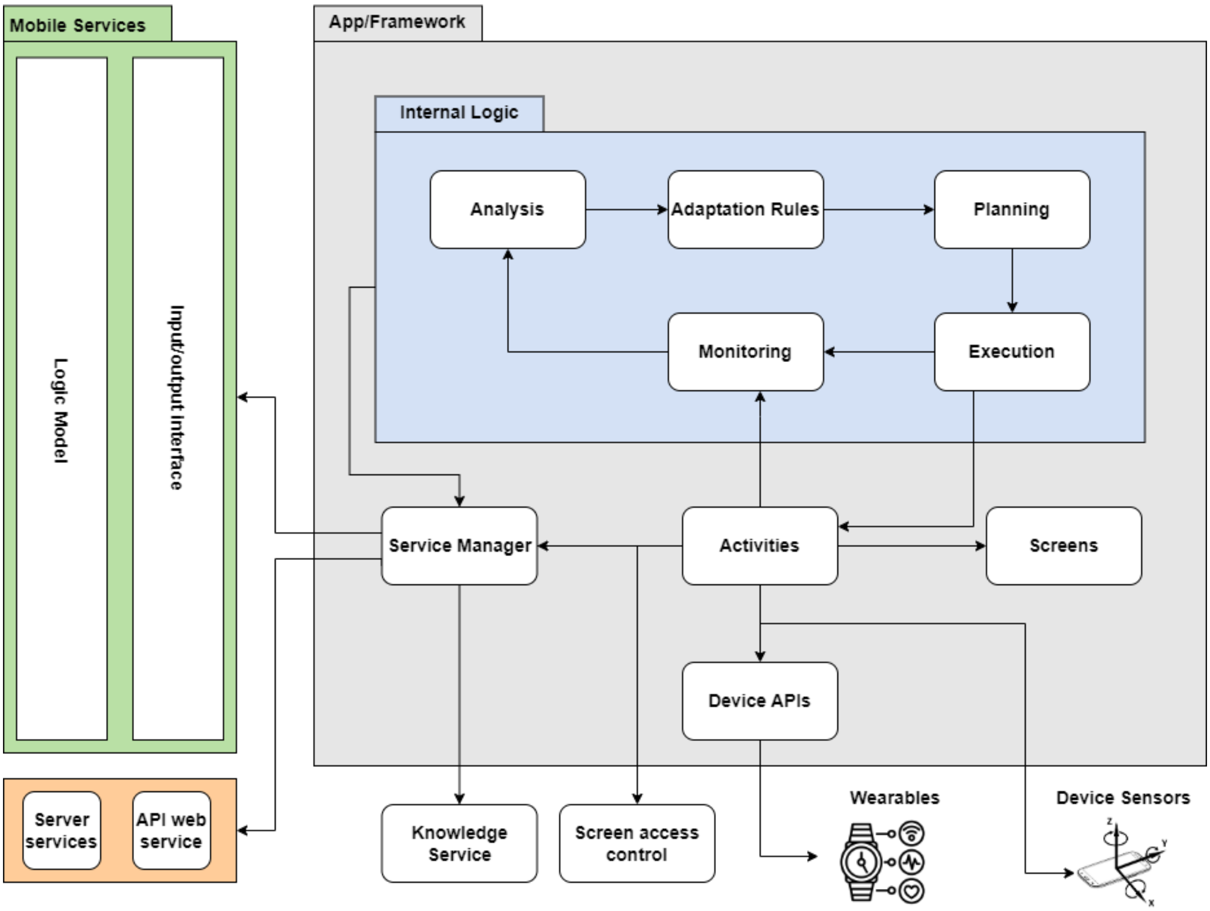


Figure 2: PARTNER's Mobile Module Architecture

Framework

Our framework provides facilities for developing self-adaptive IoHT applications based on microservices. It was based on the KREATION [12] IoHT application development framework, but it has some differences from the KREATION framework. First, our framework was developed using React Native, while KREATION was built in Kotlin. The reason for this choice was to provide a tool for developing cross-platform mobile applications. Thus, with our framework, it is possible to develop mobile applications for Android and IOS operating systems. The second difference between our framework and KREATION framework is that it provides a service management mechanism, allowing developers to create applications based on microservices architecture.

As a React Native framework, it follows the common practical componentization of each reusable element. Furthermore, as it is aimed at developing mobile applications, it uses as a base the concept of *Activities* and *Fragments* classes to control navigation logic and screen elements. From the *Activities*, it is possible to access various components and model classes that control the application's internal logic based on the MAPE-K adaptation loop proposed by IBM [14]. This internal logic must run in the background while the application is in use. The *Activities* in the framework also have internal components adapted to facilitate data collection and sensor management on smartphones, including error handling to ensure that applications can be run on different devices, even if they do not have all the sensors used by the application. Thus, an application can be designed to present more or less functionality depending on the availability of the device's sensors.

For each stage of the adaptation loop (Monitoring, Analysis, Planning, and Execution), a series of pre-implemented components are already offered so that the developer only needs to make the minimum adaptations necessary for the requirements and business rules of the application he is developing. Still, two more changes in the KREATION framework about the adaptation cycle exist. First, the adaptation rules mechanism is now a separate component that interacts with the Analysis and Planning phases, rather than being represented solely as methods within the main classes. Second, the knowledge component has been transformed into a service, enhancing its reusability. In this way, the same knowledge representation object can be used by different applications. Each stage of the adaptation loop has a main class or component, and communication between them is done using the `Observer` pattern in the same way as in the KREATION framework.

In addition to the components integrated into the application's internal logic, the framework contains a service manager component that can be accessed through *Activities*. The service manager is the component responsible for managing access to mobile services available on the same device that the application is running, such as access to the functionality of another app, for example, to share files. Furthermore, this component is also responsible for managing access to web services and can behave in two ways: in the first case, it is possible to access the service directly using a request to an IP address to access the service or by accessing the Web API which manages access to services and returns a response to the request made by the application.

It is also important to highlight the framework's third-party device API access component. With this component is possible to manage access to APIs such as GoogleFit [1] and HealthKit [2], which allows you to access health data from wearable devices on Android and IOS devices respectively. Thus, our framework allows us to develop applications that collect health-related data from wearable devices from different companies, as long as they allow data sharing through these APIs — common functionality among most wearable devices currently on the market.

Finally, about the logic for manipulating data collected from sensors, the Monitoring and Analysis phases classes offer methods for data formatting and extracting sensor features. In the current version of the framework, components are developed to collect in real-time, in addition to health data from wearable devices, the following data from the smartphone's internal sensors: Accelerometer, Gyroscope, Magnetometer, Battery, Location, and Network. Furthermore, for data sets collected from accelerometers, gyroscopes, and magnetometers, there are ready-made components for extracting the following features: mean, maximum value, minimum value, standard deviation, kurtosis, skewness, and entropy. Developers utilizing the framework have the flexibility to customize its components for collecting additional data and extracting various features tailored to the specific requirements of the application being developed.

Services

In addition to the framework, the mobile module of the PARTNER platform was also designed to contain a set of services that applications can access using the service manager component integrated into the framework. These services can be provided on the same device on which the application runs or are provided as web services.

In the current version of the mobile module of the PARTNER platform, two mobile services and two web services are available that can be accessed directly by the application using the service manager without needing a web API.

As mentioned above, one of the mobile services represents a service related to knowledge representation, currently made as a graph generated based on the GRAFIT [21] model that allows representing the relationship between sensors, features extracted from data collected by the sensors, classification algorithms, capable of classifying sets of features in different health situations, such as diseases, or health risk states. The second mobile service allows you to control access to the application screens, allowing different views of the application and its functionalities to be presented to different user profiles.

The platform currently also offers two web services. The first is a modeling logic service for building machine learning models, processing and analyzing data from these models, which currently has an artificial neural network, a decision tree algorithm, and a random forest algorithm implemented. These

[1] Google Fit is available at https://www. google.com.br/fit/
[2] HealthKit is available at https://developer.apple.com/documentation /healthkit

algorithms were trained with two public repositories for learning and identifying different types of movement based on accelerometer and gyroscope sensors. These services were developed using the Python 3 language and the Tensorflow [22], and SciKit-Learn [23] libraries, adapting the algorithms presented in [21]. The other web service provides access to reuse components for mobile device input and output interfaces.

5. Poof of Concept

We developed a proof of concept (PoC) to evaluate the functionality of the mobile module of the PARTNER platform. The IoHT self-adaptive application developed, called FRIEND, can monitor heart rate, steps, and calories through the information collected from a smartwatch or smart band sensors and identifies two movement risk situations, such as fall events, based on accelerometer and gyroscope sensors in the smartphone. FRIEND app has two different views: a Patient View, which monitors those above and presents alerts when heart rate reaches a defined threshold or when a risk situation is detected, and a Caregiver View, which presents information about one or more patients associated with the caregiver, that also receive the same alerts of the patient when the health risk situations aforementioned are identified for the application. This section will describe the PoC and how it was implemented using the proposed platform.

5.1. Proof of Concept Setup

The proof of concepts was implemented using Visual Studio Code IDE [3] in a notebook Acer with a Ryzen 5 processor and 16 GB of RAM. Also used is a Macbook Air m1 with 8 GB of RAM, for testing the version of the FRIEND application for IOS devices. The adaptation rules were built using a template proposed in [7].

We also utilized the Python code for classification graphs presented in [21] as the basis for constructing the knowledge and machine learning services used to identify fall situations in the FRIEND app.

5.2. FRIEND app

FRIEND is a self-adaptive IoHT application developed using React Native and the mobile module of the PARTNER platform. It seeks to monitor some aspects of patients' health and health risk situations. In addition, FRIEND allows caregivers to monitor their patients' information and be alerted when a health risk situation is detected.

FRIEND collects physiological data from the patient, in particular, the number of steps, calories, and exercises, through communication with the Google Fit API, so the application can be used in conjunction with smart bands and smartwatches from different brands that allow the data collected by these devices are sent to Google Fit. In addition to using physiological data obtained by wearables, FRIEND also collects data collected by the sensors of the smartphone where it is installed, such as battery data, type of network to which the device is connected, accelerometer, and gyroscope.

Based on accelerometer and gyroscope data, FRIEND uses a knowledge service based on trained machine learning algorithm models capable of identifying the occurrence of fall events with an accuracy rate above 70%. Based on the device's battery and network level (wifi or 3G, 4G or 5G mobile network), FRIEND can adapt its functionalities. For example, the alerts do not use a sound feature when the battery level is low (less than 50%). When connected to a mobile network, the application adapts by minimizing requests to Google Fit and the server. This prioritization allows for higher priority requests to the risk movement detection service, recognizing the heightened danger of such situations. Additionally, the patient's smartband or smartwatch enables automated detection of elevated heart rates.

The FRIEND application sends alerts to the patient and caregiver to inform them of health risk situations, allowing necessary actions for immediate care of the patient's health to be taken quickly. If

[3]Visual Studio Code available in https://code.visualstudio.com/

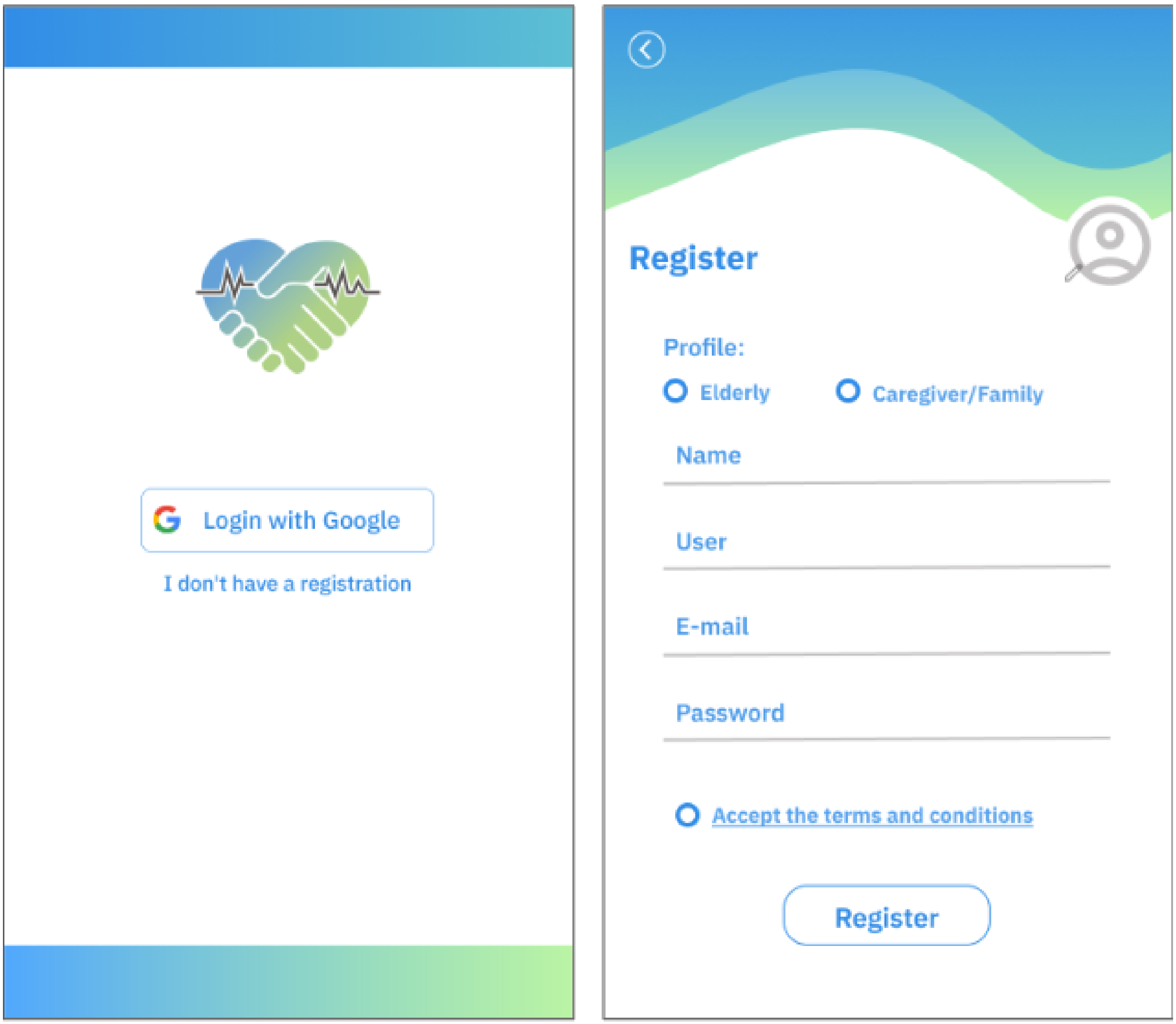


Figure 3: FRIEND app screens: Login and Register

the patient's heart rate is too high (above 120 bpm) or low (below 50 bpm), or if the FRIEND detects the occurrence of fall events, a text and audible alarm are sent to the patient and the caregiver.

In addition to alerts, the FRIEND application allows the patient to monitor their daily step count, number of calories burned during the day, and heart rate in real time. You can also see an average of this information throughout the day and week. You can also view the record of fall events. Furthermore, if a fall event occurs and is not automatically detected by the FRIEND app, the patient can manually report the incident, triggering an alert to notify the caregiver.

Figure 3 illustrates two screens from the FRIEND application: the login screen is shown on the left, and the registration screen is on the right. Users log in to FRIEND using the Google service to access Google Fit for collecting physiological data from wearable devices (such as smart bands or smartwatches). Furthermore, the more minimalist visual choice took into account login screens from other apps on the market. If the person does not have a registration when trying to log in, they are automatically redirected to the registration screen, where they can register as a patient or caregiver. The user can also go directly to the registration screen by clicking the "I don't have a registration" link.

Patient View

Figure 4 presents a set of screens visible to the patient user. As soon as the patient user logs into the application, they are redirected to the main screen (the one on the left in the Figure), where it is possible

Figure 4: FRIEND application - Patient View: Main Screen, Vital Signs Record Screen and Fall Record Screen

to see the physiological data collected with each request made to the Google Fit API. The data update depends on the speed at which Google Fit updates, and the request varies from a few milliseconds to a minute depending on the network used; as we have already mentioned, the application adapts according to the internet network used by the device. In the first interaction after logging in with the FRIEND application, for security reasons, it will be necessary for the user to provide permission to the device's sensors used by FRIEND and access to Google Fit data. It is also worth noting that immediately after the patient logs in, a background process is also started to collect data from the gyroscope and accelerometer sensors and request the knowledge service to identify possible fall events automatically. Data is sent and requested to the knowledge service every 60 seconds.

The patient's vital signs screen is shown to the right of the main screen in Figure 4. It can be accessed by the second icon from left to right in the bottom menu. This screen displays the chronological recording of physiological data collected from the Google Fit API.

The falls recording screen, a significant component of the FRIEND application, is displayed to the right of the vital signs as shown in Figure 4. This screen plays a crucial role in patient safety. When a fall is registered, an alert is sent as a notification to the device, if the FRIEND application is in the background, or as a foreground alert box in the FRIEND application, if it is open. This event is then recorded and we can access the fall registration screen to consult the fall history. Moreover, if a fall occurs but is not automatically detected, the patient can manually register it by clicking the 'Fall Report' button on the screen. Upon registration, the fall incident is recorded, and an alert is promptly sent to both the patient and caregiver.

Caregiver View

Figure 5 shows the flow for adding a patient to a caregiver's list in the FRIEND app. First, on the profile screen, which can be accessed from the rightmost option in the bottom menu, the caregiver selects the list of patients, clicks on add a patient, and then enters the username of the patient they wish to add to their care. Then, in the patient application, a notification appears that the user cares about adding him as his patient. If the patient user confirms, the patient is permanently added to the caregiver user's list, who can now monitor that patient's information. Similar to the patient, caregivers can view the

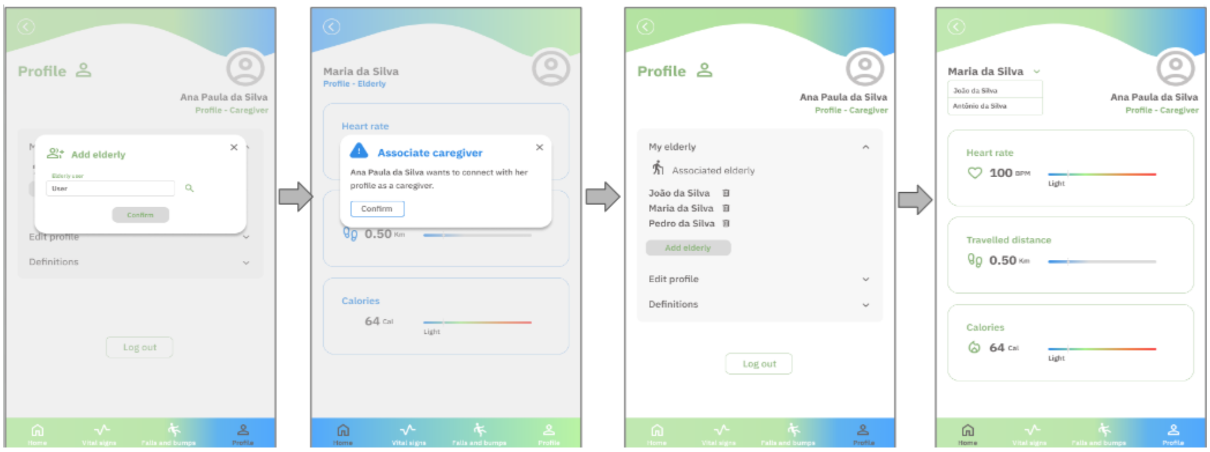


Figure 5: Flow for adding a patient to a caregiver's list at FRIEND

most recent values collected from Google Fit, as well as the records of vital signs and fall events of the patients they are associated with. Additionally, caregivers receive alerts when a health risk situation is identified for any of the patients under their care.

5.3. Practical Remarks

Once the system requirements were defined to implement the FRIEND application, our development team began to generate the system architecture based on the mobile module framework of the PARTNER platform. With the architecture and requirements, we begin development using the framework as a base and adapting the code to match the application's requirements and business rules. With the framework, it was possible to reuse all the logic of the adaptation cycle, from collecting data from wearable devices, using access components adapted to the Google Fit API, and collecting data from the application's sensors. We also use the service management component of the framework to integrate the Google login service. Additionally, using the service management component, we create a new Firebase service access component [4], for cloud data storage.

Utilizing the screen access control service within the mobile module of the PARTNER platform, we gained the ability to manage access to the various viewing modules of the FRIEND application. This allowed us to control the screens accessed and available functionalities based on the user, patient, or caregiver profile, enhancing the user experience and ensuring the relevance of the displayed information.

Furthermore, we adapted the knowledge service and modeling logic service available on the PARTNER platform to develop machine learning models. This adaptation enabled us to infer types of movements and identify fall events based on data collected from the smartphone's accelerometer and gyroscope sensors, showcasing the advanced capabilities of the FRIEND application.

The artifacts in the platform's mobile module facilitated the development of the backend of the entire application. Furthermore, some visual components in the framework were also reused to compose the screens. The most significant effort was to develop the front-end and specific visual elements designed for the FRIEND application. Finally, it is worth highlighting that when developing FRIEND, we not only verified the functionality of using the PARTNER platform for developing self-adaptive IoHT applications based on microservices but with the feedback obtained throughout the tests carried out with the application, it was possible to identify improvements that were incorporated, especially into the developed framework, linked to exception handling and optimizations for service requests.

[4]The Firebase cloud service is available at: https://firebase.google.com

6. Pratical Evaluation

To evaluate the perceived usefulness and ease of use of the FRIEND application developed using the PARTNER platform, an experiment was developed with elderly volunteers who used the FRIEND application for two weeks and presented their feedback and perceptions regarding its usefulness and ease of use. The collection and analysis of feedback was done using a questionnaire, following the Technology Acceptance Model (TAM) [15]. This section describes the details of how this evaluation was done and the results obtained.

6.1. Planning

The experiment was designed by adapting the experimental method proposed in [24] and evaluated based on the TAM [15]. First, we defined the objective of evaluating the ease of use and usefulness of the FRIEND application, and so the following research questions were developed for the evaluation:

- Is the FRIEND application useful in the User's perception?
- Is the FRIEND application easy to use in the user's perception?

To answer these questions, following the TAM model, we developed a questionnaire containing eight questions related to *Perceived Usefulness* (Table 1) and seven questions related to *Perceived Ease of Use* (Table 2). It is worth mentioning that the questions were prepared by a UI/UX specialist and evaluated by a specialist IoHT researcher who worked as a researcher and consultant during the platform project and proof of concept.

Table 1
Questions about Perceived Usefulness

	Question
QU1	I believe this application can help me effectively monitor the vital signs of the older adult in my care?
QU2	Do I see value in using this application to track older adults' vital signs compared to traditional monitoring methods?
QU3	I believe this application can help me identify health problems in older adults more quickly and efficiently?
QU4	I realize that this application can improve the quality of care I provide to the older adult?
QU5	Do I believe that using this application can contribute to better management of the health and well-being of the older adult?
QU6	Do I see clear advantages in using this application to track seniors' vital signs compared to other available alternatives?
QU7	Do I understand that this application can help me make more informed decisions about older adults' health and care?
QU8	I believe this application can provide me with more peace of mind and security when caring for the older adult?

The questions were added to a questionnaire developed using the Google Forms tool [5], with answers for each question following the Likert scale [25], which ranges from 1 to 5, where 1 represents total disagreement and 5 represents full agreement with the question.

The experiment was designed so that participants would use their own smartphones, where the FRIEND application was installed, and a smartband was provided to each participant in the experiment who acted as a patient for continuous collection of heart rate, step count and distance traveled data. The smartbands used in the experiment are Samsung Galaxy Fit 2, Samsung Galaxy Fit 3 and Amazfit Band 7 models.

[5] https:///www.google.com/forms

Table 2
Questions about Perceived Ease of Use

	Question
QF1	I think learning how to use this application was easy?
QF2	I feel this application is intuitive and easy to navigate?
QF3	I understand that this application has a clear and understandable interface?
QF4	I think it will be easy for me to view vital signs data in this application?
QF5	Do I realize that this application provides immediate and useful feedback on my actions?
QF6	Does this application provide useful and understandable feedback on vital signs data?
QF7	I trust that it would be easy for me to find help or support if I have difficulties using this application?

6.2. Execution

Before the experiment with the elderly, a pilot evaluation was carried out with three adult volunteer users. Each of these users received a different model of smartband. A presentation was then made about the application and installation of the FRIEND application on each of the volunteers' smartphones. The volunteers used the FRIEND application for a week and provided feedback that was essential for improving the application.

After the development and research team evaluated the feedback from the volunteers who participated in the pilot experiment, the application FRIEND went through a refinement phase and fixing bugs identified during the pilot experiment.

The experiment with the older adults volunteers was conducted over three weeks by a PhD researcher specializing in the area of IoHT who assisted in the development of the PARTNER platform and the FRIEND application.

Four women and one man participated in the experiment using the application from the patient's perspective, all over 65 years old. One man over 50 years old and two women over 65 years old participated in the experiment using the application from the caregiver's perspective. However, the two women who used the application from the caregiver's perspective also used the application from the patient's perspective. Thus, six different volunteers used the application during the experiment.

All users used an Android smartphone throughout the experiment, and one of them had a device that did not have an accelerometer. In this case, the FRIEND application had its behavior adapted so as not to use the automatic fall detection functionality, which requires this sensor. All other functionalities, including the option to manually report the occurrence of falls, were available to this volunteer.

Three of the female volunteers who participated in the experiment using FRIEND from the patient's perspective used Amazfit Band 7 smartbands, one female volunteer used the Samsung Galaxy Fit 3 model and the male volunteer who participated in the experiment from the patient's perspective used the Samsung Galaxy Fit 2 model.

In the first week of the experiment, a meeting was held with the volunteers to formally present the PARTNER platform project and the FRIEND application, as well as the entire design of the experiment to be conducted and its objective. The volunteers also signed the informed consent forms, in order to ensure that the participants in the experiment agreed to participate voluntarily in it.

At the beginning of the following week, the volunteers received the smartbands and the application was installed on their smartphones and configured for use by each participant in the experiment. From that moment on, the volunteers used the FRIEND application in conjunction with the smartbands for two weeks.After the two weeks, the final meeting was held and the volunteers answered the feedback questionaire.

At the beginning of the experiment, a contact group was created with the participants via WhatsApp. In addition, the researcher who monitored the experiment also registered all the participants who used the FRIEND app as a patient, as their caregiver. Thus, through occasional contact via the WhatsApp group and monitoring of data as a caregiver, it was possible to verify the use of the application.

6.3. Results

Figure 6a presents the results of the feedback from the volunteers who acted as patients for the questions regarding perceived usefulness. We noted that 80% agreed or strongly agreed with all the information that was questioned for usefulness, demonstrating that, in general, patient users agree that the application is useful. Only one patient preferred to say that he neither agrees nor disagrees regarding the usefulness of the application. Thus, we believe that, in the perception of elderly patients, in general, the application can be considered useful.

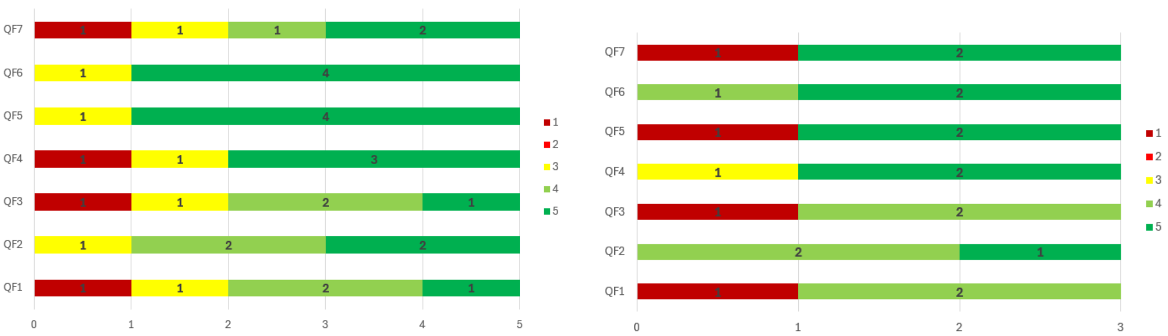
Figure 6: Feedbacks for perceived usefulnes

(a) Patient feedback for perceived usefulness (b) Caregiver feedback for perceived usefulness

Figure 6b presents the results of the feedback from volunteers who worked with caregivers for the questions regarding perceived usefulness. We noticed that there were more disagreements among the caregivers about the usefulness of FRIEND. In particular, for questions QU1, QU3, QU5 and QU8, one of the caregivers completely disagreed. For questions QU1, QU3 and QU8, the other caregivers agreed with the information, however for question QU5 there was a stalemate, as one caregiver agreed, one neither agreed nor disagreed and the other caregiver completely disagreed. Even so, we noticed that in general, for questions QU7 and QU8, most caregivers agreed on the usefulness of the application, which demonstrates that the application proved useful for caregivers, but there are still points for improvement, mainly regarding the speed of feedback and the way the data is presented to caregivers.

Figure 7a presents the results of the feedback from the volunteers who acted as patients for the questions regarding perceived ease of use. We noticed that for questions QF1, QF3, QF4 and QF7, one of the patient users completely disagreed. For the other questions, this user agreed. There was one user who chose neither to agree nor disagree with any of the questions. On the other hand, most users agreed with all the questions. This shows that, in general, most users believe that the application was easy to use. However, there are points for improvement that need to be considered, especially regarding the visualization of some data.

Figure 7: Feedbacks for perceived ease of use



(a) Patient feedback for perceived ease of use (b) Caregiver feedback for perceived ease of use

Figure 7b presents the results of the feedback from the volunteers who acted as caregivers for the questions regarding perceived ease of use. One of the caregivers completely disagreed with questions QF1, QF3, QF5 and QF7. We emphasize that this caregiver was not the same one who acted as patient and caregiver in the experiment. Regarding the other questions, this caregiver agreed. The other caregivers agreed with 7 of the eight questions, only in question QF4, one caregiver chose neither to agree nor disagree. The results show that, similar to what we observed with patients, in general, most caregivers believe that the application was easy to use, but that there are points for improvement that need to be considered.

6.4. Discussion

The results of the experiment showed that the FRIEND application, developed with the help of the PARTNER platform, proved to be useful and easy to use for most patients and caregivers. However, there are improvements that need to be made so that the app achieves higher quality and provides an even better user experience.

It is worth noting that installing the application is not so trivial and may take some time, since in addition to the FRIEND app, which is easy to configure in the opinion of the experiment participants, it is necessary to install other applications to collect data from the smartbands, such as Google Fit, and this may have impacted the perception of ease of use by the experiment participants, as reported by them to the researcher who conducted the experiment.

Furthermore, we emphasize that the questions were designed to assess both the patient's and caregiver's experience regarding the perceived usefulness and ease of use of the application developed using the platform. Since some steps of using the platform are similar in both views (patient and caregiver), some of the questions are similar for both the patient and the caregiver, which may have generated this impression of similarity in relation to some items.

We believe that the results of the experiment reinforce that the FRIEND application developed proved to be adequate for its purpose as a Proof of Concept, and that, in fact, the PARTNER platform is capable of assisting the development of self-adaptive IoHT applications with quality microservices architecture.

7. Conclusion

Internet of Health Things applications are increasingly becoming sources of research. However, several challenges must be considered when developing IoHT applications. Seeking to assist developers when creating IoHT systems, in this work, we proposed PARTNER, a practical and efficient platform for developing self-adaptive IoHT applications that use microservices. This platform seeks to provide the developer with a series of reuse artifacts that can facilitate the development of IoHT applications and mitigate the challenges inherent in developing this type of application.

In this research, we present Partner's mobile module, whose main artifact is a framework developed in Reactive Native, capable of assisting the development of self-adaptive IoHT applications and with a component that allows different services to be incorporated into the system developed, whether these services are mobile or web services. The framework also has components that use application APIs to gather health data from different wearable devices. In addition to the framework, this mobile module also consists of services for analyzing and formatting a knowledge representation of data based on the GRAFIT model proposed in [21] and a service for controlling access to screens of a mobile application.

To validate the platform, a proof of concept was conducted, during which the FRIEND application was developed. FRIEND is a self-adaptive IoHT application that monitors patient health data collected from wearables and automatically detects fall events based on accelerometer and gyroscope data from a smartphone. The usefulness and ease of use of the FRIEND application was evaluated in an experiment with elderly volunteers and in the perception of these volunteers, FRIEND proved to be useful and easy to use.

The PARTNER platform was conceived as part of a research and development project with limited time, so it was not possible to conduct a more in-depth evaluation of the use of PARTNER by other developers at this time. Therefore, we chose to focus first on an evaluation using a PoC and collecting feedback from volunteers who used this application developed with the help of PARTNER within the project period.

As future research, we intend to develop the web module of the PARTNER platform and evaluate the platform as a whole with the help of developers with experience in the IoHT area. Additionally, we plan to compare artifacts proposed to support IoHT application development from the literature with those offered by the PARTNER platform through controlled experiments.

Acknowledgments

We acknowledge that the development of the PARTNER platform was made possible through a collaborative effort and research project. It was developed as results of an R&D project funded by the startup GoTest and Funcap - Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico. This work is partially supported by INES (www.ines.org.br), CNPq grant 465614/2014-0, FACEPE grants APQ-0399-1.03/17 and APQ/0388-1.03/14, CAPES grant 88887.136410/2017-00.

References

[1] P. A. M. de Oliveira, R. M. Andrade, P. d. A. S. Neto, B. S. Oliveira, Internet of health things for quality of life: Open challenges based on a systematic literature mapping., in: HEALTHINF, 2022, pp. 397–404.

[2] J. J. Rodrigues, D. B. D. R. Segundo, H. A. Junqueira, M. H. Sabino, R. M. Prince, J. Al-Muhtadi, V. H. C. De Albuquerque, Enabling technologies for the internet of health things, Ieee Access 6 (2018) 13129–13141.

[3] J. Leng, Z. Lin, P. Wang, An implementation of an internet of things system for smart hospitals, in: 2020 IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI), IEEE, 2020, pp. 254–255.

[4] C. A. Costa, C. F. Pasluosta, B. Eskofier, D. B. da Silva, R. da Rosa Righi, Internet of health things: Toward intelligent vital signs monitoring in hospital wards, Artificial intelligence in medicine 89 (2018) 61–69.

[5] S. Zeadally, F. Siddiqui, Z. Baig, A. Ibrahim, Smart healthcare: Challenges and potential solutions using internet of things (iot) and big data analytics, PSU research review 4 (2020) 149–168.

[6] S. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, K.-S. Kwak, The internet of things for health care: a comprehensive survey, IEEE access 3 (2015) 678–708.

[7] E. C. Junior, R. M. de Castro Andrade, L. S. Rocha, Development process for self-adaptive applications of the internet of health things based on movement patterns, in: 2021 IEEE 9th International Conference on Healthcare Informatics (ICHI), IEEE, 2021, pp. 437–438.

[8] P. A. M. Oliveira, E. C. Junior, R. M. Andrade, I. S. Santos, P. A. S. Neto, Ten years of ehealth discussions on stack overflow (2022).

[9] R. M. Andrade, B. R. A. Junior, P. A. M. Oliveira, M. E. Maia, W. Viana, T. P. Nogueira, Multifaceted infrastructure for self-adaptive iot systems, Information and Software Technology 132 (2021) 106505.

[10] B. R. Junior, R. M. Andrade, M. E. Maia, T. P. Nogueira, Succeed: Support mechanism for creating and executing workflows for decoupled sas in iot, in: 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), volume 2, IEEE, 2018, pp. 738–743.

[11] M. Salehie, L. Tahvildari, Self-adaptive software: Landscape and research challenges, ACM transactions on autonomous and adaptive systems (TAAS) 4 (2009) 1–42.

[12] E. C. Junior, R. M. de Castro Andrade, L. S. Rocha, Kreation: Kotlin framework for self-adaptive ioht applications, in: 2023 IEEE 11th International Conference on Serious Games and Applications for Health (SeGAH), IEEE, 2023, pp. 1–8.

[13] A. Burger, C. Cichiwskyj, S. Schmeißer, G. Schiele, The elastic internet of things-a platform

for self-integrating and self-adaptive iot-systems with support for embedded adaptive hardware, Future Generation Computer Systems 113 (2020) 607–619.
[14] A. Computing, et al., An architectural blueprint for autonomic computing, IBM White Paper 31 (2006) 1–6.
[15] P. Silva, Davis' technology acceptance model (tam)(1989), Information seeking behavior and technology adoption: Theories and trends (2015) 205–219.
[16] R. Balaji, K. Bhavsar, B. Bhowmick, M. BS, K. Chakravarty, D. Chatterjee, A. Ghose, P. Gupta, D. Jaiswal, S. Kimbahune, et al., A framework for pervasive and ubiquitous geriatric monitoring, in: Human Aspects of IT for the Aged Population. Applications in Health, Assistance, and Entertainment: 4th International Conference, ITAP 2018, Held as Part of HCI International 2018, Las Vegas, NV, USA, July 15–20, 2018, Proceedings, Part II 4, Springer, 2018, pp. 205–230.
[17] L. Syed, S. Jabeen, S. Manimala, A. Alsaeedi, Smart healthcare framework for ambient assisted living using iomt and big data analytics techniques, Future Generation Computer Systems 101 (2019) 136–151.
[18] A. Dresch, D. P. Lacerda, J. A. V. Antunes Jr, A. Dresch, D. P. Lacerda, J. A. V. Antunes, Design science research, Springer, 2015.
[19] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, S. Linkman, Systematic literature reviews in software engineering–a systematic literature review, Information and software technology 51 (2009) 7–15.
[20] K. Schwaber, J. Sutherland, The scrum guide, Scrum Alliance 21 (2011) 1–38.
[21] E. C. Junior, R. M. D. C. Andrade, L. S. Rocha, Classification graph to the internet of health things applications, in: 2023 IEEE 11th International Conference on Healthcare Informatics (ICHI), IEEE, 2023, pp. 118–127.
[22] R. David, J. Duke, A. Jain, V. Janapa Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, T. Wang, et al., Tensorflow lite micro: Embedded machine learning for tinyml systems, Proceedings of Machine Learning and Systems 3 (2021) 800–811.
[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, the Journal of machine Learning research 12 (2011) 2825–2830.
[24] C. Wholin, P. Runeson, M. Host, M. C. Ohlsson, B. Regnell, A. Wesslén, Experimentation in software engineering: an introduction, Massachusetts: Kluwer Academic Publishers (2000).
[25] T. Nemoto, D. Beglar, Likert-scale questionnaires, in: JALT 2013 conference proceedings, volume 108, 2014, pp. 1–6.