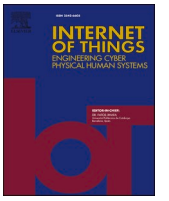
Internet of Things 24 (2023) 100929

Contents lists available at ScienceDirect

Internet of Things

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



An activity-based approach for the early identification and resolution of problems in the development of IoT systems in academic projects

Adriana Collaguazo [a,*], Mónica Villavicencio [a], Alain Abran [b]

[a] ESPOL Polytechnic University, Escuela Superior Politécnica del Litoral, ESPOL, Facultad de Ingeniería en Electricidad y Computación, Campus Gustavo Galindo Km. 30.5 Vía Perimetral, 09015863, Guayaquil, Ecuador
[b] École de Technologie Supérieure (ETS), Department of Software and IT Engineering, Montreal, Canada

ARTICLE INFO

Keywords:
Internet of Things
IoT problems
Computer Science Education
Engineering Education
IoT Taxonomy

ABSTRACT

The development of Internet of Things (IoT) systems in university courses encourages students to use multiple skills. Hence, the importance of applying teaching methods like Project Based Learning (PBL) in the development of these kinds of systems for integrating hardware and software components while facing numerous real-life problems. This study presents an activity-based Approach for the Early Identification and Resolution of problems, in a checklist format, to be used by students for preventing them from wasting time in IoT academic projects. The checklist is based on an adapted taxonomy of problems in IoT systems. This study analyzed problems identified in 48 projects carried out by 183 engineering students registered in two courses in 2020 and 2021. For designing the structured checklist, we categorized 14 types of IoT problems, analyzed root causes and solutions, and created and evaluated a taxonomy of problems.

1. Introduction

The demand for specialized professionals in the Internet of Things (IoT) has increased with the development of new IoT devices and technologies, and the price of these devices has decreased, making them affordable and facilitating access to IoT networks.

While education on IoT has an impact on the skills acquired by engineering students to interconnect hardware and software over the Internet through communication protocols [1], education on IoT itself is an emerging area of research.

Educational institutions and universities have already incorporated IoT-based courses into the curricula of engineering and computer science degrees [2] through Science, Technology, Engineering, and Mathematics (STEM) programs, where students learn by performing lab activities. The content of these IoT courses introduces the fundamentals of IoT, design, development, and deployment of IoT-based systems [3,4]. For the development of lab projects in these courses, Arduino (i.e., an open-source hardware development board programmed by sending a set of instructions to the microcontroller) is commonly used because of its low cost, wide range of sensors, ease of use, open source, and large community of developers [5,6].

Although STEM students have knowledge of hardware or software, they still encounter a number of problems in the development of IoT-based systems, such as problems in the physical connection of IoT devices, compatibility between devices or protocols, communication with IoT hardware, the use of cloud services, and software development [7]. The motivation for this study was to prevent

* Corresponding author.
E-mail address: acollag@espol.edu.ec (A. Collaguazo).

https://doi.org/10.1016/j.iot.2023.100929

Available online 7 September 2023

students from making mistakes and wasting time on IoT system development. The specific objective of the research reported here was the development of an activity-based approach for the early identification and resolution of problems in a checklist format to be used by students involved in IoT academic projects. To achieve this objective, it was necessary to enhance the taxonomy of problems in IoT systems development proposed by Makhshari [7] and the general architecture of IoT systems for academic projects proposed in [8]. Therefore, in this research work we have contributed with:

- A new version of the taxonomy of problems in IoT systems originally proposed by Makhshari [7].
- An updated version of a general architecture of IoT systems used in academic projects proposed in [8]; and
- A Problem-solving checklist for helping students to prevent and identify errors in the various phases of the development of IoT systems.

This study was based on data collected from 48 academic projects in two courses, including the development of IoT systems. From the analysis of students' problems that occurred in their IoT academic projects and their related root causes, we categorized them using an IoT-related taxonomy adapted from [7]. Next, we designed a problem-solving checklist as a solution for students enrolled in the IoT academic projects. It is worth mentioning that this study is the continuation of our main research project called CEMA, which is a Cloud-based Education Model published in 2020 [8]. The model was created to facilitate the teaching process for developing advanced mobile applications, which combine the use of IoT hardware and cloud computing services, as well as to provide access to freely available open educational resources.

The remainder of this paper is organized as follows. Section 2 describes related work. Section 3 presents the IoT-based courses and the layers of the architecture of the IoT systems used in this study. Section 4 describes our research methodology, including the proposed taxonomy of problems in systems, the student survey, and the checklist for the development of academic projects of IoT systems. Section 5 presents an analysis of the data collected on problem types, root causes, and proposed solutions. Section 6 presents the key findings and suggestions for future work.

2. Related works

This section presents related works on the development of IoT systems in the educational field, grouped by the following technologies for sending data to the IoT network or the Internet: the Arduino platform and the Sigfox technology or ESP modules. It also includes IoT system development methodologies, which help to structure an IoT project considering the multidisciplinarity of required knowledge needed for this type of system. In addition, in this section we present some IoT architectures that have been proposed for the development of IoT systems; and finally, we include taxonomies of IoT systems that have been proposed by other authors.

2.1. Academic projects with Arduino platform

The Arduino development platform emerged in 2005 to ease students and people without high technical training to develop electronics projects using kits containing all the parts necessary to carry them out [9]. The use of Arduino is focused on rapid programming and low-cost prototyping [10]. Likewise, publications highlight the use of the Arduino board in university teaching robotics, electronics, circuit design, and control of electrical or mechanical equipment. A study of 422 courses from 45 universities that used hardware development platforms reported that Arduino and Raspberry Pi were the most used in 65% and 29% of the analyzed courses, respectively.

According to the authors, these figures are strong indicators that Arduino is one of the most widely used technologies for teaching and learning low-cost prototyping in universities. In [11], Krelja et al. described the effects on learning for first-year computer science undergraduate students at Rijeka Polytechnic through the implementation of medium-complexity projects using an Arduino kit. The study concludes that this board is suitable for learning because of its ease of use and low cost. In addition, this type of project promotes problem-solving skills among students. Hertzog and Swart described how design-based courses are taught at the Central University of Technology in South Africa and how undergraduate engineering students interact in projects with Arduino as the design platform [12], allowing students to gain experience in programming software that integrates directly with hardware devices.

2.2. Academic projects using Sigfox technology or ESP modules for developing IoT system

The study in [2] presented a framework for teaching IoT system development in engineering and master's programs at the University of Zilina, Slovakia. This study presents three examples of projects developed by undergraduate students using Arduino, Sigfox (for connectivity to the IoT network), and various sensors (gas detectors, smart containers for municipal waste, and contactless deduction of electricity consumption by households' electric meters). This study mentions that carrying out these projects helps students improve their skills in the development of simple hardware and software solutions; within a short time, the students were able to put into practice their theoretical knowledge for the implementation of the projects. The study in [13] presented the application of an IoT-based project learning (IoTbPL) approach to the course of 35 Biomedical Engineering students at the School of Electrical and Computer Engineering of the Bandung Institute of Technology (ITB). The project theme was a smart baby incubator, and the students could select hardware tools to carry out the project. Some students chose the ESP8266 module to connect to the Internet and send data to the mobile app, whereas others chose the Arduino platform.

2.3. Proposals for developing IoT systems

Both in industry and in academia, alternatives have been proposed to develop IoT systems. On the one hand, the industry has sought to disseminate good practices in the development of this type of systems. And on the other hand, the research works of the universities have oriented their proposals to facilitate the teaching and assessment of learning in introductory IoT courses.

For example, there is a study about the Ignite Methodology proposed by Slama et al. [14]; this methodology originates in the industry in 2015 and had the collaboration of experts from different disciplines in order to record their best practices. This open-source methodology supports the design, configuration, and management of IoT projects through templates, checklists, and architecture models. However, the article does not include technical details regarding programming and testing the software of the IoT solution. Ignite is divided into two phases: *(1) IoT Strategy* defines the IoT strategy and prepares the organization for IoT adoption, *(2) IoT Solution Delivery* focuses on planning, building, and running IoT solutions.

Another example is the framework created by Maenpaa et al [15] with the objective of assessing IOT projects in university education. The framework for problem-based learning was created and applied with 47 students in a course taught at the university of Helsinki from 2014 to 2016. This framework contains criteria to evaluate the outcomes of the projects divided into seven themes (i.e. hardware implementation, software architecture, testing methods, human computer interaction, physical design, group work and product concept) and evaluated with a scale from 1 to 3. Prior to the evaluation, the researchers used a collaborative teaching method, using educational resources created by the global Maker community, which encourages personalized learning tailored to the skill levels of each student.

In 2019, Khanafer et al. presented guidelines for teaching an introductory elective course of IoT. This course is taught in the Computer Engineering program at the American University of Kuwait [16]. The guidelines outline the prerequisites and skills needed to be enrolled in this course, propose assessment and grading schemes such as assignments, quizzes, project, and describe the topics to be covered in the course. In the article, the authors emphasize the importance of practical experience; therefore, the course grade scheme assigns a high percentage to the project.

In 2020, our research team proposed a Cloud-based Education Model (CEMA). It was created to develop -in an academic environment- IoT systems managed by a mobile application [8]. This model consists of three phases: *(1) Identifying industry partners* It begins by contacting industry clients to define the scope of the projects to be developed by students, *(2) Creating or updating educational resources* consists of the creation of open educational resources, such as project guide, teacher slides, micro-training sessions, and laboratory practices for students, *(3) Developing projects in a cloud environment*, phase in which the students received accompaniment from the professor and two academic assistants in the development of their projects.

In 2022, Ferreira et al. [17] proposed the Three-Phase Methodology which originated in academia- for IoT project development (TpM-Pro). The phases are: *(1) Considering the business* which starts with understanding the business well, in this way the solution will satisfy the needs of the client, *(2) Gathering the requirements* where functional and non-functional requirements are defined, and *(3) Implementation* which includes the selection and evaluation of the technologies that will satisfy the previously defined requirements. The TpM-Pro define agents and roles such as Client, Project manager, Development Manager, and Multidisciplinary Development Team for organizing the development process.

Finally, Ferreira et al. investigated six development methodologies of IoT systems focused on the identification of challenges that arise during the implementation of this kind of systems [17]. Of these six, we only analyzed those that focus on IoT projects which are Ignite and TpM-Pro.

2.4. Architectures for the Internet of Things

The IoT references architectures serve as a general guideline to facilitate interoperability, simplify development, and ease implementation.

In [18] the authors present a comparison of the Internet of Things Architecture (IoT-A) and Industrial Internet Reference Architecture (IIRA) in terms of their capabilities and layers. Of the compared aspects, we focus on analyzing two of them, since they are similar to those, we propose in our research with student projects. The first is Internet orientation, which encompasses networking, transport, and data links. Both architectures deal with all these aspects but refer with more emphasis to M2M communication to address the lower layers of the Open Systems Interconnection (OSI) model. The second is things orientation, which focuses on sensors, actuators, and tags. Both IoT-A and IIRA take into account management and security mechanisms through all their layers.

A general architecture of IoT systems that facilitates the development of academic projects was proposed by us in the CEMA model, which was briefly explained in the previous subsection [8]. This architecture contains the IoT Devices layer where the sensors transmit data to the Sigfox Cloud layer. In the Sigfox Cloud layer, a callback is programmed, which allows sending this data to the Cloud Services Providers layer that stores the data. The data can be viewed from an application in the Mobile Application layer. This architecture has been improved and used for the research work reported in this article (more information can be found in Section 3.2).

2.5. IoT system taxonomies

There is a study that reports the creation of a taxonomy of IoT systems describing characteristics related to security and privacy, and focusing on the reduction of vulnerabilities in the development process of these systems [19]. This taxonomy focuses exclusively on the classification of IoT security attacks based on application scenarios such as industry, urban infrastructure, smart environment, and healthcare.

The study reported in [7] presents a bug taxonomy for IoT systems based on data mining. Specifically, software bugs in IoT systems were pulled from GitHub repositories and categorized by IoT topic. In addition, tags from semi-structured interviews and surveys directed at IoT developers were collected to define root causes for each bug category. This bug taxonomy is the basis on which we proposed a new version of the taxonomy explained in Section 4.2 of this article.

In summary, through the review of related works, we identified a gap in the absence of a problem-solving checklist for helping students to prevent and identify errors in the various phases of the development of IoT systems. The works published by university researchers mostly suggest alternatives to teach and assess learning in IoT courses; hence, we developed a problem-solving checklist that verifies activities from planning, connectivity, cloud services, hardware and software integration to project closing. These problem-solving checklist activities are founded on a cloud-based education model, an architecture of four layers for the interconnection of hardware and software components, and a taxonomy of problems applicable to projects developed in an academic environment.

3. IoT project-based courses investigated in this study

This section presents the courses used for our research and the general architecture of IoT systems used to conduct academic projects, which were developed using our Cloud-based Education Model (CEMA) [8].

3.1. Courses with IoT academic projects

The two courses investigated in this study are:

- Mobile Applications and Telematic Services (AMST) is an elective course offered to students majoring in Science, Technology, Engineering, and Mathematics (STEM).
- Programming of Telematic Systems (PST) is a compulsory course for students of the Mechatronics Engineering career.

These two courses were taught in a virtual mode from May 2020 to September 2021 for both theoretical class sessions and laboratory practices. Due to the pandemic, these courses were not held in person for the first time. The development of the students' projects was carried out in an unsupervised environment: the students did not have access to the Telematics Systems Laboratory, whereas prior to the pandemic, they used computers with pre-installed software to develop projects and test their prototypes. During the indicated period, support for students in the development of IoT system projects was strengthened with hours of mentoring by professors and teaching assistants (hardware and software assistants); therefore, all the educational resources of CEMA were applied as is –with no changes.

Both AMST and PST courses are taught for 14 weeks, the duration of an academic period at ESPOL. The academic projects in both courses were conducted in groups of three to four students.

The data for this research come from 48 students projects during three academic periods as depicted in Table 1. In each project, students used Arduino connected to the Internet with Low-power wide-area network (LPWAN) technology [20], such as Sigfox [21,22] or with ESP8266/ESP32 WiFi modules [23], to send the data from the IoT device. In addition, students used platforms for data storage in the cloud and developed web or mobile applications as required in these courses. The scope of all the projects was to develop a prototype of the IoT system managed through a web or mobile application. Appendices [24] list the topics of the 48 projects carried out in the AMST and PST courses, taken as a basis for the identification of students' problems. For example: In the PR1 Backpack tracking v2.0 project, students developed a prototype of the system adapted to a backpack, with which they monitor its location in real-time through a mobile application. The information on this project is available at [25].

3.2. General architecture of IoT systems applied in the course projects

Fig. 1 shows the general architecture of IoT systems used in the AMST and PST course projects, which is based on our previous work reported in [8]. This architecture includes four layers for the interconnection of hardware and software components.

Table 1
Project numbers per course.

Course	Semester	Student numbers	Project numbers
AMST	1S-2020	15	5
	2S-2020	17	6
	1S-2021	20	5
PST	1S-2020	42	10
	2S-2020	39	9
	1S-2021	50	13
Total		183	48

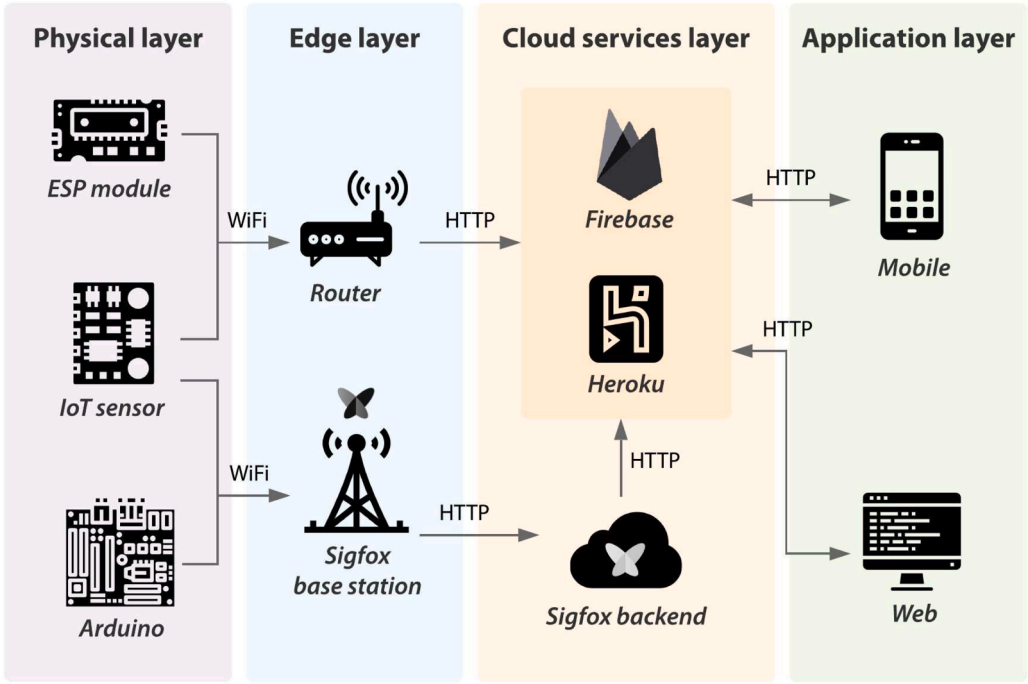


Fig. 1. General architecture of IoT systems used in academic projects [8].

1. *Physical layer:* This layer corresponds to things that become intelligent by being programmed [26], integrating hardware development boards, sensors, and actuators [27,28] to interact with the real world. Arduino and ThinkXtra Sigfox [29], iButton devices with GPS sensors [30], and Suntech with GPS sensors [31] were used in the AMST projects, whereas ESP8266 and ESP32 WiFi modules were used in the PST projects. Weight sensors (HX711 sensor), gas (MQ-2 sensor), sound (KY-038 sensor), air quality (MQ-135 sensor), and temperature and humidity (DHT11 sensor) were used in both courses.
2. *Edge layer:* This layer allows routing of the sensed data originating from the IoT devices from the physical layer to the cloud services layer without manipulating them. This layer is based on communication protocols such as HTTP, which allows data to be sent through wired or wireless technologies such as 0G [32], 3 G, 5 G, LTE, and IEEE 802.11 [33].

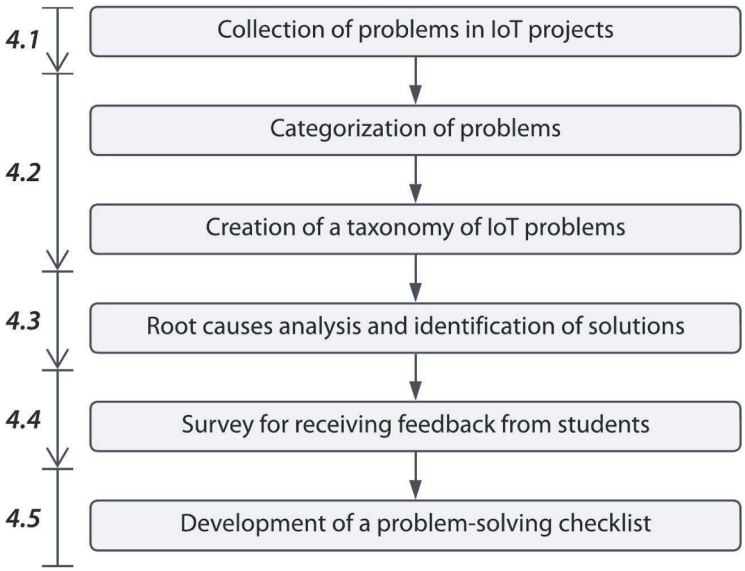


Fig. 2. Phases of the study methodology.

3. *Cloud services layer:* This layer works bi-directionally with the application layer and is responsible for the storage and processing of data required by software applications [34]. Google Cloud services such as Firebase [35] were used in some of the projects, and Heroku [36], among others.
4. *Application layer:* This layer allows the monitoring of information by end users through combined and analyzed data, which can contribute to decision-making [37]. For AMST course projects, it is necessary to develop a mobile application, whereas, for PST, these applications can be web or mobile depending on the type of project. The tools used for the development of mobile applications were the Android Studio integrated development environment (IDE) [38], and for web applications, the execution environment for JavaScript Node.js [39], and the React library [40].

4. Study methodology

The methodology used for this study contains 5 phases as depicted in Fig. 2, which are aimed at identifying problems in IoT systems developed in an academic environment, as well as the creation of a control mechanism (checklist) to prevent such errors. All the IoT systems analyzed for this study were developed using the CEMA model.

4.1. Collection of problems in IoT projects

We collected the problems reported by 183 students while developing 48 IoT projects in AMST and PST courses over three academic semesters from May 2020 to September 2021. Problems were collected from the following media:

1. Aula Virtual [41], the Learning Management System (LMS) used at ESPOL University.
2. Microsoft Teams, where hardware and software teaching assistants conducted consultation sessions with student groups.
3. Piazza [42] is a free online platform that allows students to post questions and teachers to answer them.
4. Project rubric file, which allows monitoring and evaluating the project.
5. Commit comments made by students in the GitHub repositories of their projects.

4.2. Categorization of problems and creation of a taxonomy of IoT problems

For problem categorization, we selected the IoT bug classification proposed by Makhshari and Mesbah [7] and tailored the categories and subcategories into the taxonomy of IoT problems, considering what happened in the AMST and PST courses (Fig. 3).

The IoT bug taxonomy in [7] was based on bug reports in IoT GitHub repositories. Since our needs are focused on identifying IoT problems that occur during the development of students' projects with mobile applications, we adapted the taxonomy using the types of problems collected in Phase A as shown in Table 2.

- **IoT device:** This category covers problems related to the hardware and firmware components of IoT devices.
  *Hardware:*This subcategory includes physical problems with IoT devices, such as wiring, pin state, hardware development board, sensors, and actuator problems. It also includes problems related to the limitations of IoT devices, such as limited computational power, batteries, memory, and processing capacity. Additionally, the enclosure problem is included in this subcategory, which refers to the inadequate design to protect enveloped electronics from adverse effects of the surrounding environment that can cause failures in IoT devices [43]. For example, an enclosure attached to the Arduino board: when the electronic components such as the

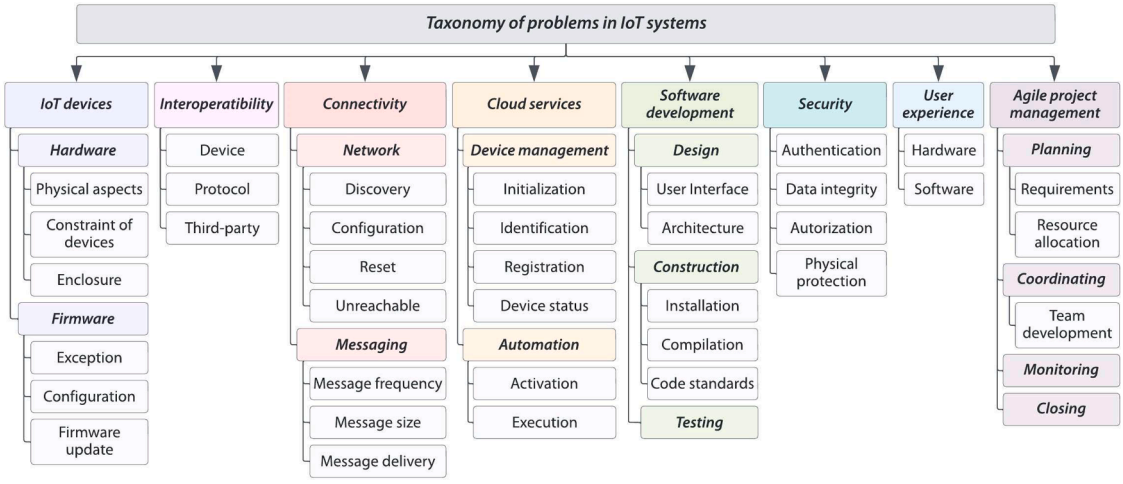


Fig. 3. Taxonomy of problems in IoT systems development.

Table 2
Initial taxonomy from [7] and related tailoring to problems observed in students' projects.

Initial taxonomy	Tailoring and improvements
IoT device	IoT device: In this category we added the enclosure problem subcategory that deals with the inadequate design of the electronic enclosure.
Compatibility	We changed it for an Interoperability category as it relates to hardware and software problems caused by the variety of heterogeneous devices.
Communication with IoT Devices	We changed it to the Connectivity category with 2 subcategories: 1) Network in which the problems related to the Internet connection are described; and 2) Messaging which deals with the problems of sending of data in IoT systems through the cloud.
Cloud/Edge Services	Cloud services: This category deals with functionality issues in IoT devices when connecting to cloud services.
General Development	We added Software development, Security, User experience, and Agile project management categories.

HX711 weight sensor and the MFRC-522 RFID reader module are not well connected, they may be damaged, de-soldered, or disconnected, as in PR4 and PR16 projects.
Firmware: This subcategory includes unexpected device exceptions, device configuration exceptions, and firmware update exceptions. For example, not uploading the instruction list (sketch) to the Arduino board or the ESP8266 module causes firmware update problems.

- **Interoperability:** Interoperability problems when using a variety of devices, communication protocols, or third-party components in the IoT system lead to massive heterogeneity [44] and interoperability problems. For example, the excess buffer size of the serial port that communicates with the Arduino Uno board with the Bluetooth module in PR1 leads to the need for a greater number of serial ports. Another example is the SIM900 GSM/GPRS shield module mounted on top of an Arduino board, which may have errors in the serial communication protocol that prevents sending or receiving SMS, as happened in the PR8 project.
- **Connectivity:** This category covers 2 subcategories:
Network: These problems are related to the establishment of an Internet connection through the IoT network or gateway router, including network discovery, network configuration, reset network (see PR3), and unreachable networks.

- A network discovery problem occurs when an IoT device cannot recognize available wireless networks through the local access point (AP) or Sigfox IoT network.
- A network configuration problem occurs when the Arduino board does not have a properly configured Service Set Identifier (SSID), username, or valid password to access the Internet.
- A reset network occurs when a student moves from one room to another in his house or other places, losing the Internet connection.
- An unreachable network problem occurs when the IoT device does not have access to the Internet because of the gateway router is down or when the device passes through a non-covered area of the Sigfox IoT network. These problems arose in the PR3, PR4, PR9, and PR13 projects.
Messaging: These problems correspond to sending data in IoT systems through the cloud. The most common problems are message frequency, size, and delivery.

- Message frequency problems occur when Sigfox devices exceed 140 regulated messages per day.
- The message-size problem occurs when Sigfox messages exceed 12 bytes. For example, in the PR14 project, the message size was exceeded when sending multiple real-time sensed data such as temperature and humidity (DHT11 sensor), longitude and latitude (GY-NEO6MV2 GPS module), concentration of harmful gases (MQ-2 sensor), detection of open flames (KY-026 flame sensor), and percentage of external battery consumption.
- The message delivery problem occurs when the callback does not transmit data to a third-party server or an IoT platform. This problem can occur in the Arduino by improperly configuring the payload with Sigfox functions. This problem can also occur in the Sigfox backend in the configuration of the custom payload, which allows the user to decode the payload using distinct, simple variables. Sigfox has only a few available variable types and cannot distinguish between different frame format types [45]. For example, in project PR15, the data obtained through the KY-038 sound sensor were not sent when it detected an obstruction in the drain because of an incorrect configuration of the callback in the backend of Sigfox.

- **Cloud services:** These are problems related to the services provided by servers in the cloud or the IoT platform.
Device Management: Includes functionality problems of initialization, identification, registration, device connection, and device properties in each IoT device that connects to cloud services. An initialization problem occurs in an IoT device when it is not properly initialized to use the cloud services. An identification problem occurs when an Arduino with the ESP8266 module does not have the hostname or connection credentials to the cloud-hosted database (e.g., Firebase). A registration problem occurs when the device token is invalid because the token expiration date is before the current date. Device status problems are related to checking whether the status of the IoT device is online.

Automation: This category of problem is related to the activation and execution of cloud computing services. The activation problem occurs when there is a mistake in the setup of a Firebase Cloud Messaging client application on Android to send and receive messages and notifications. This may be because the registration tokens were not updated in the mobile device application. The execution problem occurs when the mobile device does not have permission configured to receive background messages.

- **Software development:** This category includes the Software Development Life Cycle (SDLC) [46] and the software engineering model for IoT [47], considering requirements elicitation, design, construction, and testing.
Design: Problems related to high-level design and detailed design of IoT systems. A high-level design generates a set of documents and diagrams that define the architecture of the system, whereas the detailed design is related to the design of the user interfaces of the applications [48].
Construction: This refers to the programming of applications and IoT devices [46]. Some of these problems relate to the installation of dependencies, compilation, and coding standards.
Testing: These problems arise in the verification of the behavior of both software and IoT devices [47]. The components of IoT systems, such as sensors, applications, networks, and data storage, must be considered in the tests. However, it is very common for not all functional and non-functional test cases to be covered because of the complexity of the integration between hardware and software compared to classic test models.

- **Security:** This category includes security problems IoT systems face such as authentication, data integrity, authorization, and physical protection.

  - Authentication problems arise in users, objects, and devices that need to authenticate each other and cannot do so reliably.
  - The data integrity issue is caused by attacks that result in the loss or manipulation of data handled in the IoT system.
  - The authorization problem (between devices and/or users) corresponds to the access permissions to system resources.
  - A physical protection problem refers to the vulnerability of devices to adverse environmental conditions, such as heavy rain or high temperatures [49], or physical damage due to falls or theft. For example, in Firebase data storage, when there are no security rules that restrict read or write permissions, a data integrity problem occurs.

- **User Experience:** Refers to user experience problems that are evidenced by not making the usability of hardware or software easy and natural [47]. For example, a poorly designed IoT device (hardware) can cause a user to not use the IoT device intuitively. This can also cause the user to manipulate electronic components that should not be exposed to the user [49]. The software user experience problem occurs when a mobile or web application does not mimic the operation of IoT devices with visual elements. This can cause discontinuities in the user experience. For example, a smart plug is controlled by a mobile application that does not have two widgets: one widget to indicate the plug state (on or off), and another widget to change the plug state (turn on or turn off) [50].
- **Agile Project Management:** This category includes problems associated with project planning, coordination, monitoring, and closing.

  - The planning problem involves the elicitation of requirements, negotiation, review, and resource allocation [51]. The latter deals with sufficient people, facilities, infrastructure, and support.
  - The coordinating problem encompasses team development when there is ineffective communication between the team members affecting the delivery of the project, personnel dependency problems, which refer to the absence of an individual, causing the team to be multifunctional [52], and the unequal division of tasks among the team members, considering the responsibilities and deadlines for their tasks [53].
  - The monitoring problem is related to the lack of monitoring of project progress.
  - The closing problem is related to the incomplete completion of deliverables (products, services, documentation, or any expected result).

Our taxonomy allowed us to categorize and subcategorize the 14 problem types identified in the AMST and PST courses.

4.3. Root cause analysis and identification of solutions

Professors and teaching assistants analyzed each problem type by reviewing comments and images reported by students and proposed recommended solutions as shown in Table 3.

4.4. Survey for receiving feedback from students

A list of problems, causes, and solutions (Table 3) was proposed to students who were asked to provide feedback in relation to their own 12 projects. For this phase, a questionnaire using Google Forms [54] was developed to collect data on:

- Students' demographic information, such as gender, age, course, major, how far students were from finishing their degree, and if they had work experience.

Table 3
Initial version of the checklist.

Category	Subcategory	Id	Problem	Cause	Solution
Cloud services	Device management	PB1	The IoT device cannot transmit data to cloud services.	The IoT device token is expired.	Register the device with active tokens.
	Device management			The device passes through an area not covered by Internet or IoT network.	Switch to an area covered by Internet or IoT network.
Connectivity	Network	PB2	Geolocation data acquisition error.	Lack of coverage or interference in the satellite signal.	Incorporate a higher-gain antenna or test in a supervised/controlled environment (open environment).
	Messaging	PB3	The message delivered cannot be interpreted by the callback.	The data encapsulation format is incorrect between IoT device and backend.	Encapsulate the sent data using the valid format.
Interoperability	Protocol	PB4	Communication failure between sensor and development board.	Communication protocol.	Use another shield, development board, or electronic components.
Agile project management	Coordinating	PB6	Lack of coordination between team members of different modules of the system.	It takes time to coordinate and adapt the code segments for the correct functioning of the system.	Document the code of the development board and application using agile methodology.
	Planning	PB9	Lack of hardware materials for replacing damaged parts.	Low availability of materials in the city.	Have enough materials in stock before beginning system development.
Software development	Construction	PB5	Incorrect method call.	Discrepancy in the number or types of method parameters method.	Review the documentation of the method.
	Construction	PB7	PC does not detect the IoT device.	Drivers not installed or installed.	Install the necessary drivers according to the operating system.
	Construction	PB8	Code compilation errors.	Bad programming or incorrect selection of the open hardware development board.	Select the open hardware development board based on the minimum technical requirements.
	Construction	PB10	Sensors did not record data correctly.	The microcontroller has incorrectly programmed the sensor variables.	Use the official code given by the manufacturer.
IoT device	Hardware			The sensor is incorrectly connected to the development board.	Review datasheet from a development board and sensor.
	Hardware	PB11	Some component freezes.	Improperly connected cable short-circuit causing the development board to constantly restart.	Review physical connections or change the development board for more capacity.
	Hardware			The development board is exposed to adverse external environments.	Reset or turn off the device.
	Hardware	PB12	The development board or sensor was misconfigured.	Electrical power failure causes development board and sensor shutdown or serial cable disconnection.	Gradual programming of functions and constant recording (calibration).
	Hardware	PB13	Development board external battery level discharges too quickly.	Excess hardware components connected to the development board.	Use external battery backup or power bank (e.g. Li-Po, 1.5V alkaline rechargeable batteries).
	Hardware			Prolonged unnecessary use of the sensors in the development board.	Use power saving mode (sleep method) that completely shuts down sensors.
	Hardware	PB14	Serial connection error to the development board.	Polarity of the Tx and Rx pins inverted or on other pins in the development board.	Change Tx and Rx pins using the manufacturer's data sheet.
	Firmware			Baud rate incompatibility.	Correctly configure the serial port in baud rate in both devices to communicate.

- Frequency of problems in 12 academic projects, the level of complexity (when relevant), and their identified causes.
- Professionals and professors defined the severity levels of each problem using the following problem resolution time-based ranges: low (1–30 min), medium (> 30 min–1 h), and high (> 1 h).

The questionnaire was distributed to students registered in the AMST and PST courses during the second semester of 2021 (2S-2021 from October 2021 to February 2022). A total of 44 students from 12 academic projects participated in the online survey.

4.5. Development of a problem-solving checklist

To help students in subsequent courses adequately address and even avoid the problems identified in this study, we created in this last phase a checklist of recommended activities for the early identification and resolution of problems that may occur when developing IoT systems. The activities in this checklist were grouped using the taxonomy of problems adapted to IoT system development (Fig. 1). Each activity was elaborated based on the analysis of the causes presented in Table 3. Furthermore, additional causes identified by the students through an online survey were included.

The activities proposed in this checklist are based on the methodology described in [55] and project-based learning (PBL). For the academic projects described in this study, we defined seven phases with 30 suggested activities to recognize and prevent problems in the development of IoT systems as depicted in Table 4:

Table 4
Problem-solving checklist for the development of IoT systems in students' projects.

No.	Activity	Complies (✓) Yes	Complies (✓) No
	Planning		
1	Keep a constant flow of communication among team development (e.g. using ASANA for organizing tasks).		
2	Define activities and control your project, be agile.		
	Hardware development		
3	Check the availability of materials (e.g. sensors).		
4	Check any problem of hardware malfunction (e.g. usb data cable to connect from the board to PC is in good condition).		
5	Connect cables, jumpers, or other means of connection between components (e.g. board, sensors).		
6	Verify pins of the board are connected correctly in the datasheet (e.g. Tx and Rx pins).		
7	Check the electrical continuity and correct polarity between terminals (e.g. using multimeter).		
8	Check that the baud rate of the serial port is equal in both components (e.g. PC and board or board and serial modules).		
9	Check extern power source has charge enough (e.g. using multimeter).		
10	Check power source is disconnected when a change in the electrical circuit is required.		
11	Check board and sensors is not exposed to extreme environmental conditions.		
12	Check there is no short-circuit on board and components.		
13	Verify hardware properly function.		
14	Disconnect the board and its power source when is not in use (e.g. USB cable or battery).		
	Connectivity		
15	Check communication protocol used is compatible with backend or network services (same protocol).		
16	Verify quality of the signal obtained (e.g. Sigfox backend or WiFi signal strength).		
	Cloud services control and monitoring		
17	Verify ESP module is sending properly to the cloud (e.g. using Arduino IDE).		
18	Verify token is active in the Sigfox Backend.		
19	Check the custom payload sent to the Sigfox backend have less than 12 bytes in the board programming.		
20	Verify data types of the variable(s) match those defined in the callback of the Sigfox backend (only Sigfox).		
21	Verify correct sensed data is being received from devices (e.g. sending messages through commands or check messages in Sigfox backend).		
	Software development		
22	Check required libraries have been installed to establish connection with cloud services.		
23	Check documentation/programming standard of the methods to be used when coding (e.g. Android development, official documents).		
24	Check application permissions necessary in mobile application (e.g. GPS, camera, WiFi sensors embedded).		
25	Running the source code of the mobile application (e.g. Android Studio).		
26	Solving bugs in the code (e.g. review documentation of similar models specific board).		
	Integration		
27	Test the system and check that all the test cases defined in the user stories are successful.		
28	Verify that the IoT system is working properly.		
	Closing		
29	Verify that the documentation of the IoT system is complete in the GitHub repository (i.e. architecture diagrams, source code).		
30	Make a live presentation showing the operation of the IoT system.		

1. *Planning-monitoring and control:* In this phase, the students are in charge of planning and distributing tasks among the members of the development team.
2. *Hardware development:* This phase includes the delivery and/or acquisition of physical equipment. Activities 3 to 6 refer to a review of the status of the hardware components and their connection to the development board. Activities 7 to 14 help verify the correct operation of the hardware, as well as the necessary care to prevent equipment from suffering breakdowns during development.
3. *Connectivity:* In this phase, the students are in charge of creating the necessary configurations to establish device communication. Activities 15 and 16 allow checking the use of an adequate communication protocol and that the quality of the signal obtained is acceptable.
4. *Cloud services control and monitoring:* In this phase, students connect and configure cloud services to store information obtained from hardware devices. Activity 17 applies to students whose projects use the ESP module [56] to send data to the internet. However, activities 18 to 20 apply to students who use Sigfox technology, which allows the data to be sent to the Sigfox backend to be properly structured. Activity 21 allows checking the reception of the data using either the ESP or the Sigfox module.
5. *Software development:* This phase applies to the development of the source code in the Arduino IDE, as well as the code of the web or mobile application. Activities 22 through 24 help students make sure they have the necessary dependencies installed and access permissions to run their applications, as well as to ensure the use of coding standards when programming. Activities 25 and 26 refer to the execution of code to check for possible errors.
6. *Integration:* The integration phase of the project is considered one of the most important aspects of this checklist. At this stage, the operating part of the hardware must be connected to the web or mobile application; therefore, it is necessary to test the system as a whole to check that it works as requested. Activity 27 indicates that students must ensure that they meet specified functional and non-functional requirements. Activity 28 verifies the correct operation of the complete IoT system, tests the flow of data obtained from the hardware devices and the correct storage in the cloud platforms and obtains the proper information to be displayed to the final user.
7. *Closing:* In this phase, the students complete their project development activities. Activities 29 and 30 help verify that the documentation of the project is complete and stored in the repository and that a final live presentation of the operation of the IoT system has been made.

5. Analysis of the data collected on problem types and root causes

This section presents an analysis of the data collected in this study, including the frequencies of the problems in developing IoT systems and their corresponding root causes and solutions using the proposed taxonomy. It also reports the evaluation of the taxonomy of problems based on the students surveyed.

5.1. Problems, causes and solutions identified by professors and teaching assistants

The frequency of the problems identified by professors and teaching assistants in the students' projects is summarized in Fig. 4, using the 14 types of problems from the proposed taxonomy. The most frequent types of problems are (Fig. 4):

- PB6 - Lack of coordination between team members,
- PB8 - Code compilation errors, and
- PB10 - sensors did not record data correctly.

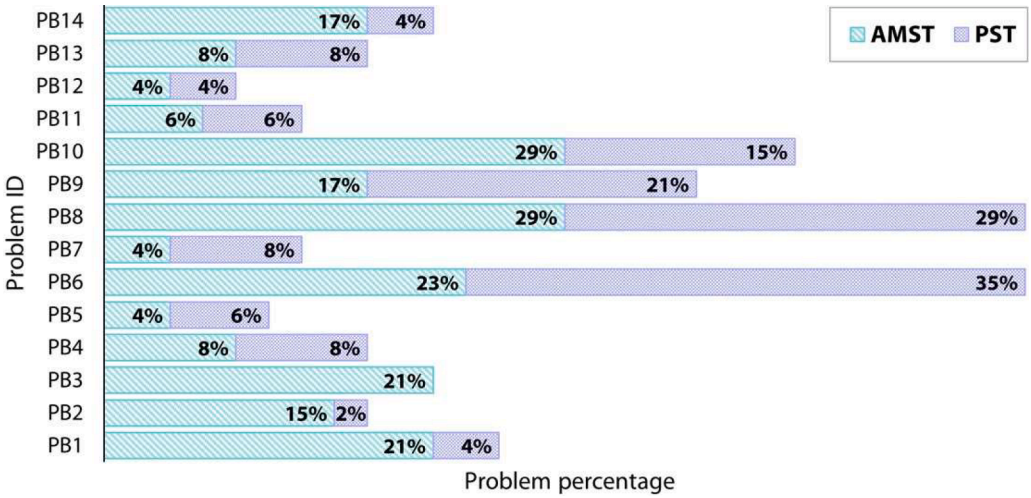


Fig. 4. Frequencies of problems collected from IoT academic projects.

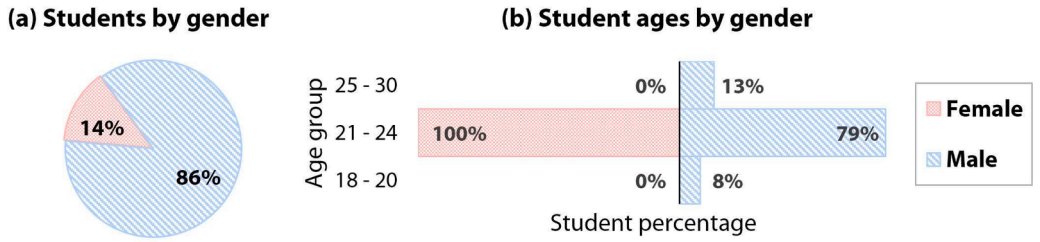


Fig. 5. Age versus gender of students.

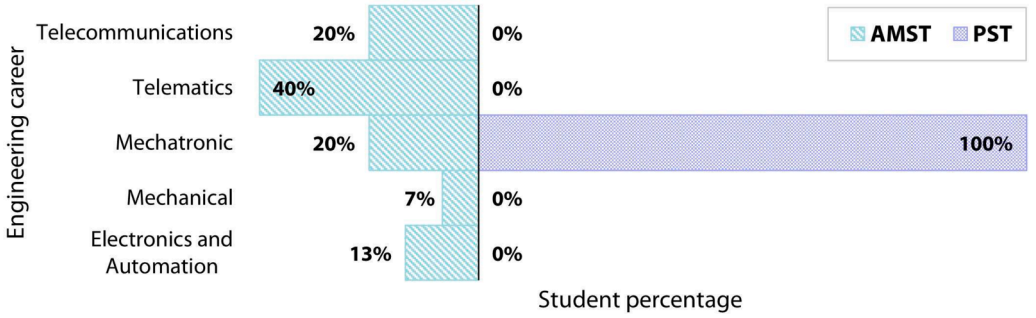


Fig. 6. Percentage of students by career.

These problems mainly arose because the class sessions were taught virtually in 2020 and 2021, so the students had to build each part of their IoT system individually and then integrate them with their work team; however, it was often not possible to get together to do so. In addition, communication between team members was not always adequate and only through digital media.

5.2. *Students survey - demographics*

The students surveyed were mostly male (86%) and mainly in the age group between 21 and 24 years (Fig. 5). The careers to which the surveyed students belonged were Mechatronics, Mechanical, Telematics, Telecommunications, and Electronics and Automation engineering. In the AMST course, most students are registered in the Telematics Engineering major, while in PST, all students are in the Mechatronics Engineering program (Fig. 6). This is because the PST course is required in Mechatronics, while AMST is an elective course for various engineering programs.

Regarding the level of progress in which the students are when they take the courses, regardless of their career, the students prefer to take the courses after the third year of study. In the case of mechatronics, there are students in their second year of study (200-II), which is due to changes made in the curriculum, as shown in Fig. 7.

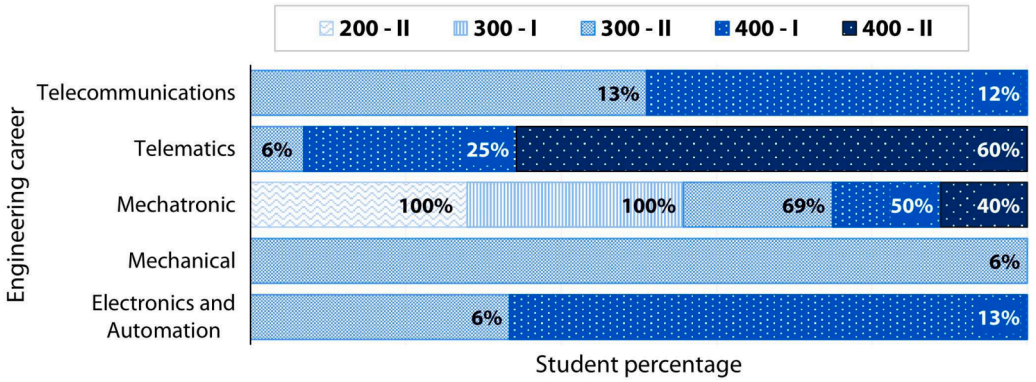


Fig. 7. Level of progress of students in their careers.

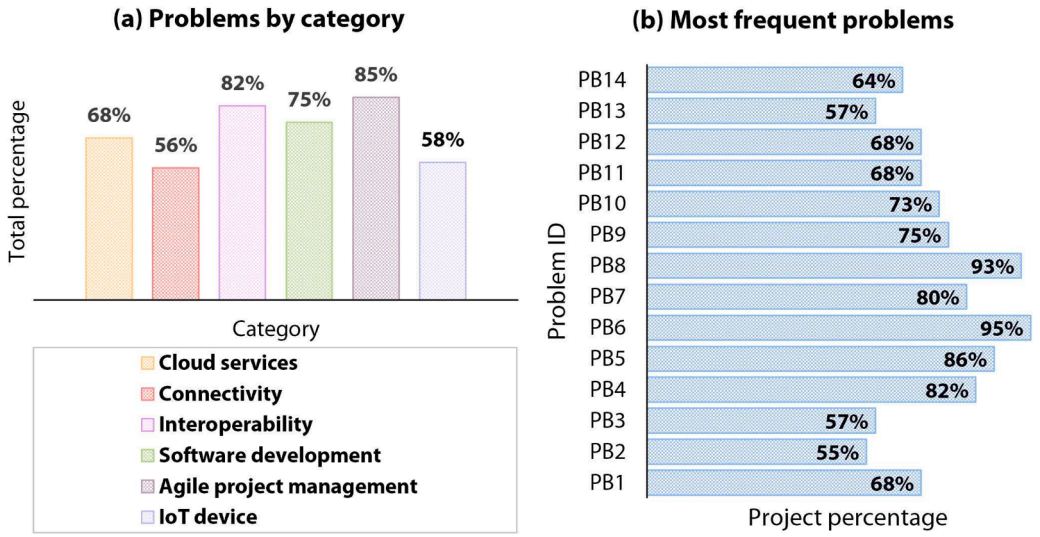


Fig. 8. Occurrence of problems according to the perception of the students.

5.3. Students survey – problems categorization by students

According to the surveyed students, the category of problems that occur most frequently is Agile project management, followed by Interoperability and Software development (Fig. 8.(a)). The most frequently reported problems were PB6 (95%), PB8 (93%), and PB5 (86%) (Fig. 8.(b)). This is similar to that obtained from the analysis of professors and teaching assistants, since problems PB6 and PB8 frequently appear in student projects.

Regarding the level of severity, 14 types of problems were evaluated by students using a low, medium, and high scale (Fig. 9). Problems PB1 (the IoT device cannot transmit data to cloud services), PB3 (the message delivered cannot be interpreted by the callback), and PB9 (lack of hardware materials for replacing damaged parts) were the ones they reported with the highest level of severity. We consider that problem PB9 has a high severity level because students need to purchase IoT devices to carry out the projects, which sometimes break and need spare parts. Students stated that it was difficult to get around because of COVID-19 to obtain replacement hardware components.

In the survey, students were also asked about the root causes of the 14 problem types. For this question, students could select the causes of the problems and add causes that were not on the list. The additional causes identified by the students were freely expressed by writing the corresponding explanation. After that, the causes were categorized by professors and teaching assistants according to the taxonomy of IoT problems, as shown in Table 5.

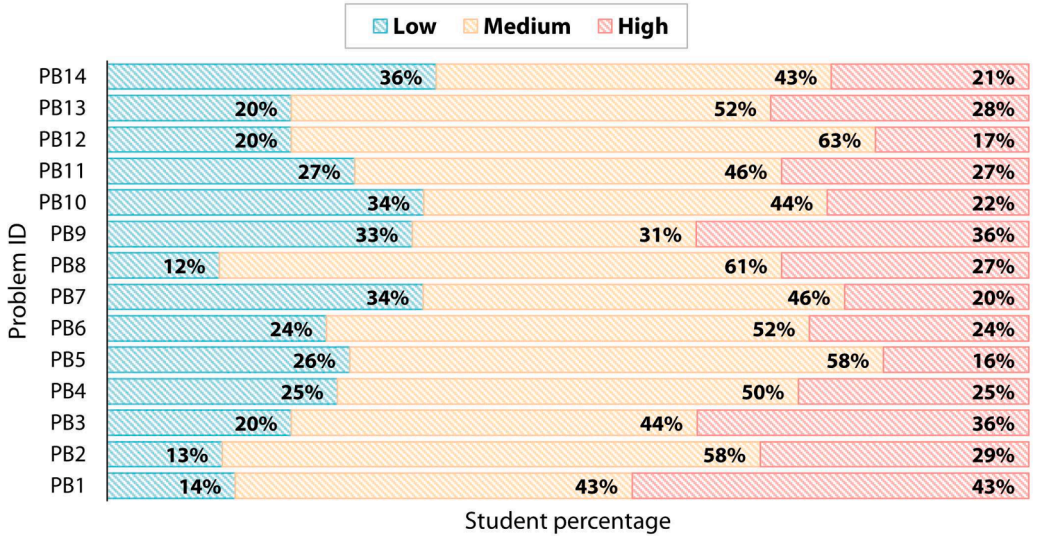


Fig. 9. Severity level of each problem type according to students.

Table 5
Additional causes of problems identified by students in an online survey.

Id	Problem	Additional cause
PB1	The IoT device cannot transmit data to cloud services	Lack of knowledge of how to make the connection with new components. Making mistakes when connecting the ESP8266 module to a database. Difficulty in configuring the connection to Firebase in Arduino.
PB2	Geolocation data acquisition error	Problems with the cell phone's operating system.
PB4	Communication failure between sensor and development board	Poorly calibrated sensor.
PB5	Incorrect method call	New code in Arduino with errors. Confusion in the use of new methods.
PB7	PC does not detect the IoT device	Data cables are in disrepair.
PB10	Sensors did not record data correctly	Limitations of simulators due to lack of libraries when simulating the connection to the database.
PB11	Some component freezes	Some types of variables used in the microcontroller made it reboot.
PB12	The development board or sensor was misconfigured	The first time a microcontroller is used, it needs to be configured and it is necessary to learn how to do that.

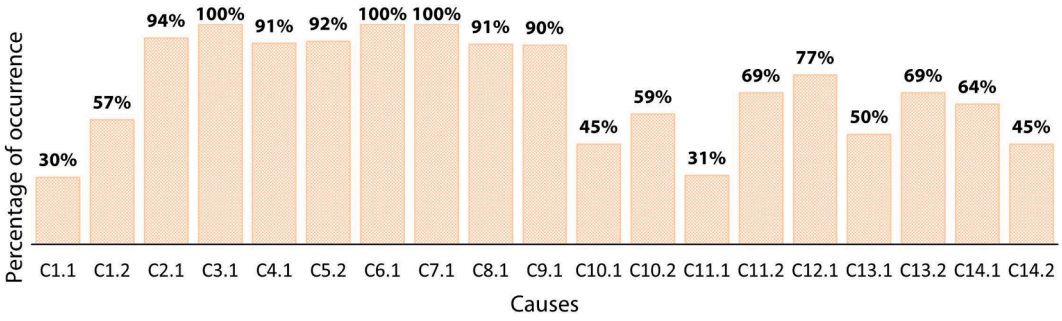


Fig. 10. Percentage of root cause per problem as identified by students.

Fig. 10 shows all causes and their percentage. Each cause appears with its indicator, where the first number represents the problem with which the cause is associated; and the second number is the cause number according to the order presented in Table 3. As shown in Fig. 10, 100% of the students agreed with the root causes proposed for problems PB3, PB6, and PB7.

6. Conclusions

IoT systems have become part of our lives, and teaching how to develop such systems has been included in the curricula of university programmes. It is common for students to face several problems during the elaboration of these systems, particularly when they lack knowledge of the tools to be used. This study collected students' problems faced in IoT-based courses and categorized them for the development of a structured problem-solving tool in a checklist format in order to prevent students from making mistakes and wasting time in IoT system development. For developing the checklist, it was necessary to enhance the general architecture of IoT systems proposed in the CEMA model as well as the taxonomy of problems in IoT systems created by Makhshari [7]. From a group of 48 projects developed, which were carried out in three academic terms, 1S-2020, 2S-2020, and 1S-2021, 14 types of problems were identified among the groups of students. To address or avoid such problems, we created a problem-solving checklist that can be used by students for the development of future projects. This checklist can also be used by students from other universities who have developed IoT systems with similar characteristics.

We also include the results of the online survey administered to students in the second academic term, 2021. The most frequent problems in a sample of 12 projects were related to 3 categories: agile project management, software development, and IoT device. This checklist arose because university students do not have enough experience and knowledge to anticipate the problems they may encounter when working with hardware and software elements simultaneously, since many times, the laboratory practices carried out by students in other subjects have been sufficiently tested for the student to work in a controlled environment and without unexpected failures. It is important that software engineering concepts are taught to engineering students who develop IoT systems to avoid problems that arise due to poor coding or resource management of their projects. In our study, we found that the second category, in which more problems arose, was software development. Therefore, it is important to foster student skills in this area.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

I have shared the link to my data (Appendices and GitHub Repositories) at the Attach File step.
GitHub Repositories (Original data) (GitHub)
Appendices (Original data) (Google Drive)

References

[1] R. Sharma, Internet of Things: an approach for advancement in educational institution, India International Conference on Information Processing (IICIP) (2016) 1–4.
[2] D. Jarinova, A. Kanalikova, D. Ticha, Project oriented teaching in Internet of Things education, 18th IEEE International Conference on Emerging eLearning Technologies and Applications (ICETA) (2020) 248–253.
[3] S.J. Lee, A. Jung, M. Yun, Creative Internet of Things (IoT) for undergraduates, 14th International Conference on Computer Science and Education (ICCSE) (2019) 567–572.
[4] M. Al-Emran, S.I. Malik, M.N. Al-Kabi, A survey of Internet of Things (IoT) in education: opportunities and challenges. Studies in Computational Intelligence 846, 2020, pp. 197–209.
[5] R. Grover, S. Krishnan, T. Shoup, M. Khanbaghi, A competition-based approach for undergraduate mechatronics education using the arduino platform, 4th Interdisciplinary Engineering Design Education Conference (IEDEC) (2014) 78–83.
[6] H.K. Kondaveeti, N.K. Kumaravelu, S.D. Vanambathina, S.E. Mathe, S. Vappangi, A systematic literature review on prototyping with Arduino: applications, challenges, advantages, and limitations, Comput. Sci. Rev. 40 (2021) 100364.
[7] A. Makhshari, A. Mesbah, IoT bugs and development challenges, IEEE/ACM 43rd International Conference on Software Engineering (ICSE) (2021) 460–472.
[8] A. Collaguazo, M. Villavicencio, A. Abran, Education model for developing IoT and cloud mobile applications, IEEE World Congress on Services (SERVICES) (2020) 251–258.
[9] W.-L. Hsu, W.-T. Chen, H.-H. Kuo, Y.-C. Shiau, T.-Y. Chern, S.-C. Lai, W.-H. Fan, Establishment of smart living environment control system, Sens. Mater. 32 (2020) 183–195.
[10] A.J. Swart, Analyzing the application of two main microcontrollers in engineering education – a case study of three IEEE conferences focusing on education, Adv. Sci. Technol. Eng. Syst. J. 6 (3) (2021) 339–346.
[11] E. Krelja Kurelovic, J. Tomljanovic, M. Kralj, Students' attitudes about learning on Arduino projects, 14th International Technology, Education and Development Conference 1 (2020) 125–129.
[12] P.E. Hertzog, A.J. Swart, Arduino - enabling engineering students to obtain academic success in a design-based module, IEEE Glob. Eng. Educ. Conference (EDUCON) (2016) 66–73.
[13] A.W. Setiawan, Implementation of internet of things in biomedical measurement and instrumentation course project, IEEE Glob. Eng. Educ. Conference (EDUCON) (2020) 1657–1661.
[14] D. Slama, F. Puhlmann, J. Morrish, R.M. Bhatnagar, Enterprise IoT: Strategies and Best practices for connected products and services, O'Reilly Media, Inc., 2015.
[15] H. Maenpaa, S. Varjonen, A. Hellas, S. Tarkoma, T. Mannisto, Assessing IOT projects in university education - a framework for problem-based learning, in: 2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering Education and Training Track (ICSE-SEET), 2017, pp. 37–46.
[16] M. Khanafer, M. El-Abd, Guidelines for teaching an introductory course on the Internet of Things, in: 2019 IEEE Global Engineering Education Conference (EDUCON), 2019, pp. 1488–1492.
[17] L.C.B.C. Ferreira, P.R. Chaves, R.M. Assumpção, O.C. Branquinho, F. Fruett, P. Cardieri, The three-phase methodology for IoT project development, Internet of Things 20 (2022), 100624.
[18] M. Weyrich, C. Ebert, Reference architectures for the Internet of Things, IEEE Software 33 (1) (2016) 112–116.
[19] K. Chen, S. Zhang, Z. Li, Y. Zhang, Q. Deng, S. Ray, Y. Jin, Internet-of Things security and vulnerabilities: taxonomy, challenges, and practice, J. Hardware and Syst. Security 2 (2) (2018) 97–110.
[20] U. Raza, P. Kulkarni, M. Sooriyabandara, Low power wide area networks: an overview, IEEE Commun. Surveys Tutorials 19 (2) (2017) 855–873.
[21] A. Lavric, A.I. Petrariu, V. Popa, Long range SigFox communication protocol scalability analysis under large-scale, high-density conditions, IEEE Access 7 (2019) 35816–35825.
[22] F. Pitu, N.C. Gaitan, Surveillance of SigFox technology integrated with environmental monitoring, 15th International Conference on Development and Application Systems (DAS) (2020) 69–72.
[23] B. Karaduman, M. Challenger, Model-driven development for ESPbased IoT systems, IEEE/ACM 3rd International Workshop on Software Engineering Research and Practices for the IoT (SERP4IoT) (2021) 9–12.
[24] A. Collaguazo, Appendices (2022). URL https://bit.ly/3OwJWEp.
[25] A. Collaguazo, GitHub repositories belonging to the AMST Course (2020). URL https://github.com/orgs/AMST-FIEC-ESPOL/repositories.
[26] W. Lv, F. Meng, C. Zhang, Y. Lv, N. Cao, J. Jiang, A general architecture of IoT system, IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC) 1 (2017) 659–664.
[27] E. Navarro, N. Costa, A. Pereira, A systematic review of IoT solutions for smart farming, Sensors 20 (15) (2020) 4231.
[28] L. Ambrosio, P.L. Siqueira Paulino, J. Antiquera, G.P. Aquino, E. Cesar Vilas Boas, EcoWaste: a smart waste platform enabling circular economy, IEEE 19th Student Conference on Research and Development (SCOReD) (2021) 411–415.
[29] Thinxtra, Thinxtra Sigfox developer Xkit. URL https://thinxtra.com/iot-connectivity/xkit/.
[30] Sigfox Partner Network, iButton GPS. URL https://partners.sigfox.com/products/ibutton-track.
[31] Suntech International Ltd., ST710 – custom telematics for your business. URL http://suntechint.com/st710/.
[32] B. Alqahtani, B. AlNajrani, A study of internet of things protocols and communication, 2nd international conference on computer and information sciences (ICCIS) (2020) 1–6.
[33] N. Pawar, T. Bourgeau, H. Chaouchi, Study of IoT architecture and application invariant functionalities, IFIP/IEEE International Symposium on Integrated Network Management (IM) (2021) 667–671.
[34] C.M. Sosa Reyna, E. Tello Leal, D. Lara Alabazares, Methodology for the model-driven development of service oriented IoT applications, J. Syst. Archit. 90 (2018) 15–22.
[35] Firebase. URL https://firebase.google.com/.
[36] Heroku, Cloud application platform. URL https://www.heroku.com/home.

[37] A. Alshehri, R. Sandhu, Access control models for cloud-enabled internet of things: a proposed architecture and research Agenda, IEEE 2nd International Conference on Collaboration and Internet Computing (CIC) (2016) 530–538.
[38] Android Developers, Meet android studio. URL https://developer.android.com/studio/intro.
[39] OpenJS Foundation, Node.js. URL https://nodejs.org/en/.
[40] Meta Platforms Inc., React – a JavaScript library for building user interfaces. URL https://reactjs.org/.
[41] Escuela Superior Politécnica del Litoral, Aula virtual (2022). URL https://aulavirtual.espol.edu.ec/.
[42] Piazza, Ask. Answer. Explore. Whenever. URL https://piazza.com/.
[43] F. Süli, An overview of enclosures, housings, and packages, electronic enclosures, housings and packages (2019) 3–16.
[44] A. Čolaković, M. Hadžialić, Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues, Comput. Networks 144 (2018) 17–39.
[45] Sigfox, Callback principles (2022). URL https://support.sigfox.com/docs/custom-callback-creation.
[46] P. Bourque, R.E. Fairley, IEEE Computer Society, Guide to the Software Engineering Body of Knowledge - SWEBOK, 3rd Edition, IEEE Computer Society Press, 2014.
[47] D.J. Mala, Integrating the Internet of Things Into Software Engineering Practices, IGI Global, 2019.
[48] S.A. Ehikioya, E. Guillemot, A critical assessment of the design issues in e-commerce systems development, Eng. Rep. (2020).
[49] A. Oliveira, C. Resende, A. Pereira, P. Madureira, J. Gon¸calves, R. Moutinho, F. Soares, W. Moreira, IoT sensing platform as a driver for digital farming in rural Africa, Sensors (Basel, Switzerland) 20 (12) (2020) 1–25.
[50] C. Rowland, E. Goodman, M. Charlier, A. Light, A. Lui, Designing Connected Products: UX For the Consumer Internet of Things, 1st Edition, O'Reilly Media, Inc., 2015.
[51] T. Raharjo, B. Purwandari, Agile project management challenges and mapping solutions: a systematic literature review, 3rd International Conference on Software Engineering and Information Management (2020).
[52] R. Hoda, L.K. Murugesan, Multi-level agile project management challenges: A self-organizing team perspective, J. Syst. Softw. 117 (2016) 245–257.
[53] K.T. Lyra, M.L. Alves, F.H.C. Silva, K. Souza, S. Isotani, An agile project management experience, XXXII Brazilian Symposium on Software Eng. (2018) 240–249.
[54] Google Forms, Encuesta de evaluación de la taxonomía de problemas y causas en el desarrollo de proyectos de sistemas IoT en los cursos de AMST y PST (FIEC-ESPOL). URL https://forms.gle/FEQNqM3sk8BmFkQDA.
[55] R. Crespo, E. Lopez-Caudana, P. Ponce, An immersive week for undergraduate engineering students for developing IoT competencies a case study at Tecnologico de Monterrey in Mexico, IEEE 11th International Conference on Engineering Education (ICEED) (2019) 90–95.
[56] A. Maier, A. Sharp, Y. Vagapov, Comparative analysis and practical implementation of the ESP32 microcontroller module for the Internet of Things, 7th International Conference Internet Technologies and Applications (ITA) (2017) 143–148.