Internet of Things 23 (2023) 100900

Contents lists available at ScienceDirect

Internet of Things

journal homepage: www.sciencedirect.com/journal/internet-of-things

Validation of a development methodology and tool for IoT-based systems through a case study for visually impaired people

Gleiston Guerrero-Ulloa [a,b], Ariel Fernández-Loor [a], Francisco Moreira [a], Paulo Novais [c], Carlos Rodríguez-Domínguez [b,*], Miguel J. Hornos [b]

[a] Faculty of Engineering Sciences, State Technical University of Quevedo, Quevedo, Ecuador
[b] Software Engineering Dept., Research Centre for Information and Communication Technologies (CITIC-UGR), University of Granada, Granada, Spain
[c] Department of Informatics, School of Engineering, University of Minho, Braga, Portugal

ARTICLE INFO

Keywords:
Internet of Things (IoT)
Development methodology
Test-driven development
Obstacle detection
Obstacle identification
Visually impaired people

ABSTRACT

In this article, we validate the Test-Driven Development Methodology for Internet of Things (IoT)-based Systems (TDDM4IoTS) and its companion tool, called Test-Driven Development Tool for IoT-based Systems (TDDT4IoTS). TDDM4IoTS consists of 11 stages, including activities ranging from system requirements gathering to system maintenance. To evaluate the effectiveness of TDDM4IoTS and TDDT4IoTS, in the last four academic years from 2019, System Engineering students have developed several IoT-based systems as part of their training, from the sixth semester (third academic year). Ñawi (phonetically, Gnawi), which is the case study presented herein, is one of them, and intends to assist visually impaired people to move through open environments. Ñawi consists of a device, a mobile application and a web application. The device interacts with the environment and issues alerts to the user whenever it recognizes obstacles in their path. The mobile application targets two user roles: assisted person and caregiver. Assisted people can use the device and log in into a server when they leave home, so that the mobile application identifies and notifies obstacles in their path. All the collected data is gathered into the server, so that caregivers receive notifications and can monitor the location of their assisted people at any place and time. The web application allows caregivers to query and view more extensive information (details of events, trajectories, etc.). TDDM4IoTS has been evaluated regarding both the roles of the project members and the development cycle stages. A survey was used to evaluate the methodology. Out of a total of 47 respondents, 30 had used TDDM4IoTS and 96.66% of them were very satisfied or satisfied, with nobody unsatisfied.

1. Introduction

The Internet of Things (IoT) is becoming the technological support paradigm to assist people in different aspects of daily life and in an increasing number of domains or application areas, such as smart homes or buildings, healthcare, education, transportation, agriculture, etc. Therefore, the need to have an adequate development methodology for this technological paradigm arises. A review on the status of development methodologies for IoT-based Systems (IoTSs) [1] concludes that the Test-Driven Development

* Corresponding author.
E-mail address: carlosrodriguez@ugr.es (C. Rodríguez-Domínguez).

https://doi.org/10.1016/j.iot.2023.100900

Available online 4 August 2023

Methodology for IoT-based Systems (TDDM4IoTS) [2] is one of the methodologies that most adequately fulfils their life cycle. Moreover, its support tool, called Test-Driven Development Tool for IoT-based Systems (TDDT4IoTS) [3], intends to integrate the development of IoT devices (IoT hardware and its configuration) and the development of mobile and web applications, which will be used for interaction between the user and the system.

In this paper, we present the outcomes of a validation of TDDM4IoTS and TDDT4IoTS, performed by developers with due experience in IoTS development. The validation of both (methodology and support tool) has been carried out through their application to a case study, which consists of the development of an IoTS named Ñawi (a word from the Quechua dialect that means Eye). Its main objective is to assist and give greater confidence to People with Orientation and Visual Problems (POVP), particularly visually impaired people, while they move around open environments. At the same time, Ñawi tries to alleviate the concern of their caregivers. Ñawi can detect obstacles, identify them, and notify the assisted person through sound and voice in the headphones, as well as the caregivers through a web and/or mobile application developed for them.

Ñawi is one of the most relevant IoTSs developed with our methodology and tool, and, consequently, it has been chosen as the case study to present in this article. However, the validation study of the methodology and tool has been much broader. In fact, this paper also presents the results obtained from the surveys carried out on System Engineering students from the academic year 2019–2020, who have developed other IoTSs as part of their training from the third year of their training, including as a final degree project.

The motivation and importance of using Ñawi as the case study chosen to be presented in this paper comes from that, according to the World Health Organization [4], 2.2 billion people suffer from impaired near or distant vision, of which 88.4 million suffer from moderate or severe impairment of distant vision or blindness. These statistics indicate that, if we consider that the world population in 2022 was 8 billion, 9 out of every 100 people suffer from some degree of visual impairment. This organization determines that more than 90% of people who suffer from blindness live in developing countries. With the use of technology, particularly IoT, the difficulties that this group of people may have in mobilizing themselves can be somewhat alleviated.

POVP have mobility and orientation difficulties that make them vulnerable. This is also a matter of concern for their family members and caregivers, who are responsible for their care and monitoring. To help reduce the concern of their relatives and keep them informed of events that may occur in the navigation of POVP under their care, in this paper, we present the development of Ñawi. This system can help POVP to move with greater freedom and confidence by being able to be alerted at a safe distance when they encounter an obstacle in their way.

During the development of the IoTS corresponding to our case study, which includes the implementation of two (web and mobile) applications, the guidelines for an inclusive design, dictated by the World Wide Web Consortium (W3C), were considered. W3C is an organization that leads the development of accessibility standards for web and mobile applications [5]. These standards allow people with different needs to use applications specifically designed for them at any time [6–8]. Assistive technologies also care for people with special needs. In fact, they provide devices that seek to improve the lifestyle of people with disabilities, and, together with IoT and W3C standards, they increase the participation of these people in society [6].

Therefore, the present work makes 3 main contributions: (1) a prototype of an IoTS, called Ñawi, is presented as a solution developed with low-cost components to help POVP in their mobility; (2) the validation of TDDM4IoTS as a development methodology that covers all stages of the software/system life cycle; and (3) the validation of the functionality of the TDDT4IoTS development tool, which integrates both the design and configuration of the IoT hardware, and the development of both mobile and web applications, as these are the main artifacts for users to interact with IoT devices. The remainder of the paper is organized as follows: Section 2 presents the related work. Section 3 describes the development methodology and the results obtained by applying each of its stages to the development of the IoTS proposed as a solution to our case study. Section 4 validates the development methodology used. Finally, Section 5 outlines the conclusions and future work.

2. Related work

Technological advances in different fields, such as sensors/actuators, communication networks, software engineering, artificial intelligence, and human-machine interaction, among others, have made IoT a new paradigm of computing systems, which allows many people to have access to supportive solutions to manage their daily life, thus improving their lifestyle [9,10]. IoT is also characterized by enabling the interaction between multiple heterogeneous devices, making it possible to provide ubiquitous services without human intervention [11]. The elderly and people with disabilities, or even chronic patients, are among the target beneficiaries of these services. In fact, many IoTSs have been specifically proposed as solutions to problems that specially affect to these people, such as assistance to people who live alone [9,10,12–14], and orientation problems for the elderly and/or people with visual impairment [15–17], among others. However, it is also true that the works reviewed do not cover all the needs of these people [18].

In order to create technological solutions that help people with disabilities to function without the help of other people, technology must make up for the disability present in them. In this case, the aim is to help people POVP, so the main objective for using technology is the identification and recognition of objects.

There are some technologies for the identification and recognition of objects, such as machine learning and deep learning. Of these technologies, machine learning is the ideal one to be executed as part of a mobile application since it requires less hardware resources. Machine learning technologies (MLTs) include Google Cloud Vision (GCV), Microsoft Azure Cognitive Services and the Clarifai General Model (CGM). GCV has been used in object recognition in several contexts, such as in hospitals, food recognition, obstacles in the way, to name a few [19]. In the work carried out by Karagiannis et al. [20], they used this MLT in image recognition, to determine matches between foods and supplements that were about to be consumed with programmed medications.

GCV-based solutions have been presented to help visually impaired people recognising obstacles along a trajectory. Among those that can be mentioned are the works carried out by Berger et al. [21], Akhil et al. [22], Rajendran et al. [23], and Goncalves and Paiva [24], which present very good results. GCV makes it easy for developers to understand image content without worrying about training machine learning models, which are provided by the API via an easy-to-use REST API [21].

In the work of Berger et al. [21], GCV successfully recognized more than 75% of the objects that appeared in the user's path. Akhil et al. [22] used it to reach user's destination, so that the system gives them an alternate path whenever an obstacle is encountered. In the same line, Rajendran et al. [23] implement a system to guide visually impaired people in their journey, consisting of glasses with ultrasound sensors that detect and recognize obstacles.

In the system presented by Goncalves and Paiva [24], the user must focus with the cell phone camera to capture the image, which is processed for recognition. These authors orient their system to the recognition of locations, logos and store names. Akhil et al. [22] mention that the response time to find an alternative path is due to the type of internet connection. However, they do not document the success rate with respect to object recognition. Both papers together with Rajendran's, do not mention the percentage of accuracy in object recognition.

A similar system to that of Rajendran et al. [23] is the one proposed by Ali A. et al. [25], although they differ basically in the way the system receives the command when the photo should be taken, in addition to the computer vision API they use. In the system by Ali A. et al. [25] it does so by receiving a voice command, and the API used is Microsoft Azure Custom Vision (customized with images of Indian environments), which achieves 73.1% accuracy.

Other concerns of some researchers include providing necessary assistance for POVP to navigate independently. Among the works found, there are systems with similar objectives to the one proposed in this work, such as devices that help these people navigate open environments [15–17,26–29]. These systems aim to improve the independence and confidence of POVP while also alleviating the concerns of their caregivers. To achieve their goals, authors have used different technological resources, such as sensors, simple-board microcontrollers, LED diodes, infrared, ultrasound, GPS (Global Positioning System), Bluetooth, headphones, and activation and/or notification buttons, to name a few.

Based on one of the first artefacts used by people with visual problems to achieve a greater independence in their movements, i.e., the cane, some authors [16,17,29] have developed a cane with technological components so that this artefact fulfils additional functions. Thus, Mala et al. [16] use it to inform assisted people about their current location using Bluetooth headsets, while Saquib et al. [17] present a cane, called BlinDar, to share the location of the assisted person in the cloud; however, its authors do not specify for what or how they use such information. Likewise, Vineeth et al. [29] also use GPS to obtain the location of the assisted person and send their location through a GSM (Global System for Mobile communications) module. This function is performed only when the assisted person presses a button and not when the system has detected a problem. The detected obstacles are notified to the user by means of physical vibrations of increasing intensity, depending on the proximity to the obstacle. Furthermore, the system presented by Vineeth et al. [29] issues a kind of orders or commands (stop, go down, go up, etc.) to its user.

Another indicator that has been considered for the development of this type of device is the maximum distance at which it will be able to detect the obstacle. This distance is different in each of the reviewed works. In fact, it is 10 m in Mala et al. [16] work, 2 m in Saquib et al. [17] works, and 4 m in Vineeth et al. [29] work. The Ñawi system not only detects the obstacles, but also recognizes them when objects are about 2 m away.

Each of the above-mentioned projects seeks to create IoT-based devices to help POVP. However, none of the proposed devices offer assisted people' family members or caregivers a mechanism to notify them of possible mishaps that may occur to the assisted people under their care. It also does not have a web and/or mobile application to set notification preferences, view the assisted person location history, and manage both assisted person and device information.

Regarding development methodology, we have used TDDM4IoTS to test its effectiveness in the development of IoTSs, since it is the only methodology that covers the whole life cycle of systems/software, according to the review carried out by [1], which considers the ISO/IEC/IEEE 15289:2019 standard [30]. This standard specifies the processes and procedures that are to be applied throughout the life cycle of a system or software product and provides a common framework for managing the life cycle of systems and software, including planning, implementation, assessment, and management of systems and software projects.

3. Development methodology and results of its application to the case study

TDDM4IoTS [2] involves 11 phases or stages to develop an IoTS: (1) Preliminary Analysis, (2) Technology Layer Design, (3) Detailed Requirements Analysis, (4) Model Generation and Adaptation, (5) Test Generation, (6) Software Generation, (7) Model Refinement, (8) Software Refinement, (9) Hardware and Software Deployment, (10) Deliverable Assessment, and (11) Maintenance. It is a test-driven development methodology specifically intended to develop IoTSs, considering their aspects and including the implementation of the values and principles of agile methodologies [31]. It also uses the definitions of the Scrum framework, such as

Table 1
Project team members and their roles for our case study.

Team members	Role
First author	Project facilitator
Second author	Counselor
Three System Engineering undergraduates	Development team
Visually impaired volunteer and non-visually impaired volunteer	Client/End user

the list of Sprints, and iterative and incremental development, as well as recommends use cases instead of user stories, and regular and on-demand meetings, among other aspects.

The roles played by each of the team members who have developed the project corresponding to the case study addressed in this article are shown in Table 1.

3.1. *Preliminary analysis*

The feasibility study took a short time because the small size of the project corresponding to this case study. We immediately determined its *economic and technical feasibility,* since it was a low-cost project that only needed cheap and easy-to-acquire devices. Likewise, we determined its *operational feasibility*, as it was a simple handling system, with a few easy-to-execute instructions, and the caregiver will be able to assemble and disassemble the device without any problem.

3.1.1. *Analysis of requirements*

POVP find it difficult to move in public places, and many of them do not do so out of fear or embarrassment. An easy-to-use and low-intrusive system is required to help these people feel safer to walk. The more independent it makes them the better the system will be. The assisted people must be notified when s/he is at a safe distance from an obstacle, so that s/he can avoid it. Moreover, caregivers need to monitor the displacement of the assisted people under their care and know their location at any time. In addition, caregivers must be informed of the obstacles encountered by the visually impaired people under their care, especially when patients have a fall or collision with one of them. Therefore, caregivers must be informed at any time, especially about the complications and unforeseen events that may arise when their assisted people move. Fig. 1 shows the use case diagram created for this case study by using the TDDT4IoTS tool [3].

For the success of the system, it is essential to consider its execution environment, as well as the fulfilment of its functional requirements (i.e., client needs). Upon detecting a crash or fall of the assisted person, the device should send a message to the caregiver. The caregiver can view the notifications in both the web application and the mobile app. The activities of identifying the functional and non-functional requirements of the system were done before planning the activities that involve the construction of the system, to achieve a successful planning. The planning of the tasks/deliverables to be performed to achieve the established objectives is shown in Table 2. The priority reflects the importance of the activity/deliverable for both the product owner and the development team.

3.1.2. *Environment analysis*

The device will be deployed in such a way that POVP can carry it with them during their daily walks (it is intended to help them to be more independent in their walks). As it is designed to be used in open environments, moments and/or places with varying levels of noise and lighting, as well as the presence of objects with which they can collide are expected.

The transmission of data from the device to the storage server for further processing must be carried out through a wireless connection and maintained while the assisted person remains away from home. The photographs of the obstacles captured by the device and the user location data, as well as the alerts and notifications must be distributed among the components of Ñawi system, i.e., the device and the (mobile and web) applications. Caregivers mainly need to know the current situation of their respective assisted people at any time, though they may be also interested in knowing the history of what happened to the people under their care in their daily walk. Therefore, photographs of obstacles that have caused accidents to the assisted person are securely stored remotely for subsequent consultation only by the corresponding caregivers. Thus, privacy is preserved. The remaining photographs are discarded

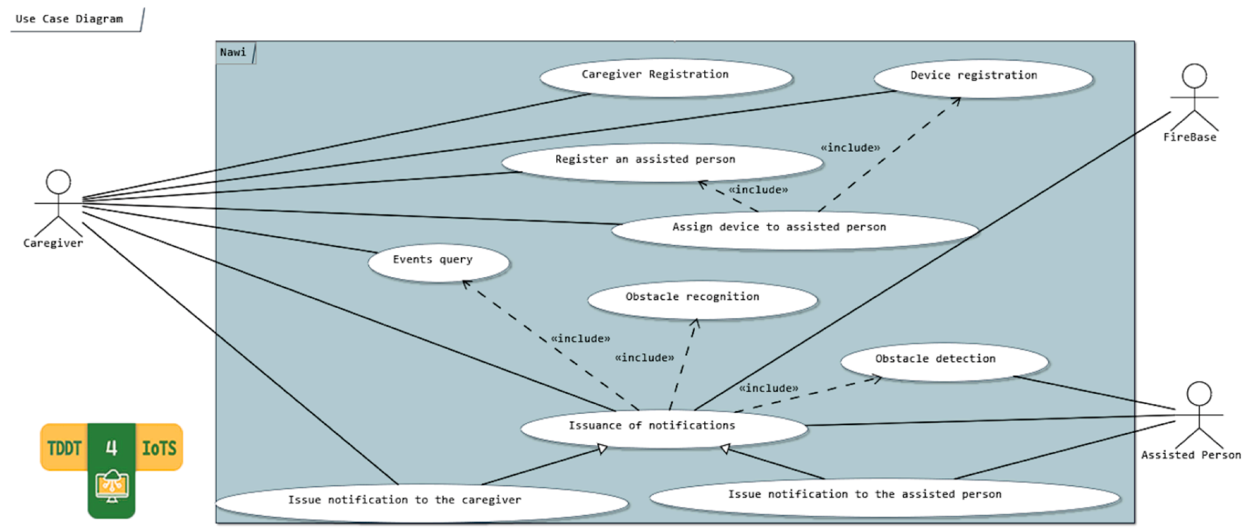


Fig. 1. Use case diagram for Ñawi.

Table 2
List of tasks/deliverables planned.

Tasks/deliverables to be performed	Hours	Priority
Writing the tests with the client	12	High
System architecture design	8	High
Device architecture specification	12	Medium
Web application architecture specification	4	Medium
Mobile application architecture specification	4	Medium
Definition of (internal) development standards	6	High
Back-end model design and implementation (database)	8	High
Construction of web services	16	High
Mobile Application Front-End Design and Implementation	40	Medium
Web application front-end design and implementation	24	Medium
Construction of the back-end of both (web and mobile) applications (business logic)	16	Medium
Implementation of obstacle detection	5	High
Algorithm training for object recognition	16	Medium
Implementation of object recognition	5	High
Configuration of the database on the remote server	1	Low
Implementation of the issuance of notifications	8	Medium
Integration of the application with arduino code	8	High
End device assembly	6	High
Communication implementation between back-end and front-end of the mobile app	5	High
Integration testing	8	Medium
System deployment	4	High
Deliverable assessment	1	High

immediately after processing them.

3.1.3. Technology analysis

The system will be deployed on an upper body garment. The power supply must be by means of a battery. The most accurate sensors when it comes to capturing data for the detection of obstacles in environments with the expected characteristics are the ultrasonic ones. To capture the image of the obstacle, a camera will be used.

As the device needs to send and receive data while the assisted person is on the move, the mobile internet service is adequate. Due to the speed required to send and receive data, 5 G technology would be the ideal one. However, this service is not currently available in all places, since there are regions where mobile internet service is provided only through 4 G, LTE, 3 G, and/or 2 G technologies. Response time in an application like Ñawi is very important. That is why we have used local processing, fog computing, and cloud computing [11] for object detection, object recognition, and alert issuance to user caregivers respectively, like the work of Rajavel et al. [32], with the difference that they do it when they recognize some unusual behaviour of the person. The use of a service such as Firebase would be necessary for the real-time distribution of the required data among the device and the (web and mobile) applications. The mobile application would provide a solution to the requirement of receiving notifications at any time, to know the status of the assisted person. However, since caregivers are also interested in knowing the history of what happened with the people under their care, they need a web application to view the information generated in a complete and more detailed way.

As there were several alternative options to choose the corresponding hardware components, development tools, methodologies, and programming languages, a list of different products were prepared together with the criteria to select the most suitable ones for the development of Ñawi (See Tables 3 and 4).

Regarding the selection of the hardware devices to be acquired and incorporated into our IoTS, the first criterion used was that the corresponding device had to meet the functional requirements, in addition to helping to meet the non-functional requirements to the

Table 3
Selection of the hardware components to be used.

Function	Selected device	Alternative devices (Among others)	Selection criteria
Taking photographs	ESP32-Cam	OV7670 VGA Module Lens 300KP	Availability, price, features [33].
Obstacle detection	HC-SR04 Ultrasonic Module	Infrared Detector Sensor Module Laser Sensors	Price, availability, fewer operational restrictions to operate in the environment [34].
Controlling sensors	NodeMCU ESP8266 12E Wi-Fi Development Board	ESP8266 ESP-01 Arduino UNO Arduino Mega Wi-Fi Module (required with all the above) Raspberry	Two in one with all the benefits. Smaller than Arduino UNO board and has built-in Wi-Fi [35].
Fall detection	GY-521 MPU6050 3-Axis Sensor Module	GY-61 ADXL337 3-Axis Accelerometer Sensor Module MPU9250 200 Hz WT-901Sensor Module	Price, availability, functionalities, experience in using it [36].

Table 4
Selection of software technologies and platforms to be used.

Function	Alternatives (Among others)	Selected software	Selection criteria
Web application development	C# Visual Basic .Net PHP Python	Java (JDK 11, JEE 8)	The fourth-best programming language [37], and one of the safest [38]. Moreover, it is the development language for mobile applications, which facilitates software reuse.
Real-time data sharing	Amazon Web Services Azure Heroku	Firebase	Firebase is a Google's mobile platform that helps to quickly develop high quality applications. Easy access for testing. Developer proficiency [39].
Users' data storage	MySQL MariaDB Firebird	PostgreSQL 13	The best database manager among free software relational database systems [40].
Implementation of object recognition	Java R Matlab	Python	The most popular and best scientific programming language, and the second most popular programming language overall [37].

greatest possible degree. The other criteria were availability, price, functionalities, size, and developer experience in its use. The list of hardware resources finally acquired along with the criteria that led us to select each of them is shown in Table 3.

Proposing an economic solution is one of the objectives of this work. That is why the use of open-source software was considered as the main criterion to select the development tools needed. The subsequent selection criteria were their functionalities or benefits, and the experience in their use by the development team, giving greater weight to this last criterion. The list of software resources selected together with the criteria that predominated in the choice of each of them is shown in Table 4.

3.1.4. *Feasibility analysis*

The acquisition of sufficient requirements was one of the first activities that were carried out to later perform the feasibility analysis. In case the results of the system feasibility study were positive, then we would proceed to work on the planning of its development. Otherwise, the project would be abandoned early (though, in a large project, this is not the only time in which the project can be abandoned). Therefore, following the TDDM4IoTS for the development of the proposed system, a feasibility study, considering the scope of the system to be developed, the number of people involved or interested in the project, and the technology to be used, among other indicators, was carried out from three different perspectives or points of view:

Technical Feasibility. Due to the great advancement of technology in recent decades, both the quality/price ratio and the supply/demand ratio of the technological resources necessary to develop an IoTS have increased. This allows an easy access to different technologies to develop almost any type of IoTSs at a relatively inexpensive cost. In addition, the development team is trained (its members are students in the ninth semester of the System Engineering degree and with 2 years of experience as IoTS developers),

Table 5
Code generation and editing tools.

Function	Alternatives (Among others)	Selected tools	Selection criteria
Design and development of the web application interface, and overall software refinement.	Eclipse 2021 IntelliJ IDEA	Apache NetBeans 12.4	Mastery of the tool [41].
Design and development of the mobile application interface.	Swift para iOS AIDE - IDE for Android IntelliJ IDEA Apache Netbeans	Android Studio 4.0	The application was developed on Android because the number of Android users exceeds the number of iOS users [9]. Android Studio is among the most popular worldwide [41,42]. In addition, our developers have more experience on Android.
System architecture modelling (applications and device). Device design, Arduino code generation. Automatic software generation (model, and access to data via web services).	Modeling tools StarUML GitMind Violet UML Editor Fritzing	TDDT4IoTS[1]	It allows designing IoT devices and, from the use cases, generating models (class diagrams), tests and code for both the classes and the user interface, as well as for the Arduino board. It also allows code to be burned into the Arduino board. It is the only tool that integrates device design and system modelling [3].
Image processing for web and mobile applications.	Photo Pos Pro Google Nik Collection	GIMP 2.10.12	Features, user experience by developers. The most popular among image editing tools [43].
Additional editor for writing source code for the different applications.	Sublime Text PyCharm PyDev Spyder IDLE	Visual Studio Code	One of the best programming code editors for multilanguage [44]. Mastery of the tool.

[1] Web application used for IoTS modelling and automatic code generation [3] that has been devised as a support tool for the TDDM4IoTS methodology [2].

motivated and involved to develop the system, both in terms of hardware and software. Professionals who play the roles of project facilitator and client or end user also have experience in IoTS development. Moreover, the development team is confident that it is possible to develop the system since there are many easily accessible resources and tools. In addition, the methodology is role-oriented, and project team members support each other. The list of code generation and editing tools used in the development of this IoTS is shown in Table 5, together with the criteria considered to select each one of them.

Economic Feasibility. As mentioned previously, the aim is to develop a low-cost IoTS so that both the project and its result are economically viable, even for people with few economic resources. As the different hardware components are low cost and the tools for the system development are free software, the economic feasibility of the project is guaranteed. The cost of the components to build the device amounts to approximately 32 US dollars, based on their price as of August 1, 2023.

Operational Feasibility. Both (web and mobile) applications are easy to maintain, due to being based on well-specified models. The web application will be deployed on a server (http://www.tddt4iots.tech/nawi) for testing this IoTS. It should be accessible from any browser, reason why developers should carefully check its correct functioning on the most popular browsers. For its operation, we should hire a hosting and buy a domain to publish the web application and, as part of it, include a link to download the mobile application or publish it in the Play Store. Moreover, the use of the device and the mobile application in areas with and without Internet connection was considered. Object recognition should also work when the device is not connected to the Internet, while the captured and generated data (for object identification) will be sent as soon as the smartphone reconnects to the Internet.

3.2. *Technology layer design*

For the design of the system architecture, which is shown in Fig. 2, both functional and non-functional requirements have been considered. As recommended by TDDM4IoTS, IoTS design tools can be used to more clearly represent the elements needed to develop the project.

The device components (i.e., sensors and an ESP32-Cam board) will be integrated into an upper body garment wearing by the assisted person. The device communicates with the mobile application, which uses the smartphone as a gateway to communicate with the server, to send and receive alerts. The sensors/actuators will be controlled by an ESP8266 NodeMCU board, which uses the assisted person's smartphone as a gateway. In this smartphone, a mobile application will be also installed so that obstacles in front of the visually impaired person can be identified.

The modern concept known as Cloud Computing makes it possible to offer various types of software services, including development tools that help users develop innovative applications quickly and efficiently [11]. In fact, the physical or hardware design of the IoT device was done using the TDDT4IoTS web application [3]. Fig. 3 shows the interconnection of the main devices used in the construction of the IoT device for the Ñawi system, which are the following ones: (a) ultrasonic sensors, (b) GY-521 MPU6050 3-axis sensor module, (c) ESP8266 development board, (d) ESP32-Cam, (e) buzzer, (f) resistor, and (g) battery.

The system automatically takes pictures whenever an obstacle is detected at 1.5 m and then every 50 cm if the POVP do not change direction on their way. Nevertheless, the assisted person can control whether to take a picture of the obstacle (to identify it when it is detected) or he/she takes the picture by pressing a built-in button for that function. When the mobile application identifies the object detected by the device, it generates the notification to the caregiver and the respective alert (voice and sound) to the person with visual impairment. The caregiver can view the event history via the web application and can also monitor the person under their care (more specifically, their current location, travel history, and events of the day) using the web application or the mobile application.

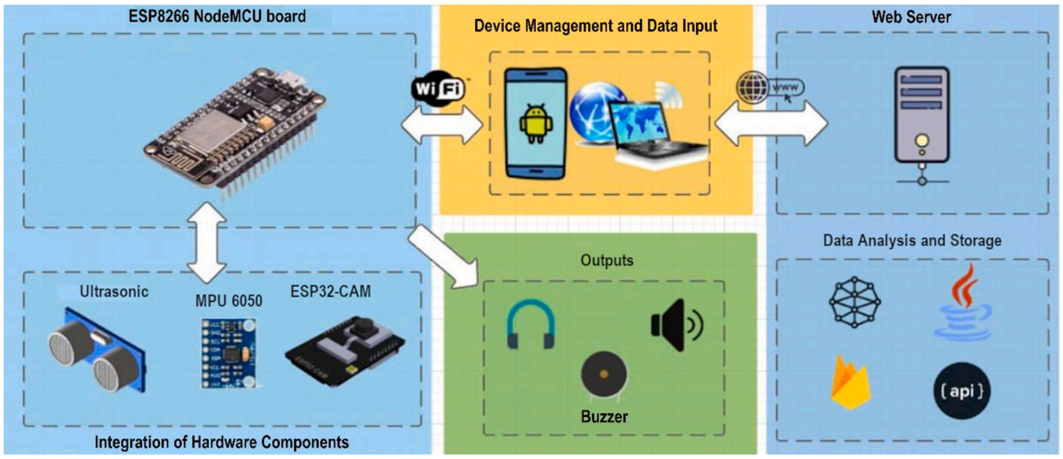

ESP8266 NodeMCU board
Device Management and Data Input
Web Server
WiFi
Ultrasonic
MPU 6050
ESP32-CAM
Outputs
Buzzer
Data Analysis and Storage
{api}
Integration of Hardware Components

Fig. 2. Architecture for Ñawi system.

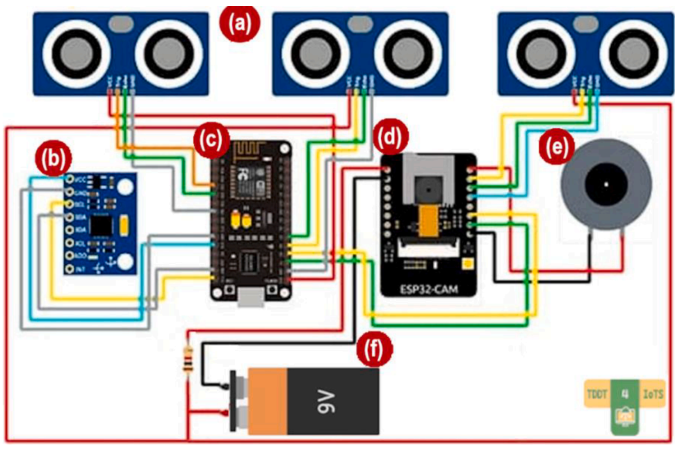


Fig. 3. IoT device hardware design.

3.3. *Detailed requirements analysis*

Use case diagrams (see Fig. 1) and detailed use cases have been used for the requirements analysis stage. In this phase, the TDDT4IoTS tool [3] was used to facilitate the work of developers. As an example, Tables 6 and 7 show the description of two of the use cases that have been implemented in the system.

3.4. *Model generation and adaptation*

For the generation of the model, the TDDT4IoTS tool [3] was used. TDDT4IoTS allows developers to create the use case diagram and description for each use case. In it, a language based on punctuation characters and other symbols to mark certain elements when writing use case descriptions is used by the developers. From these descriptions, the tool automatically generates the class diagram (shown in Fig. 4), the unit tests, and the Java software code. It also allows the creation of the sequence diagram (we are working to generate it automatically in the same way as the class diagram). Therefore, the model generated from the detailed use cases obtained in the client requirements elicitation phase is almost complete. In fact, developers just had to add a few attributes and methods to the automatically generated classes to finish completing the system model.

Table 6
Obstacle recognition use case description.

Use case: Obstacle recognition	
Actors:	User (assisted person)
Purpose:	Alerting the visually impaired person to obstacles in their path.
Type:	Primary
Pre-conditions:	The device connects to the end-user application.
Description:	Once the obstacle is detected, the device takes photos of it, and they are analysed by the mobile application. Thus, the obstacle is identified so that the visually impaired person knows which object is in their way.
Normal Flow:	
Author Actions	System Response
1. This use case starts when the device is connected to the mobile app.	
	2. The system receives the return time of the wave sent by the ultrasound sensor.
	3. The system calculates the distance at which an obstacle is located.
	4. If the distance is less than or equal to 1.5 m, the system takes a photo of the obstacle.
	5. The system processes the photo corresponding to the obstacle to recognise it, obtaining its name.
	6. The system transforms the obstacle name into a voice.
	7. The system reproduces the obstacle name in voice in the assisted person's headphones.
	8. This use case ends when the user is informed about the obstacle in front of him/her.
Alternative Flow:	
	2.1. The mobile application does not recognize the obstacle.
	2.2. The mobile app gets "unidentified obstacle" as the name of the obstacle.
	2.3. This case ends.
Post-conditions:	The device remains connected to the obstacle recognition application.
Relationship:	Include: Issuance of the notification

Table 7
Device registration use case description.

Use case: Device registration	
Actors:	Caregiver user
Purpose:	Register a device in the system.
Type:	Primary
Pre-conditions:	The user must be authenticated as a caregiver in the application.
Description:	The caregiver user must enter the device data, including its MAC address, which will make it possible for the device to be uniquely identified.
Normal Flow:	
Author Actions	System Response
1. This use case starts when the caregiver wants to register a device in the application.	
2. The caregiver accesses the device registration interface.	
	3. The system displays the fields to be filled in: ID, MAC address, name, and status.
4. The caregiver enters the device data requested by the system: ID	5. The system verifies that the device ID is not assigned to any device in the database.
6. The caregiver sends the MAC address of the device to be queried.	7. The system verifies that the MAC address of the device is not assigned to any device stored in the database.
8. The caregiver sends the data to be saved.	9. The system stores the data in the database.
	10. The use case ends when the system displays the interface with the list of registered devices.
Alternative Flow:	
	5.1. The ID of the device to be registered matches the ID of a device in the database.
	5.2. The system displays a feedback message indicating that said ID is already registered in the database.
	5.3. Go to step 10.
	7.1. The MAC address of the device to be registered matches the MAC address of a device in the database.
	7.2. The system displays a feedback message indicating that said MAC address is already registered in the database.
	7.3. Go to step 10.
Post-conditions:	The device is registered in the database.
Relationship:	Include: Data validation

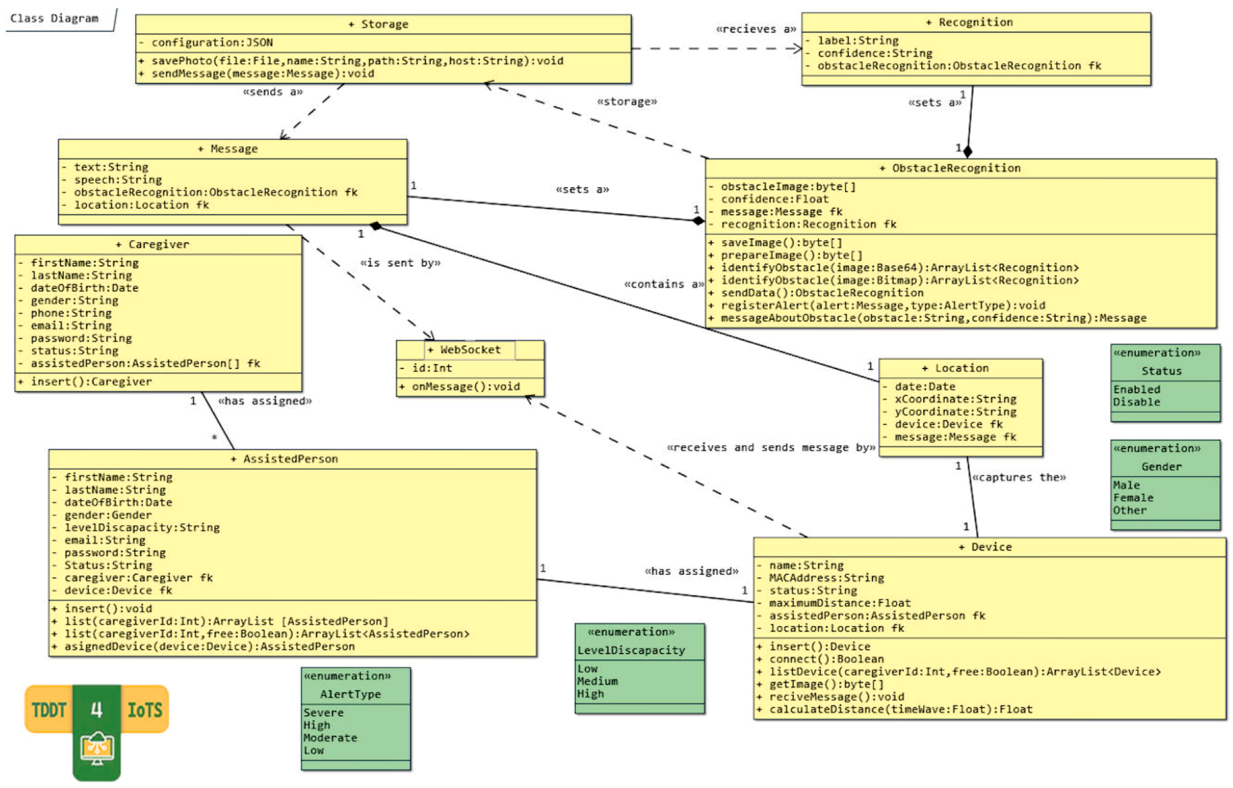
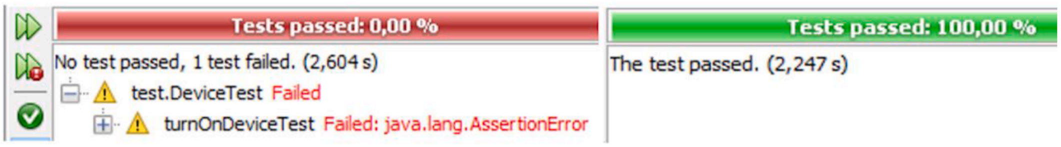


Fig. 4. Class diagram automatically generated by TDDT4IoTS.


Tests passed: 0,00 %
No test passed, 1 test failed. (2,604 s)
test.DeviceTest Failed
turnOnDeviceTest Failed: java.lang.AssertionError

Tests passed: 100,00 %
The test passed. (2,247 s)

Fig. 5. Captures of two results obtained from the test execution.

3.5. *Test generation*

In this phase, the tests to be carried out are generated from the detailed use cases of the system to be developed. Among the most frequently executed functional tests within the development lifecycle of a system or software are unit tests and integration tests. The unit tests are automatically generated by the TDDT4IoTS web application. These unit tests are downloaded to be executed in an IDE (Integrated Development Environment) or any other development environment that the developer prefers. Code Listing 1 presents an example of one of the unit tests generated to be performed, while Fig. 5 shows a couple of possible results of its execution. Integration testing is important when the solution has been developed with a divisional approach. Some of the integration tests used are shown in Table 8.

Another type of testing that becomes the most important at the time of product delivery to the client is acceptance testing. Acceptance tests were specified by setting up a simulated environment with the objects that are most frequently encountered in an open environment. In addition, different environments were set up, a simulated environment for a real user, and a real environment for a simulated user. The latter tests were executed in the deliverable assessment phase.

3.6. *Software generation*

The software generated by our tool [3] must be adjusted by adding the necessary code to meet the system objectives. It should be noted that 100% of the code generated by the tool was used in the development of the IoTS software. The tool generates a Maven project with Spring Boot. Code Listing 2 shows the code generated for the Device entity class of API web RESTful.

In the pom.xml file of the Maven project generated by our tool, the Lombok open-source library is added to help reducing the amount of boilerplate and tedious code snippets that must be written when developing Java applications. For example, when the *@Data* annotation is added to a class, Lombok will automatically generate *equals*(), *hashCode*(), *toString*(), as well as *getters* and *setters* for all the attributes of the class. This reduces the amount of code developers must write and maintain, making it more concise and readable [45].

Code Listing 3 shows the code of the class that allows implementing data persistence with Spring Boot, which is reduced to a simple interface that is derived from the JpaRepository class.

The interface derived from the JpaRepository class in Spring Boot simplifies access and manipulation of entities in relational databases. It offers methods for basic CRUD (Create, Read, Update and Delete) operations, without the need to write custom SQL queries. It also allows defining custom methods and supports pagination and result sorting. This interface streamlines the development of JPA applications by reducing repetitive code and improving database access efficiency [46].

Code Listing 4 presents a class implementation that enables REST web services in a Spring Boot application, allowing seamless operations and data manipulations through the REST API. Leveraging Spring Boot's annotations, this class defines the corresponding

Table 8
List of some of the integration tests used.

Test description	Method or function*	Evaluation case
Create a user	Caregiver insert (): Caregiver	The caregiver model object is sent with all the data to be saved. In case of success, the caregiver will receive a message in response and will be redirected to the login point.
Register an assisted person	AssistedPerson insert (): AssistedPerson	To register an assisted person the AssistedPerson form is sent with the data. In such a form, the ID of the caregiver or person in charge of the patient is included.
Power on the device	Device connect (): Boolean	Both the assisted person and the MAC address of the device (device class member data) are used. Then the method will try to connect the assisted person app to the device. If the connection is successful, the assisted person will receive data from each of the sensors integrated into the device.
Identify objects	ObstacleRecognition identifyObstacle (base64): ArrayList<Recognition> ObstacleRecognition identifyObstacle (bitmap): ArrayList<Recognition>	If the ESP32-Cam camera has taken the photo, the image will be sent in Base64 format and this image will be processed to be identified. But if the user uses their smartphone to take it, the image will be in Bitmap format. If the image sent is identified successful, it will be saved in the storage.
Register alerts	registerAlert (alert, type)	The alert registration is carried out with the data of detected obstacle sent by the sensors. When the alert registration is successful, the user will receive a notification.

* Class to which it belongs, method: Type of value returned.

Listing 1
Example of a unit test in Java programming language.

Listing 1. Example of a unit test in Java programming language.
```
@Test
    public void turnOnDeviceTest()
    {
     Device device = new Device("Device01", "ÑAWIV1.0","A2:13:12:BB:3C:43", false);
     DeviceDAO deviceDAO = new DeviceDAO(device);
     assertTrue(deviceDAO.getMessage(),
     deviceDAO.get(device.getMACAddress (), device.getAssistedPerson.getId()).length() > 0);
    }
```

Listing 2
Device entity class (Device.java) code in Java programming language.

Listing 2. Device entity class (Device.java) code in Java programming language.
```
package com.app.tddt4iots.entities;
    import com.app.tddt4iots.enums.*;
    import lombok.Data;
    import lombok.NoArgsConstructor;
    import javax.persistence.*;
    import java.io.Serializable;

    @Entity
    @Table(name = "Device")
    @Data
    @NoArgsConstructor
    @AllArgsConstructor
    public class Device implements Serializable {
     @Id
     @GeneratedValue(strategy = GenerationType.AUTO)
     private Long id;
     @Column(name = "name", nullable = false, unique = false, length = 30)
     private String name;
     @Column(name = "MACAddress", nullable = false, unique = false, length = 30)
     private String MACAddress;
     @Column(name = "status", nullable =  false, unique = false)
     private Status status;
     @Column(name = "maximumDistance", nullable = false, unique = false)
     private Float maximumDistance;
     @JoinColumn(name = "idAssistedPerson", referencedColumnName = "id")
     @OneToOne
     private AssistedPerson assistedPerson;
     @JoinColumn(name = "idLocation", referencedColumnName = "id")
     @OneToOne
     private Location location;
    }
```

Listing 3
Device data access class (DeviceDao.java).

Listing 3. Device data access class (DeviceDao.java).
```
package com.app.tddt4iots.dao;

    import com.app.tddt4iots.entities.Device;
    import org.springframework.data.jpa.repository.JpaRepository;

    public interface DeviceDao extends JpaRepository<Device, Long> {
    }
```

endpoints and uses Spring MVC's capabilities to handle HTTP requests and responses effectively. It showcases the streamlined development of efficient and user-friendly RESTful web services using Spring Boot, eliminating boilerplate code and ensuring robust API maintenance [47].

Listing 4
Web services class to manipulate device data (DeviceApi.java).

Listing 4. Web services class to manipulate device data (DeviceApi.java).

```
package com.app.tddt4iots.apis;
    import com.app.tddt4iots.entities.Device;
    import com.app.tddt4iots.dao.DeviceDao;
    import org.springframework.beans.factory.annotation.Autowired;
    import org.springframework.http.ResponseEntity;
    import org.springframework.web.bind.annotation.*;
    import java.util.List;
    import java.util.Optional;

    @RestController
    @RequestMapping("/device")
    public class DeviceApi implements RestController {
     @Autowired
     private DeviceDao deviceDAO;
     @GetMapping
     public ResponseEntity<List<Device>> getDevice() {
     List<Device> listDevice = deviceDAO.findAll();
     return ResponseEntity.ok(listDevice);
     }

     @PostMapping
     public ResponseEntity<Device> insertDevice(@RequestBody Device device) {
     Device newDevice = deviceDAO.save(device);
     return ResponseEntity.ok(newDevice);
     }

     @PutMapping
     public ResponseEntity<Device> updateDevice(@RequestBody Device device) {
     Device upDevice = deviceDAO.save(device);
     if (upDevice != null) {
     return ResponseEntity.ok(upDevice);
     } else {
     return ResponseEntity.notFound().build();
     }
     }

     @DeleteMapping(value = "{id}")
     public ResponseEntity<Device> deletePersons(@PathVariable("id") Long id) {
     deviceDAO.deleteById(id);
     return ResponseEntity.ok(null);
     }
     }
```

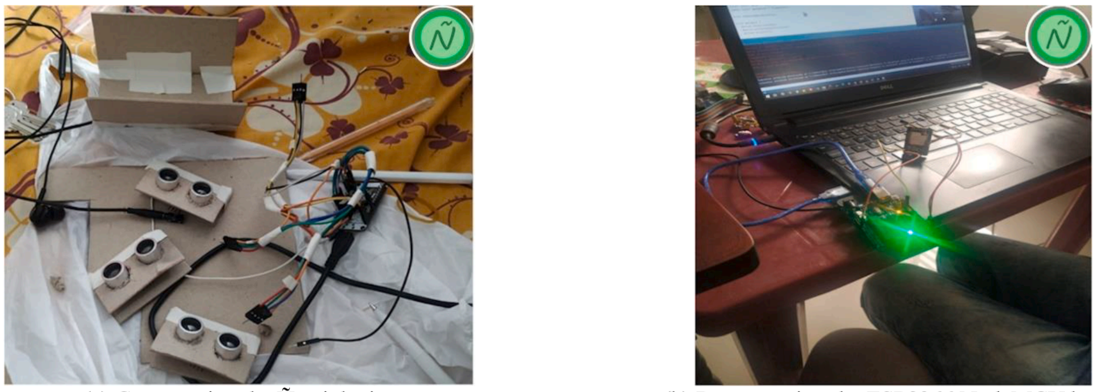

(a) Constructing the Ñawi device (b) Programming the ESP8266 NodeMCU board

Fig. 6. Working on the hardware components of the Ñawi device.

By annotating methods with HTTP operations like GET, POST, PUT, and DELETE, DeviceApi class seamlessly handles client requests and delivers appropriate responses. It processes incoming requests, executes essential business logic, and formats the data or results in commonly used formats, such as JSON or XML. Implementing the RestController interface enables the class to inherit

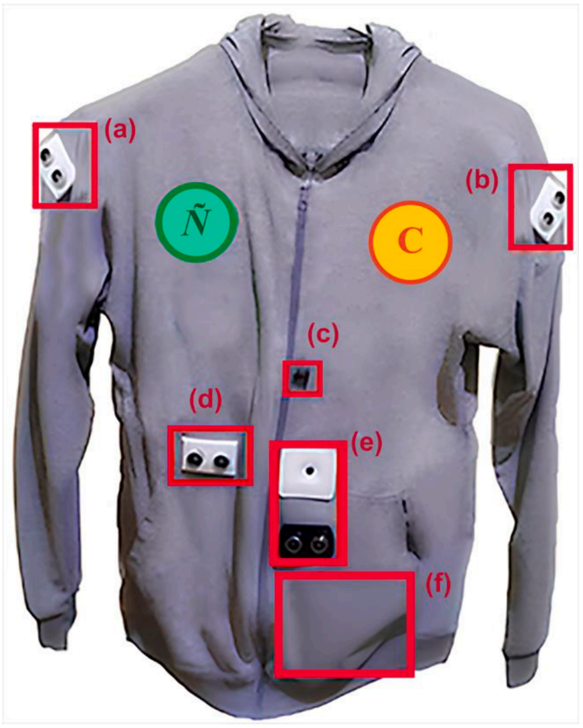


Fig. 7. Deployment of the hardware components on the Ñawi device.

essential behaviours and features needed for handling RESTful communications, further simplifying the API development and maintenance.

3.7. *Model refinement*

The model generated automatically by using the tool TDDT4IoTS must be refined to obtain the proper and complete model. Thus, for example, validation issues, such as methods and control attributes corresponding to security policies, should be added, with which the software is generated.

This refinement is extremely important. In fact, a very meticulous analysis must be carried out, checking that no functional requirements are left out, as well as that all non-functional requirements are met.

3.8. *Software refinement*

Just as the generated model must be improved, after automatically generating part of the software (classes and definition of their methods), said initial software must also be refined to add the necessary business logic for the system to behave as expected. In our case, when using the automatic code generation tool, we had to implement the business logic of many methods for the classes to fulfil their responsibilities.

The tool used, i.e., TDDT4IoTS, does not yet have the implementation of sufficient functionalities to generate an operational (web or mobile) application. Nonetheless, it is noteworthy that, although this tool is still in testing, the code it automatically generates is very useful and is fully used in the final version of the implemented system.

3.9. *Hardware and software deployment*

Part of the business logic was implemented using RESTful web services. Service-oriented architecture is certainly one of the solutions for communication in a heterogeneous system [48]. So far, both (web and mobile) applications consume these services. Both the web application and the PostgreSQL database are hosted on this same server. The program code that controls the device was developed in Arduino Studio and burned into the ESP8266 NodeMCU board. Fig. 6 shows some of the times when developers were working on the device. Thus, Fig. 6(a) shows a photo of when the developers were integrating the different hardware components to build the Ñawi device. Once this device was built, it was possible to test the Arduino code that was burned into the ESP8266 NodeMCU board to control the sensors and actuators. Fig. 6(b) shows the moment when that code was being burned into the ESP8266 NodeMCU board.

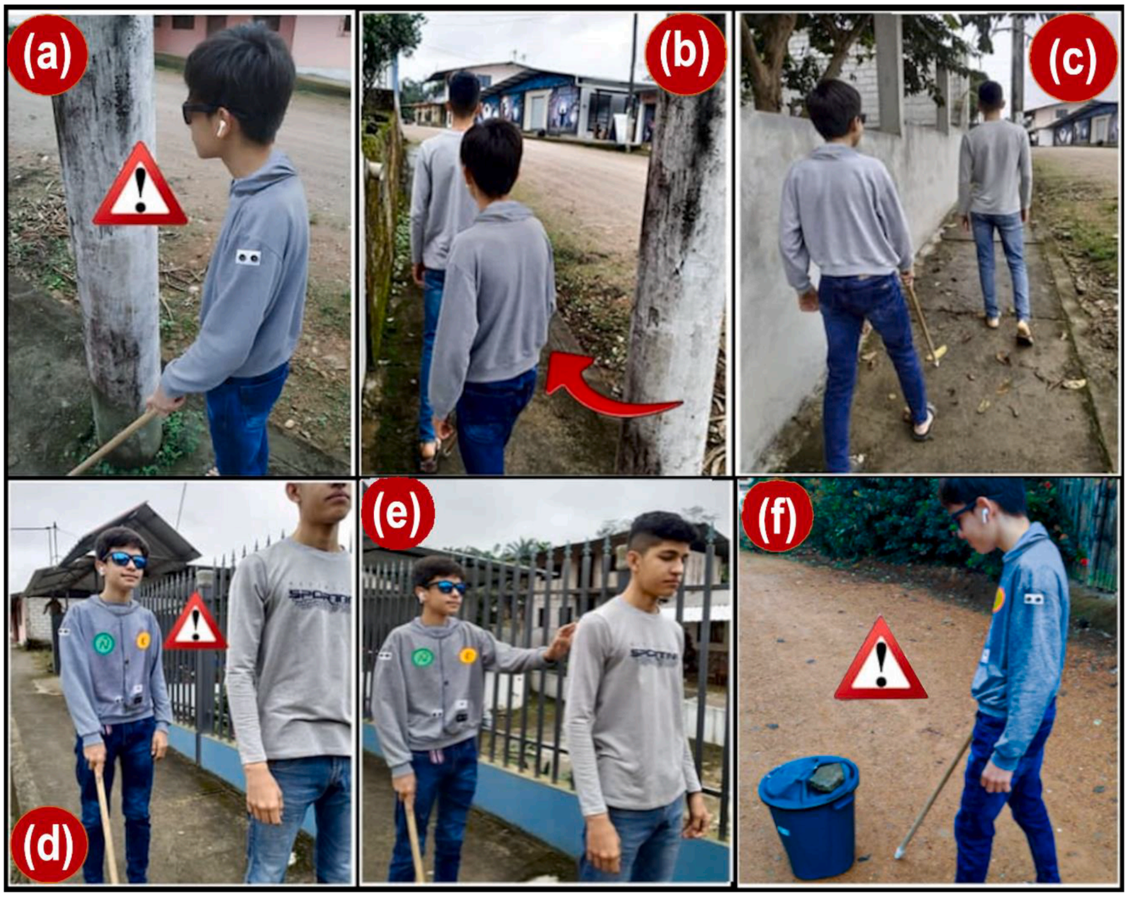
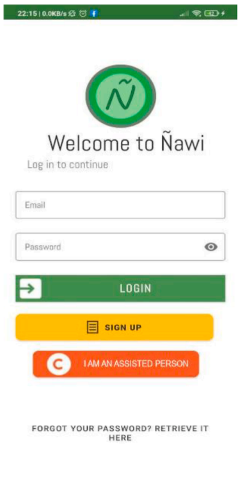
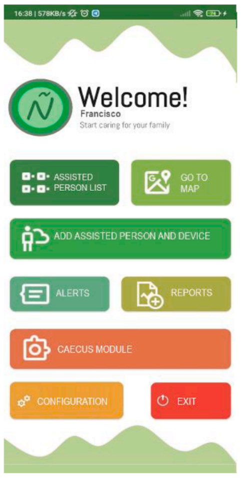


Fig. 8. Photographs of the blind person testing the Ñawi device.

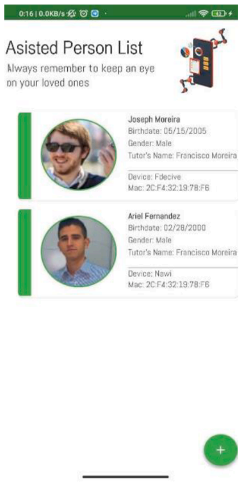
The hardware components that make up the Ñawi device were deployed on a zip-up hoodie, as shown in Fig. 7, though they could have been deployed in any other garment. On the sides, on the upper part of the sleeves, we have installed two ultrasonic sensors (a and b). On the front, we have inserted the button to activate the camera (c), other ultrasonic sensor (d), and the camera (e), while the ESP3286 NodeMCU board together with other elements (GY-521 MPU6050 3-Axis Sensor Module, battery, and resistances) are placed



(a) Login screen.



(b) Options for caregivers.

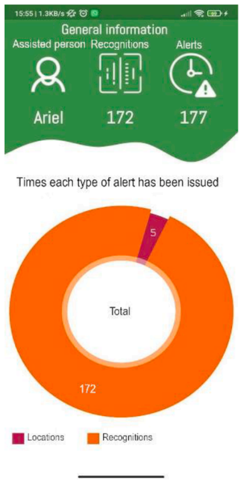

Asisted Person List

Always remember to keep an eye on your loved ones

Joseph Moreira
Birthdate: 05/15/2005
Gender: Male
Tutor's Name: Francisco Moreira
Device: Fdecive
Mac: 2C:F4:32:19:78:F6

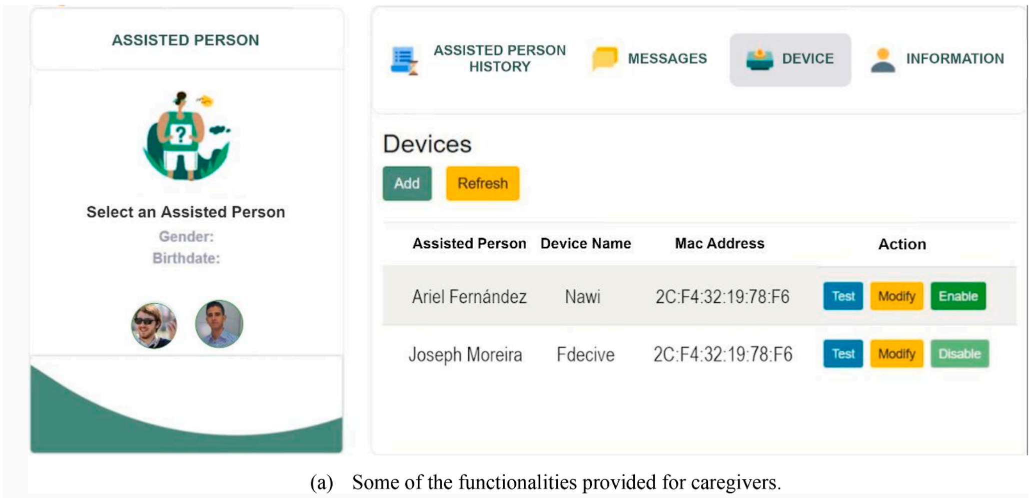
Ariel Fernandez
Birthdate: 02/28/2000
Gender: Male
Tutor's Name: Francisco Moreira
Device: Nawi
Mac: 2C:F4:32:19:78:F6

(c) Assisted person list.



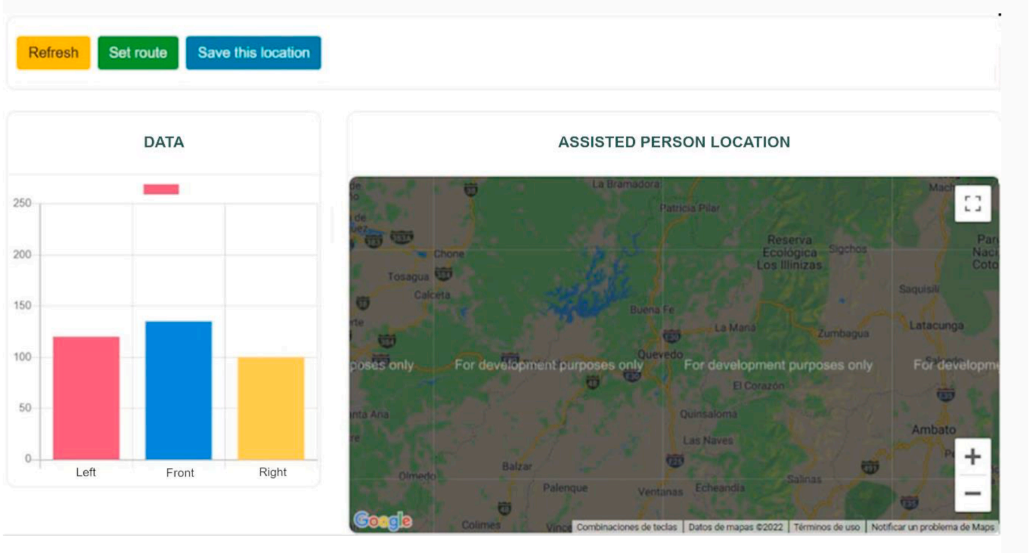
(d) Reports.

Fig. 9. Screenshots of the Ñawi mobile application.



Assisted Person	Device Name	Mac Address	Action
Ariel Fernández	Nawi	2C:F4:32:19:78:F6	Test Modify Enable
Joseph Moreira	Fdecive	2C:F4:32:19:78:F6	Test Modify Disable

(a) Some of the functionalities provided for caregivers.



(b) Data captured by the system in real time.

Fig. 10. Screenshots of some of the GUIs of the web application.

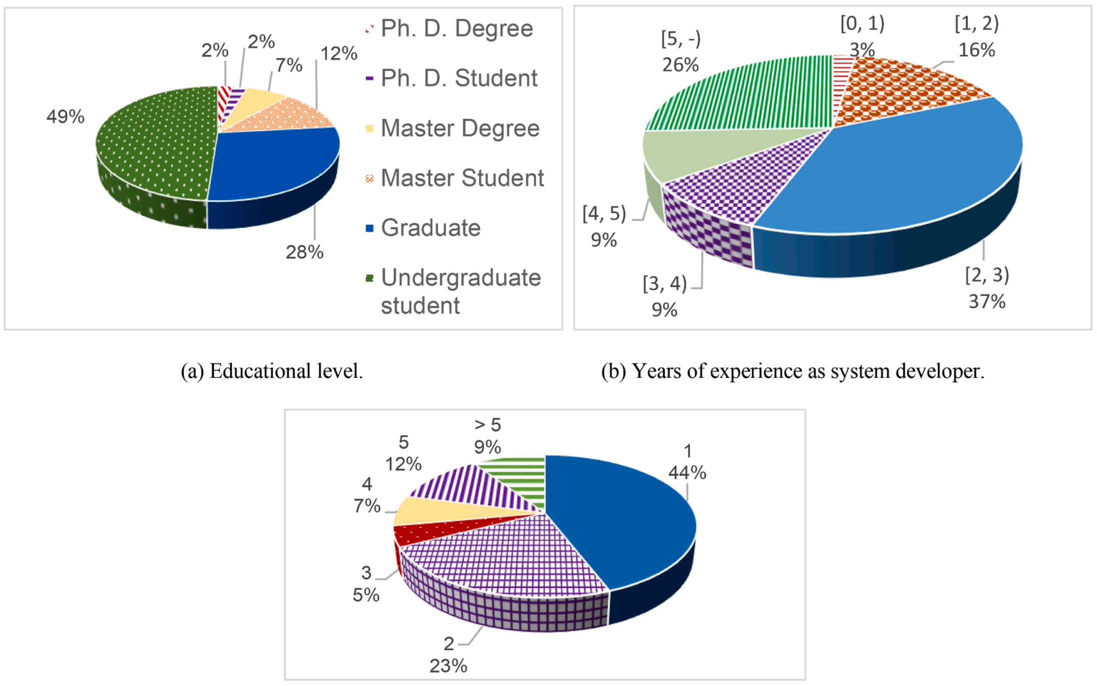
inside the hoodie pocket (f). The assisted person must carry their smartphone, in which the mobile application must be installed and running (with the role corresponding to assisted people).

3.10. Deliverable assessment

To evaluate Ñawi, three people with different characteristics were considered: a person with visual impairment (assisted person 1), a person without visual impairment pretending to be visually impaired (assisted person 2), and a person with basic skills in the use of a smartphone interested in monitoring the person who is under their care (caregiver). The two assisted people used the Ñawi device with the mobile application running with the assisted person role, while the caregiver evaluated the mobile app running with the caregiver role activated and the web application.

Assisted person 1 tested the Ñawi device in test scenarios that were fully controlled, as shown in Fig. 8. This blind person walked in an environment which was little known to him, in which there were permanent obstacles, such as electric lighting poles (a and b), a wall (c), and a container (f), to mention only a few. Likewise, moving people, animals or objects could be in his path, such as a person (d and e), a dog, a cat or a car, among others. Assisted person 2 (who was not visually impaired, just simulating) tested the Ñawi system in less controlled scenarios.

Both assisted people felt comfortable with the device deployed in the zip-up hoodie, considering that the country (Ecuador) where the tests were carried out has a warm climate. Therefore, we should analyse another alternative garment or means to deploy the device components for hotter or colder climates. Moreover, both assisted people stated that the mode in which the device components were deployed in the garment made it nothing intrusive, except for the components inserted in the pocket. We will try to place these

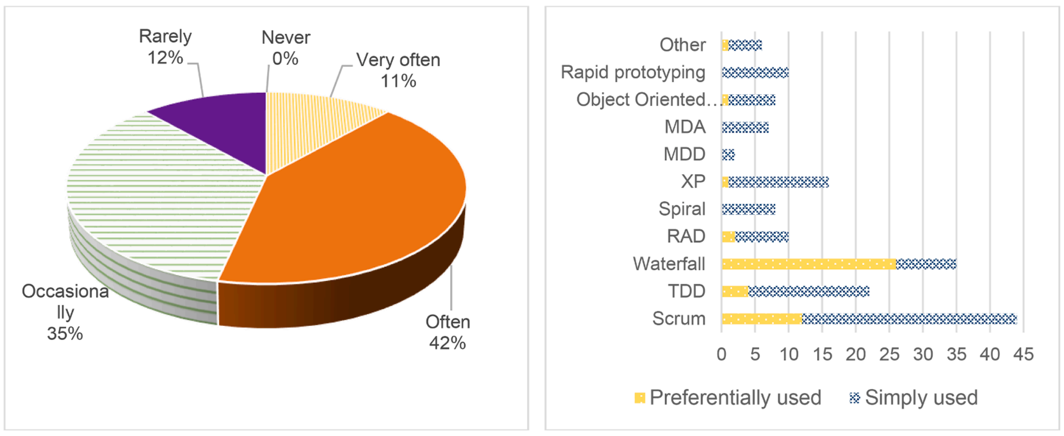


(a) Educational level.

(b) Years of experience as system developer.

(c) Number of IoTS development projects in which they have participated.

Fig. 11. Data on surveyed developers.



(a) Frequency of use of methodologies.

(b) Methodologies used.

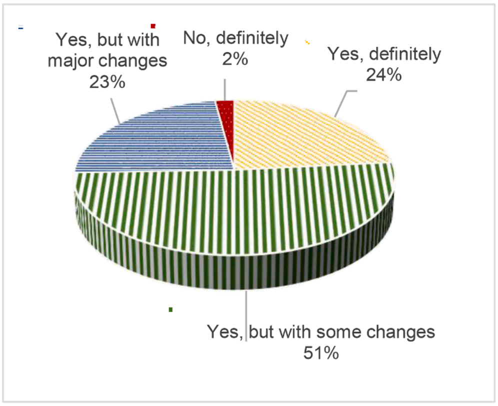
Fig. 12. Use of methodologies in software development.

components in a more adequate place in a future version of the device.

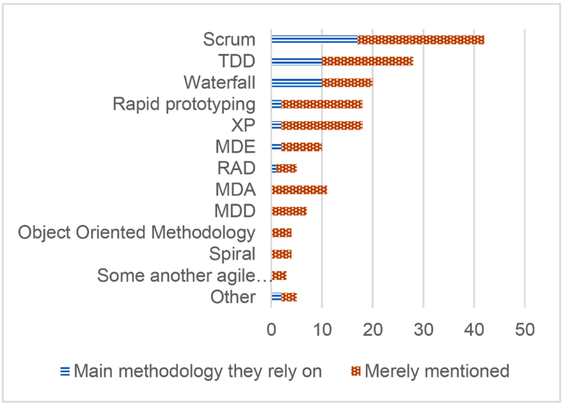
While assisted person 2 was moving through the tests with the Ñawi device, he was requested to fake falls in order to check the effectiveness of issuing notifications to the caregiver. It was verified that the caregiver received the notifications immediately, being able to know their assisted person location at that very moment. The average response time (from the taking of the photograph and the subsequent obstacle identification or from the fall until the caregiver receives the corresponding notification) was estimated to be less than 0.5 s. We think this delay is due to the quality of the mobile internet service in the city (and the region) in which Ñawi was tested.

Regarding the caregiver role, both (web and mobile) applications were evaluated by 14 people, of whom 8 were men and 6 women, aged between 15 and 50 years. All of them were very interested in the system, and expressed comments of acceptance and satisfaction, as well as that they would recommend its use.

Fig. 9 shows various screenshots corresponding to some of the graphical user interfaces (GUIs) of the mobile application of Ñawi. The texts in these GUIs will be displayed in Spanish or in English, at the choice of the user. Fig. 9(a) shows the login screen as a caregiver user, which requires entering an email address and the corresponding password. When the user has been authenticated, a



(a) Use of traditional methodologies.



(b) Traditional methodologies they rely on.

Fig. 13. Use of traditional methodologies for IoTSs development.

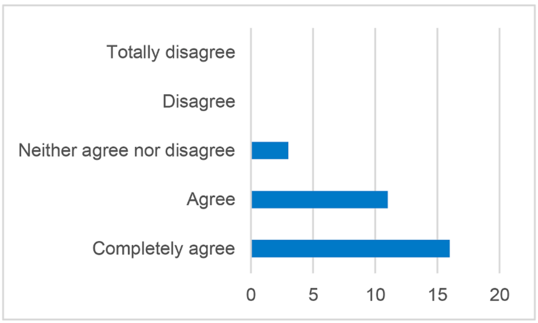
screen like the one shown in Fig. 9(b) is displayed. As can be observed, the main options available to the caregiver user are the following ones: list of the people under their care, view an assisted person on the map to know their location, add new assisted people and their corresponding device, view alerts, and get reports. The *Caecus module* option offers the chance to view the history of object identification. Moreover, the last two options allow the device configuration and exiting the application, respectively. Fig. 9(c) shows the list of assisted people for whom the logged-in caregiver user is responsible. And Fig. 9(d) shows an example of screen with graphical reports of certain information related to a given assisted person under the care of this caregiver.

Screenshots of some of the GUIs of the Ñawi web application are shown in Fig. 10. The Ñawi web application is accessed via the HTTPS (HyperText Transfer Protocol Secure) protocol and its users (i.e., caregivers) only have access to their own information and that of the people under their care. For this, as in the mobile application, they have to authenticate with an email address and a password. Passwords are stored encrypted in the database. Some of the input interfaces are developed based on modal windows, such as typical data entry interfaces (e.g., those corresponding to data on caregiver users, assisted person users, and devices, among others). Fig. 10(a) shows the web application screen with the options for the caregiver to view and enter information on assisted people and devices (as can be observed in this screenshot, to carry out the tests, both users used the same device). Fig. 10(b) shows the number of the objects detected by the three (left, front and right) ultrasonic sensors of the Ñawi device during the last trajectory, as well as the location of the assisted person in real time.

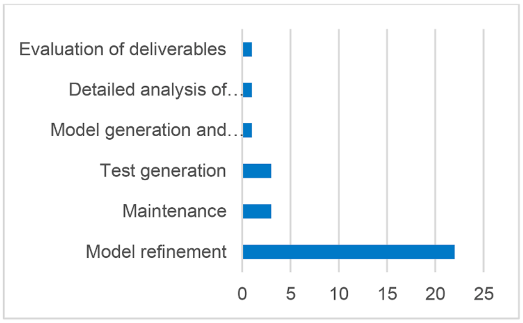
3.11. Maintenance

It is necessary to perform a periodic check or maintenance task of the power supply of the Ñawi device since it will run out over time. In fact, the 9 v battery installed in the device must be replaced every 6 – 8 h to ensure its proper operation. This time fluctuates between 6 and 8 h depending on the detection work performed and the number of alerts issued and displayed; in addition, the brand of the battery also influences its duration, since some of them last longer than others. Its duration range has been estimated during testing. Besides, smartphone battery of the assisted person must be charged daily, and an active mobile data plan must be maintained so that both assisted person and caregiver are always connected to the system.

As a system in which the safety of the assisted person and the peace of mind of the caregiver are entrusted, the caregiver must ensure a proper operation when the device is turned on for its use by the visually impaired person. Therefore, checking the accuracy of



(a) Degree of acceptance of the stages.



(b) Stages considered to be unnecessary.

Fig. 14. Acceptance of the stages included in TDDM4IoTS.

the sensors and other components that are essential in the obstacle detection should be done periodically. If any anomaly or malfunction is detected in any device component, this should be replaced by other with the same characteristics but in perfect condition.

As for the sensors and other components deployed in the zip-up hoodie, they can be disassembled to be adapted into other garments that the assisted people carry with them, that is, such components can be placed in those garments and places on them where they are most comfortable for each assisted person, provided they can perfectly perform their function. This task of disassembling the components of the device must be done with care, as well as the assembly in the new location, keeping in mind that all the sensors and components must be connected to the ESP8266 NodeMCU board.

4. Validation of TDDM4IoTS

TDDM4IoTS is intended to help students and novice developers to build IoTSs more easily. TDDM4IoTS covers the steps and activities that must be performed in the development of an IoTS. Being based on TDD (Test-Driven Development), this methodology focuses on building clean code and strictly adhering to system requirements, thus leveraging resources efficiently.

For the validation and assessment of TDDM4IoTS as an IoTS development methodology, students of System Engineering at the State Technical University of Quevedo (Ecuador) have been involved. This degree is designed in 10 academic cycles or levels of study (two per academic year, i.e., one per semester). At the end of the tenth academic cycle, the degree of System Engineer is obtained. As of the 2018–2019 academic year, System Engineering students from sixth to ninth semester of their university studies have been developing IoTS pilot projects. In December 2019, TDDM4IoTS was proposed as the development methodology for these students to develop their IoTSs. When the evaluation survey described in this section was conducted, some of them were already professionals (System Engineers). Some of the IoTSs developed by these students during their training period have been published in different international conferences or journals specialized in topics related to IoT. Examples of these publications are the ones entitled *"IoT-Based System to Help Care for Dependent Elderly"* [9], *"IoT-Based Smart Medicine Dispenser to Control and Supervise Medication Intake"* [49], *"IdeAir: IoT-Based System for Indoor Air Quality Monitoring"* ([50,56], b), *"Internet of Things (IoT)-Based Indoor Plant Care System"* [51], and *"IoT-Based System for Classroom Access Control and Resource Management*" [52]. Members of the teams that developed these and other IoTSs were asked to basically determine the effectiveness of the lifecycle stages considered in the methodology proposed to develop this type of systems. The survey was conducted using the Sphinx software and published on its servers. Fig. 11 shows some data on the surveyed developers, which were 47 in total.

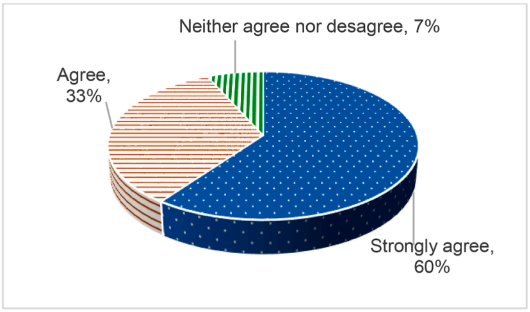
Fig. 11(a) shows the respondents' educational level, while Fig. 11(b) shows the number of years of experience as system developers, and Fig. 11(c) shows the number of IoTS projects they have participated in.

Another aspect included in the survey was the use of any of the existing development methodologies, regardless of the nature of the systems in which these methodologies had been applied. Fig. 12 shows a couple of graphs on the use of methodologies in system development. More specifically, Fig. 12(a) shows the frequency with which developers have rigorously adhered to a methodology, while Fig. 12(b) shows the methodology they have preferentially applied, either totally or partially.

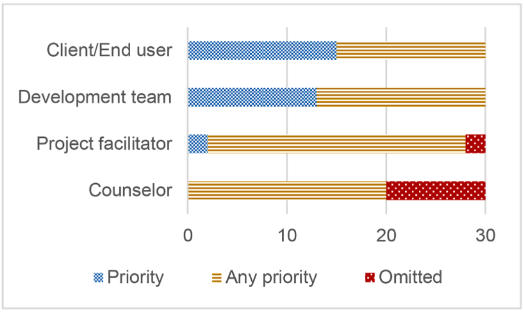
The culture of using a methodology is not the desired one, as shown in Fig. 12(a). Nonetheless, developers who often use a methodology are the predominant ones. Moreover, the traditional Waterfall methodology is chosen in first place to use it preferentially, although the one that has been applied by a greater number of developers is the Scrum methodology, as shown in Fig. 12(b). Those who prefer to use the waterfall methodology have experience as software developers of at least 2 years, representing 46.81% of the total number of respondents.

4.1. Traditional methodologies applied in IoTS development

Regarding the methodologies for the development of traditional software systems, there are respondents who consider that they can be applied to IoTS development. However, most of them are of the opinion that changes should be made in them to be applied to the development of systems in this new paradigm. The graph in Fig. 13(a) shows the results on answers obtained to the question about the possibility of using a traditional system development methodology in IoTS development. Of those who are inclined to the direct



(a) Degree of agreement with assigned roles.



(b) Priority of assigned roles.

Fig. 15. Considerations of respondents about the roles played by the project team members in TDDM4IoTS.

application of a traditional software system development methodology without any change (corresponding to those responding "Yes, definitely" in Fig. 13(a)), 45% of them bet on Waterfall methodology, and 27% on Scrum, while those who prefer TDD are 18%, and the rest are committed to RAD (Rapid Application Development). IoT system developers rely on some traditional software development methodologies that could be adapted or, in some cases, used directly for IoTS development. Fig. 13(b) shows the methodologies they rely on for IoTS development. The methodologies selected by the developer as second or subsequent have been considered as merely mentioned.

4.2. *TDDM4IoTS stages*

Of the developers surveyed in the TDDM4IoTS assessment study, there were 30 respondents who had used this methodology, while 17 had not used it. The analysis presented hereafter refers to the 30 developers who used TDDM4IoTS. Regarding the development stages proposed in TDDM4IoTS, the opinion of the respondents was diverse (see Fig. 14). As can be seen in the graph shown in Fig. 14 (a), the majority agree or completely agree with the stages included in the methodology, while there are three who are neutrals on this issue, and none in disagreement or total disagreement. In addition, 28.20% of them think that some other phases should be added to the methodology (although they do not mention which ones), while 71.80% say that no phase should be added. In our opinion, these results indicate that TDDM4IoTS should be revised if we want to meet the expectations of all developers regarding an ideal IoTS development methodology.

Additionally, the developers surveyed suggest that there are redundant stages (which are not performed), such as *model refinement*, which is the stage with the highest percentage of responses to this question (73.33%). The stages considered unnecessary by some developers are shown in the graph represented in Fig. 14(b).

To better interpret what is shown in Fig. 14(b), we must consider that only the development of proofs of concept and prototypes was considered for the assessment of our methodology, since the developers were students in their training period. Therefore, the IoTSs obtained have not been put into production, but their development process culminated with the testing and evaluation of the corresponding IoTS by the client/end user. This may be the reason why 10% of respondents state that the *maintenance* stage is unnecessary. Perhaps this is also why the developers did not feel the need to refine the models and formally generate tests to increase the quality of the IoTS finally developed, after meeting the functional requirements specified by the users. These same reasons may be why one respondent considered the *detailed analysis of requirements* and *model generation and adaptation* stages as unnecessary stages.

The test generation stage is considered a loss of time by about one-third of the surveyed developers, as reflected in their responses. Although some researchers [53–55] have documented the advantages and successes that can be achieved by using TDD in building systems, the responses of the respondents may be due to the nature (small projects and of day-to-day domains) of the IoTSs developed. Nonetheless, being aware of the importance of considering testing in the development of an IoTS (and regardless of the opinion of some of the respondents), we have developed the first version of the TDDT4IoTS tool [3], and our idea is to continue improving it, to support developers in obtaining tests automatically, which is intended to improve the appreciation that developers have of this stage.

4.3. *Roles of the project team members*

The roles that TDDM4IoTS specifies should be played by project team members are four, namely: client/end user, development team, counsellor, and project facilitator. The graphs in Fig. 15 show the respondents' views on these roles. Fig. 15(a) shows the respondents' degree of agreement with the roles played by the project team members in which they were involved. In addition, considering that they were asked about the order of importance of each of the roles to be played by each project team member, Fig. 15 (b) shows the priority given by the surveyed developers to the roles of the project team members. In it, Priority represents the number of respondents who consider the corresponding role to be the most important (i.e., they selected it in the first place), while *Any priority* denotes the number of respondents who consider a given role to be important but not the most relevant one. And *Omitted* is used for those roles that were not selected by some respondents, which would mean that they considered them as roles that could be dispensed with.

It should be emphasized that the priority was considered by the order of marking. Therefore, only one role is highlighted with the *Priority* label in the answer provided by each respondent, which corresponds to the one marked or selected in first place. As can be seen in Fig. 15(b), the most important roles that cannot be omitted according to the opinion of 100% of the respondents are those corresponding to *client/end user* and *development team*. In addition, the *counsellor* role is a priority for none of respondents, while this role can be omitted for 33% of them. This opinion in favour of the omission of this role may be because, in the projects that the respondents developed prior to this evaluation, all the members of the development teams had the same level of knowledge (being students of the same academic cycle within the same degree) about what they needed to know to implement the corresponding IoTS. Likewise, 7% of those surveyed think the *project facilitator* role can be omitted. As this role was played by the professor of the subject, the respondents could have confused said role in the project with the one played by the teacher in the subject.

When respondents were asked about their degree of agreement regarding whether they consider TDDM4IoTS an appropriate development methodology for IoTS and whether they would use it in future developments, 63.33% were strongly agreed, and 33.33% agreed. There was only one disagreed opinion on the future use of this methodology.

5. Conclusions and future work

In this paper, an IoTS specially designed for visually impaired people, called Ñawi, has been presented as a case study to validate both TDDM4IoTS, which is a development methodology specifically intended to build IoTSs, and TDDT4IoTS, which is its support tool, intended to better integrate hardware and software development in IoTSs. Ñawi consists of a device (in which a series of sensors and other hardware components have been integrated to detect and identify obstacles that assisted people can encounter in their path) and a mobile application for the visually impaired people to be assisted, while two (web and mobile) applications are available for the caregivers who are in change of these people. The device together with the mobile application can help POVP to move in different places, making them more independent and confident regarding their safety. The system also aims to provide greater peace of mind to caregivers, since it manages to prevent accidents caused by the presence of obstacles in the way of people with visual impairments. These must pay attention to the messages that the system sends them (through voice and sounds) so that they can make the right decisions at the right time, and thus be able to avoid or overcome the obstacles that are in their way.

The evaluation of the TDDM4IoTS methodology has been carried out by conducting a survey among developers of at least one IoTS, who had to fill in a questionnaire. There was a total of 47 respondents, of which 30 had used TDDM4IoTS. According to the answers of these 30 respondents, it can be concluded that the development life cycle proposed in TDDM4IoTS is accepted by 100% of them if we include in this group those who are indifferent to it (neither agree nor disagree, which corresponds to 10% of these respondents). If the same inclusion criterion is applied, we can conclude that 100% of them also accepted both the stages and roles proposed in TDDM4IoTS. Moreover, we think that this methodology should be reviewed to check if any unnecessary stage (group of activities) has really been included or if it is necessary to add any stage (or activity to any of the proposed stages), since 20% of the respondents suggest that TDDM4IoTS should include some other stage, while 10% of them consider unnecessary or redundant some stages of its life cycle (Model refinement: 2, and Maintenance: 1). Nonetheless, the context in which the surveys were conducted coincides with what was expressed by the authors of TDDM4IoTS, stating that not all the stages of their methodology must be necessarily executed, and that the intensity of their execution will depend on the type of project and the knowledge of the system domain by the members of the project team.

As future work, we will look to improve the Ñawi system by incorporating ultrasonic sensors into a soft cane to detect obstacles at floor level, such as kerb, stair treads or other objects. This will allow the device to detect obstacles that are not within the field of view of the camera. Additionally, we will work on providing the camera with a mechanism that allows it to tilt in order to recognize these obstacles. To eliminate the risks of not hearing the (voice and sound) alerts that the system plays when the visually impaired person is in noisy places, we will also consider implementing notifications through vibrations. Another area of improvement is to study the possibility of deploying sensors at the level of the lower extremities of the assisted people so that they can move around without needing to carry any other assistive device, such as a cane. Regarding the development methodology, and after analysing the feedback provided by the developers during the validation process presented in this work, we will propose a new version of TDDM4IoTS that incorporates improvements to cover large and small projects. These improvements in the methodology will also entail making the corresponding updates in the TDDT4IoTS tool that supports it.

Declaration of Competing Interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Carlos Rodríguez-Domínguez reports article publishing charges was provided by University of Granada. Carlos Rodríguez-Domínguez reports financial support was provided by Junta de Andalucía Consejería de Educación. Carlos Rodriguez-Domínguez reports financial support was provided by Ministerio de Ciencia, Innovación y Universidades.

Acknowledgements

This research was funded by MCIN/AEI/10.13039/501100011033, i.e., the Spanish Ministry of Science and Innovation (State Research Agency), grant number PID2019-109644RB-I00, and by the Andalusian Regional Government (Junta de Andalucía) and by the European Regional Development Fund (ERDF), a way of making Europe, grant number B-TIC-320-UGR20. Funding for open access charge: Universidad de Granada / CBUA.

References

[1] G Guerrero-Ulloa, C Rodríguez-Domínguez, MJ Hornos, Agile methodologies applied to the development of Internet of Things (IoT)-based systems: a review, Sensors 23 (2) (2023) 790–824, https://doi.org/10.3390/S23020790.

[2] G Guerrero-Ulloa, MJ Hornos, C Rodríguez-Domínguez, et al., TDDM4IoTS: a test-driven development methodology for Internet of Things (IoT)-based systems, in: M Botto-Tobar, M Zambrano Vizuete, P Torres-Carrión, et al. (Eds.), Applied Technologies. ICAT 2019. Communications in Computer and Information Science, eds, Springer, Cham, Switzerland, 2020, pp. 41–55, https://doi.org/10.1007/978-3-030-42517-3_4.

[3] Guerrero-Ulloa G, Carvajal-Suarez D, Pachay-Espinoza A, Brito-Casanova G (2021) TDDT4IoTS: test-driven development tool for IoT-based system. https://bioforest.uteq.edu.ec/tddt4iots/. Accessed 26 May 2022.

[4] WHO: World Health Organization (2022) Blindness and vision impairment. https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment. Accessed 7 Feb 2023.

[5] Henry SL, Thorp J (2008) Web content accessibility and mobile web: making a website accessible both for people with disabilities and for mobile devices. https://www.w3.org/WAI/standards-guidelines/wcag-mobile-overlap/. Accessed 26 Dec 2022.

[6] J Abma, S Abou-Zahra, C Adams, et al., Web Content Accessibility Guidelines (WCAG) 2.2, W3C Accessibility Guidelines, 2021.

[7] Escobar JJM, Matamoros OM, Padilla RT, et al (2020) Smart guide system for blind people by means of stereoscopic vision. In: Arai, K, Kapoor, S, Bhatia, R (eds) Intelligent Systems and Applications. IntelliSys 2020. Advances in Intelligent Systems and Computing, vol 1252. Springer, Cham, pp 527–544. https://doi.org/10.1007/978-3-030-55190-2_39.

[8] M Martinez Gutierrez, JR Rojano Caceres, Evaluating the effectiveness of accessible web sites for deaf users, in: Proceedings - 2019 International Conference on Inclusive Technologies and Education, 2019, https://doi.org/10.1109/contie49246.2019.00032. CONTIE 2019 129–134.
[9] G Guerrero-Ulloa, C Rodríguez-Domínguez, MJ Hornos, IoT-based system to help care for dependent elderly, Commun. Comput. Inf. Sci. 895 (2019) 41–55, https://doi.org/10.1007/978-3-030-05532-5_4.
[10] N Gulati, PD Kaur, FriendCare-AAL: a robust social IoT based alert generation system for ambient assisted living, J. Ambient Intell. Humaniz. Comput. 13 (4) (2022) 1735–1762, https://doi.org/10.1007/s12652-021-03236-3.
[11] N Almurisi, S Tadisetty, Cloud-based virtualization environment for IoT-based WSN: solutions, approaches and challenges, J. Ambient Intell. Humaniz. Comput. 13 (10) (2022) 4681–4703, https://doi.org/10.1007/s12652-021-03515-z.
[12] A Ayimdji Tekemetieu, H Pigot, C Bottari, S Giroux, From speech acts to assistance acts for cognitive assistance in ambient assisted living: how to nudge cognitively impaired people to act independently, J. Ambient Intell. Humaniz. Comput. (2022) 1–27, https://doi.org/10.1007/s12652-022-03735-x.
[13] G Marques, Bhoi AKumar, VHC de Albuquerque, K.S. Hareesha, IoT in Healthcare and Ambient Assisted Living, Springer, 2021, pp. 83–84, https://doi.org/10.1007/978-981-15-9897-5.
[14] G Vallathan, A John, C Thirumalai, et al., Suspicious activity detection using deep learning in secure assisted living IoT environments, J. Supercomput. 77 (2021) 3242–3260, https://doi.org/10.1007/s11227-020-03387-8.
[15] T Linn, A Jwaid, S Clark, Smart glove for visually impaired, in: Computing Conference 2017, London, UK, IEEE, 2018, pp. 1323–1329, https://doi.org/10.1109/sai.2017.8252262.
[16] NS Mala, SS Thushara, S Subbiah, Navigation gadget for visually impaired based on IoT, in: Proceedings of the 2017 2nd International Conference on Computing and Communications Technologies, ICCCT 2017, Institute of Electrical and Electronics Engineers Inc., 2017, pp. 334–338, https://doi.org/10.1109/iccct2.2017.7972298.
[17] Z Saquib, V Murari, SN Bhargav, BlinDar: An invisible eye for the blind people making life easy for the blind with Internet of Things (IoT), in: RTEICT 2017 - 2nd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, Proceedings. Institute of Electrical and Electronics Engineers Inc., 2017, pp. 71–75, https://doi.org/10.1109/rteict.2017.8256560.
[18] D Romo, Manufacture of an electronic device prototype for invident people, Investig. Tecnol. ISTCT 2 (2020) 121–129. http://investigacionistct.ec/ojs/index.php/investigacion_tecnologica/article/view/62.
[19] E Iadanza, G Benincasa, I Ventisette, M Gherardelli, Automatic classification of hospital settings through artificial intelligence, Electronics 11 (11) (2022) 1697–1739, https://doi.org/10.3390/electronics11111697.
[20] D Karagiannis, K Mitsis, KS Nikita, Development of a low-power IoMT portable pillbox for medication adherence improvement and remote treatment adjustment, Sensors 22 (15) (2022) 5818–5832, https://doi.org/10.3390/s22155818.
[21] Berger A, Vokalova A, Maly F, Poulova P (2017) Google glass used as assistive technology its utilization for blind and visually impaired people. In: Younas M, Awan I, Holubova I (eds) Mobile Web and Intelligent Information Systems, MobiWIS 2017. Lecture Notes in Computer Science. Springer, Cham, Switzerland, vol 10486, pp 70–82. https://doi.org/10.1007/978-3-319-65515-4_6.
[22] Akhil R, Gokul MS, Sanal S, et al (2018) Enhanced navigation cane for visually impaired. In: Perez, G, Tiwari, S, Trivedi, M, Mishra, K (eds) Ambient Communications and Computer Systems. Advances in Intelligent Systems and Computing, vol 696. Springer, Singapore, pp 103–115. https://doi.org/10.1007/978-981-10-7386-1_9.
[23] PS Rajendran, P Krishnan, DJ Aravindhar, Design and implementation of voice assisted smart glasses for visually impaired people using Google vision API, in: 4th International Conference on Electronics, Communication and Aerospace Technology, Coimbatore, India, ICECA 2020. IEEE, 2020, pp. 1221–1224, https://doi.org/10.1109/iceca49313.2020.9297553.
[24] J Goncalves, S Paiva, Inclusive mobility solution for visually impaired people using Google cloud vision, in: IEEE International Smart Cities Conference, ISC2 2021, Manchester, United Kingdom, IEEE, 2021, pp. 1–7. https://10.1109/isc253183.2021.9562892.
[25] A H Ali, SU Rao, S Ranganath, et al., A Google glass based real-time scene analysis for the visually impaired, IEEE Access 9 (2021) 166351–166369, https://doi.org/10.1109/access.2021.3135024.
[26] N Harum, NA Zakaria, Z Ayop, et al., Smart Book reader for visual impairment person using IoT device, Int. J. Adv. Comput. Sci. Appl. 10 (2) (2019) 251–255, https://doi.org/10.14569/ijacsa.2019.0100233.
[27] S Tayyaba, MW Ashraf, T Alquthami, et al., Fuzzy-based approach using IoT devices for smart home to assist blind people for navigation, Sensors 20 (13) (2020) 3695–3706, https://doi.org/10.3390/s20133674.
[28] K Vasanth, M Macharla, R Varatharajan, A self assistive device for deaf & blind people using IOT, J. Med. Syst. 43 (4) (2019) 88–95, https://doi.org/10.1007/s10916-019-1201-0.
[29] I Vineeth, YHVS Sharan, Y Karthik, BK Priya, Smart cane for visually impaired person, in: 2021 International Conference on Intelligent Technologies, CONIT, 2021, pp. 1–6, https://doi.org/10.1109/conit51480.2021.9498563, 2021.
[30] IEEE Standards Association, Systems and Software Engineering - Content of Life-Cycle Information Items (documentation), 15289, ISO/IEC/IEEE, 2019, pp. 1–94, https://doi.org/10.1109/ieeestd.2019.8767110, 2019(E) 2019.
[31] K Beck, M Beedle, A van Bennekum, et al., Manifesto for Agile Software Development, Manifesto for Agile Software Development, 2001. https://agilemanifesto.org/. Accessed 24 Nov 2021.
[32] R Rajavel, SK Ravichandran, K Harimoorthy, et al., IoT-based smart healthcare video surveillance system using edge computing, J. Ambient Intell. Humaniz. Comput 13 (6) (2022) 3195–3207, 10.1007/s12652-021-03157-1.
[33] Robocraze (2022a) All about ESP32 Camera Module. https://robocraze.com/blogs/post/all-about-esp32-camera-module. Accessed 9 Jan 2023.
[34] K Gillespie, Ultrasonic Sensors: Advantages and Limitations, MaxBotix, 2019. https://www.maxbotix.com/articles/advantages-limitations-ultrasonic-sensors.htm/. Accessed 9 Jan 2023.
[35] Robocraze (2022b) Arduino VS NodeMCU. https://robocraze.com/blogs/post/arduino-vs-nodemcu. Accessed 9 Jan 2023.
[36] Liocrebif (2022) Gy-521 Mpu6050 6dof module 3-axis acceleration gyroscope. https://www.liocrebif.com/showroom/gy-521-mpu6050-6dof-module-3-axis-acceleration-gyroscope.html. Accessed 9 Jan 2023.
[37] S Veeraraghavan, 11 Best Programming Languages to Learn in 2021 | Simplilearn, Simplilearn - Online Certification Training Course Provider, 2021. https://www.simplilearn.com/best-programming-languages-start-learning-today-article. Accessed 20 Nov 2021.
[38] Mend.io (2020) What are the most secure programming languages? – WhiteSource report. https://www.mend.io/most-secure-programming-languages/. Accessed 20 Nov 2021.
[39] Batschinski G (2022) Firebase alternatives – top 10 competitors. https://blog.back4app.com/firebase-alternatives/. Accessed 5 Jan 2023.
[40] Ankush (2022) Top 13 open source database software for your next project. https://geekflare.com/open-source-database/. Accessed 5 May 2023.
[41] Google Trends, NetBeans, Eclipse, IntelliJ IDEA, 2023. https://trends.google.es/trends/explore?cat=5&geo=EC&q=%2Fm%2F01fchg,%2Fm%2F01fs1d,%2Fm%2F03v0mn. . Accessed 9 Jan 2023.
[42] Medewar S (2022) The 7 best IDEs for mobile application development (in Spanish). https://geekflare.com/es/best-ide-for-mobile-app-development/. Accessed 9 Jan 2023.
[43] D Carter, GIMP Offshoot Aims to Fix its Name Problem | Creative Bloq, Creative Bloq: Art and Design Inspiration, 2019. https://www.creativebloq.com/news/gimp-glimpse. Accessed 20 Nov 2021.
[44] CM Simpao, 10 Best Python IDE & Code Editors [Updated Guide], Hackr.io, 2022. https://hackr.io/blog/best-python-ide. Accessed 19 Apr 2023.
[45] Bonteanu A-M, Tudose C (2023) Multi-platform performance analysis for CRUD operations in relational databases from Java programs using hibernate. In: Hsu CH, Xu M, Cao H, et al. (eds) Big Data Intelligence and Computing. DataCom 2022. Lecture Notes in Computer Science. Springer, Singapore, Singapore, vol 13864, pp 275–288. https://doi.org/10.1007/978-981-99-2233-8_20.
[46] S Almog, Know your debugger. Practical Debugging at Scale, Apress, Berkely, CA, 2023, pp. 3–27, https://doi.org/10.1007/978-1-4842-9042-2_1.

[47] YR Kirschner, M Walter, F Bossert, et al., Automatic derivation of vulnerability models for software architectures, in: IEEE 20th International Conference on Software Architecture Companion, ICSA-C 2023, IEEE, L'Aquila, Italy, 2023, pp. 276–283, https://doi.org/10.1109/icsa-c57050.2023.00065.
[48] S Barroso, P Bustos, P Núñez, Towards a cyber-physical system for sustainable and smart building: a use case for optimising water consumption on a smartcampus, J. Ambient Intell. Humaniz. Comput. (2022), https://doi.org/10.1007/s12652-021-03656-1.
[49] Guerrero-Ulloa G, Hornos MJ, Rodríguez-Domínguez C, Fernández-Coello MaM (2020b) IoT-based smart medicine dispenser to control and supervise medication intake. In: Iglesias CA, J, Moreno Novella I, Ricci A, et al. (eds) Intelligent Environments 2020. Ambient Intelligence and Smart Environments, vol 28. IOS Press, pp 39–48. https://doi.org/10.3233/aise200021.
[50] G Guerrero-Ulloa, A Andrango-Catota, M Abad-Alay, et al., IdeAir: IoT-based system for indoor air quality control, in: V Julián, J Carneiro, RS Alonso, et al. (Eds.), 13th International Symposium on Ambient Intelligence - ISAmI 2022, eds, Springer, Cham, 2023, pp. 197–206, https://doi.org/10.1007/978-3-031-22356-3_19.
[51] G Guerrero-Ulloa, A Méndez-García, V Torres-Lindao, et al., Internet of Things (IoT)-based indoor plant care system, J. Ambient Intell. Smart Environ. 15 (1) (2023) 47–62, https://doi.org/10.3233/ais-220483.
[52] G Guerrero-Ulloa, J Villafuerte-Solorzano, M Yánez, et al., Internet of Things (IoT)-based system for classroom access control and resource management, in: J Bravo, S Ochoa, J Favela (Eds.), Proceedings of the International Conference on Ubiquitous Computing & Ambient Intelligence (UCAmI 2022). UCAmI 2022. Lecture Notes in Networks and Systems., Cham, eds, Springer, 2023, pp. 604–615, https://doi.org/10.1007/978-3-031-21333-5_61.
[53] Lautenschläger E (2022) The perception of test driven development in computer science – outline for a structured literature review. In: Abramowicz, W, Auer, S, Stróżyna, M (eds) Business Information Systems Workshops, BIS 2021. Lecture Notes in Business Information Processing, vol 444. Springer, Cham, pp 121–126. https://doi.org/10.1007/978-3-031-04216-4_13.
[54] S Parsa, Testability driven development (TsDD). Software Testing Automation, Springer, Cham, Cham, 2023, pp. 159–189, https://doi.org/10.1007/978-3-031-22057-9_4.
[55] W Sheikh, Automated unit testing and test-driven development approach to teaching C++, in: W Sheikh (Ed.), 2022 Intermountain Engineering, Technology and Computing, IETC 2022. Institute of Electrical and Electronics Engineers (IEEE), ed, Technology and Computing, IETC 2022. Institute of Electrical and Electronics Engineers (IEEE), Orem, UT, USA, 2022, pp. 1–6, https://doi.org/10.1109/ietc54973.2022.9796750.
[56] Guerrero-Ulloa G, Andrango-Catota A, Abad-Alay M, et al (2023) Development and Assessment of an Indoor Air Quality Control IoT-Based System. Electronics 12(3):608–628. https://doi.org/10.3390/electronics12030608.