Informatics in Education, 2020, Vol. 19, No. 1, 129–157
© 2020 *Vilnius University*
DOI: 10.15388/infedu.2020.07

Teaching Ubiquitous Computing Using Simulations Based on Smartphone Sensors

Maria Joelma P. PEIXOTO[2], Paulo A. S. DUARTE[2], Pedro T. ARAÚJO[2], Pedro I. C. PINTO[1], Wellington W. F. SARMENTO[1], Fernando A. M. TRINTA[2], Windson VIANA[1,2]

[1]*Digital Systems and Media, Federal University of Ceará (UFC), Fortaleza, Ceará, Brazil*
[2]*Group of Computer Networks, Software Engineering and Systems (GREat) – Federal University of Ceará (UFC), Fortaleza, Ceará, Brazil*
e-mail: jlmpxt@gmail.com, pauloduarte@great.ufc.br, pedrotaraujo@alu.ufc.br, pedro.italo3693@gmail.com, wwagner@virtual.ufc.br, fernando.trinta@ufc.br, windson@virtual.ufc.br

Received: August 2019

Abstract. Mark Weiser coined the term Ubiquitous Computing (UbiComp) describing a future in which everyday life-objects would have embedded computers providing services anytime and anywhere. This paradigm is theme recurrent in many graduate courses of Computer Science around the world. To better understand the challenge of teaching Ubiquitous Computing (UbiComp), we surveyed 15 professors and 60 graduate and undergraduate students from 16 universities. According to this survey, the two most challenging Ubicomp concepts to explain in a lecture are context-awareness and middleware platforms. Results also showed professors' difficulty in finding tools to assist the practical teaching of UbiComp's concepts. Current UbiComp tools require high programming skills or they are not designed for educational purposes. Therefore, this work presents the design, development, and evaluation of LUCy (Learning Ubiquitous Computing Easily), a Virtual Learning Environment which aids UbiComp practical classes. LUCy has two main elements: a Web tool and an Android mobile app. The former provides UbiComp theory materials, videos, practices, and simulations. The latter uses smartphones features and sensors to run simulations of UbiComp concepts. We evaluated LUCy during Context-Awareness classes in UbiComp courses, at the same university, along with three distinct semesters. In different three sessions, we gathered information about LUCy's pedagogical and usability issues. Then, we performed a quasi-experiment using a pretest and posttest design methodology with twenty-seven students. Results showed LUCy practices significantly improves students reasoning about Context-Aware concepts.

Keywords: ubiquitous computing teaching, sensor-based simulation, context-awareness, learning environment.

1. Introduction

Technological progress has contributed to the cheapening and dissemination of various computing devices (e.g., laptops, smartphones, sensors, actuators) as well as different wireless technologies. This progress brings the human closer to the computer vision advocated by Mark Weiser. In his research, Weiser imagined the future in which technology would be so embedded in people's daily lives that it would be indistinguishable from the environment, mixing it with everyday objects (Weiser, 1991). In "The Computer for the 21st Century", Weiser introduced the term Ubiquitous Computing (UbiComp). In this new paradigm, computational systems would be embedded in elements of daily life, being able to exchange data between themselves without much intervention of users (Weiser, 1991). UbiComp concepts play vital roles in the implementation of many scenarios of the Internet of Things (e.g., Smart Buildings, Smart Shopping).

Due to the technological and research potential of UbiComp, many universities around the world have undergraduate and graduate courses that are teaching several concepts of this domain. They include themes such as the capture of experiences and intentions, adaptability, decentralization, service discovery, heterogeneity, and context-awareness (Lima, 2011; Chalmers, 2015). These concepts are studied in Computer Science courses and related fields (e.g., Computer Engineering, Digital Design). The main challenges for teaching UbiComp[1] include the multidisciplinary nature of these research field and the lack of tools to support classes about its concepts (Cárdenas-Robledo and Peña-Ayala, 2018; Girouard *et al.*, 2018).

In our literature review, we have found studies reporting experiences in teaching UbiComp concepts, such as (Richards *et al.*, 2012), (Silvis-Cividjian, 2015), and (Chalmers, 2015). Professors use artifacts or digital tools that are not specifically designed for UbiComp teaching. Some examples include Node-RED[2], created for the Internet of Things; App Inventor[3], focused on teaching programming logic; and the traditional mobile application programming environments (e.g., Android Studio[4]).

In this context, our research focused on the design, development, and evaluation of a tool to assist professors in Ubiquitous Computing classes. The core novelty of our approach is the usage of students' mobile phones to make the simulation of theoretical concepts easier by using mobile phone sensors' data. Our research followed the User-Centered Design (UCD) methodology. First, we performed a literature review to understand the recurrent difficulties in teaching UbiComp and possible requirements to create this tool. We also applied a survey of UbiComp teaching in 16 universities. We collected information from 15 professors and 60 graduate and undergraduate students. According

[1] In this paper, we adopt the term UbiComp as synonymous of Pervasive Computing. IBM created the term Pervasive Computing in 1998 and claimed a world fulfilled with computing services accessible anywhere and anytime. At its beginning, Pervasive Computing was more related to smart spaces and mobile computing scenarios (Nieuwdorp, 2007). Nowadays, it is more challenging to identify a clear border separating these two concepts.

[2] `https://nodered.org/`

[3] `http://appinventor.mit.edu`

[4] `https://developer.android.com/studio`

to this survey, we have identified that the two most challenging UbiComp concepts to explain are Context-Awareness and Middleware. We also confirmed the challenge for professors in finding tools to assist the practical teaching of UbiComp concepts.

Based on the results of this survey, we designed an environment that uses students' smartphones and their sensors to help UbiComp classes. We choose these artifacts because they are more financially accessible by most students. Our tool is called LUCy – Learning Ubiquitous Computing Easily. LUCy offers a practical approach to simulate and learn concepts taught in the UbiComp classes. LUCy has two modules: a Web tool and an Android mobile application. The former allows the professor or the student to configure the simulation of the studied concepts (e.g., activation of a contextual rule based on the device movement). The latter allows the student to perceive the execution of his configuration with the aid of the device sensors (e.g., the smartphone flashlight turns on after a certain level of shaking the device). The tool was designed using modular and extensible principles so that professors can use it to explain several Ubicomp concepts, such as context-awareness, heterogeneity, adaptability, and also extend it with new ideas.

We evaluated LUCy with the following research questions in mind:

(RQ1): How well accepted is our simulation tool for teaching ubiquitous computing by students and professors?

(RQ2): What is the impact of our simulation tool in students' learning of theoretical concepts of UbiComp?

We evaluated LUCy during three sessions of a Computer Science course, with undergraduate and graduate students, and with six Computer Science professors from five different universities. Our objective was to measure aspects of usability and user-interface quality of the tool, as well as the pedagogical aspects of LUCy. The usability evaluation was measured by the System Usability Scale SUS (Brooke, 2013), where the average score of 79.85 points was obtained, indicating the tool's good usability. We aim in this paper help the structuring of Ubiquitous Computing courses. Also, we seek to give insights concerning the design of learning tools to assist UbiComp teaching.

The remainder of the paper is organized as follows: initially, we present the methodology that we follow in the conception and design of LUCy in Section 2. After this, we describe LUCy in Section 3. In Section 4, we present the evaluations we have performed with professors and students, as well as the results obtained from these experiments. We have analyzed these results in Section 5. In Section 6, we expose the related work to this research. In Section 7, we present our final considerations and the next steps of our research.

2. Methodology

Our research followed the User-Centered Design (UCD) methodology to avoid unnecessary information and useless features on LUCy. Our primary goal was to design LUCy according to the real needs of its potential users (i.e., professors and students). Based on an international standard, the UCD outlines the steps throughout the design and development life-cycle of a product.

In this product life-cycle, the target users participate in all phases. In UCD, the developer does not only define how end-user interaction with the desired system, product, or service will happen, but he also conducts a series of interviews and tests with real users to validate the desired outcome throughout all its building process, from design to the implementation and final evaluation. According to Wood and Romero (2010), UCD is the practice of creating products so that users can use them with the minimum of stress and maximum efficiency. This methodology has proved to be very useful for system design in general (Wood and Romero, 2010).

2.1. *Literature Review and Online Survey*

The first step of our research was to investigate the scenario of UbiComp courses around the world. First, we collected data from the scientific literature. We found a few papers dealing with the Ubiquitous Computing teaching theme. Section 6 presents part of the state-of-art in Ubiquitous Computing teaching. We also carried out an online survey with 60 students and 15 professors from several universities, in which Ubicomp is taught in graduate and undergraduate courses. The objective was to identify the structure of the classes, the main concepts studied, and what are the most significant difficulties faced by professors and students.

First, we sent an e-mail to professors that research on Ubiquitous Computing we known. After, we used Research Gate[5], a popular social network of researchers to find other UbiComp professors around the world. Those that answered, we sent a link to an online survey and also asked them to send an e-mail to their students with a similar questionnaire, specially created for students.

Fig. 1 presents the main issues raised by our online survey Professors identified four major challenging topics: Context-Awareness[6], Location-Based Systems[7], Middleware[8], and Adaptation[9]. Students pointed out Middleware and Context-Awareness theme as the most challenging topics to understand. Few professors included programming exercises in their classes. The majority of them propose application projects to their students, using support platforms such as Android, iOS, Arduino, Raspberry, and the Aware Framework (Ferreira *et al.*, 2015). We also asked them what features a teaching environment for UbiComp should have. Answers included the usage of sensors, simple simulators, and integration with real devices as essential features of a tool which aims to support their classes. We present detailed information about this survey in the

[5] `https://www.researchgate.net`

[6] Context-Awareness is the ability of a system to modify its behavior according to changes in the user's context.

[7] Location-based systems or services (LBS) refers to software-level services that deliver content and data according to the user's location (e.g., map-based service, car navigation system).

[8] Middleware is a computer software that provides services to upper layers (i.e., software applications) beyond those available from the operating system. In Ubicomp, middleware provides facilities to developers such as context management, recommendation services, device discovery services Maia et al. (2013).

[9] Adaptation refers to the ability of a system (adaptive system) to adapt its behavior (e.g., content, features, data presentation) to each user based on information about the user profile and her context of use.

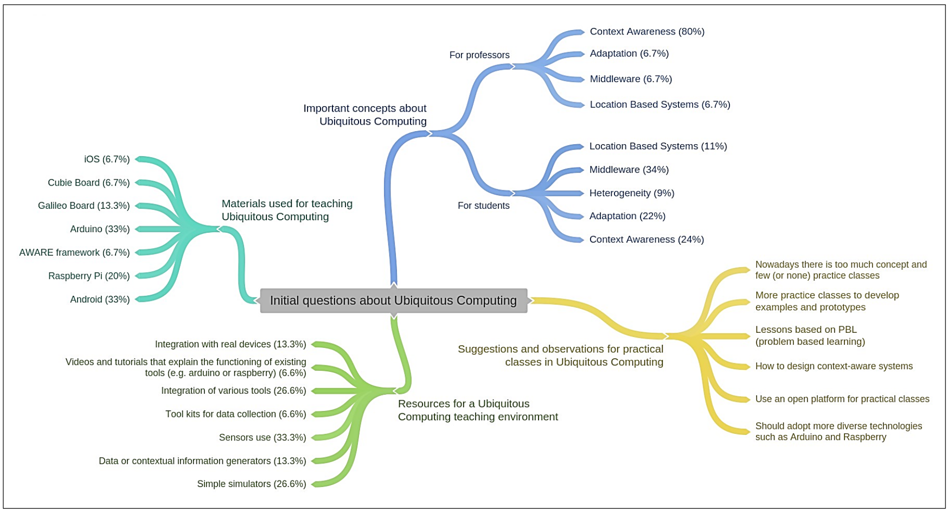

Initial questions about Ubiquitous Computing
Important concepts about Ubiquitous Computing
For professors
Context Awareness (80%)
Adaptation (6.7%)
Middleware (6.7%)
Location Based Systems (6.7%)
For students
Location Based Systems (11%)
Middleware (34%)
Heterogeneity (9%)
Adaptation (22%)
Context Awareness (24%)
Materials used for teaching Ubiquitous Computing
iOS (6.7%)
Cubie Board (6.7%)
Galileo Board (13.3%)
Arduino (33%)
AWARE framework (6.7%)
Raspberry Pi (20%)
Android (33%)
Suggestions and observations for practical classes in Ubiquitous Computing
Nowadays there is too much concept and few (or none) practice classes
More practice classes to develop examples and prototypes
Lessons based on PBL (problem based learning)
How to design context-aware systems
Use an open platform for practical classes
Should adopt more diverse technologies such as Arduino and Raspberry
Resources for a Ubiquitous Computing teaching environment
Integration with real devices (13.3%)
Videos and tutorials that explain the functioning of existing tools (e.g. arduino or raspberry) (6.6%)
Integration of various tools (26.6%)
Tool kits for data collection (6.6%)
Sensors use (33.3%)
Data or contextual information generators (13.3%)
Simple simulators (26.6%)

Fig. 1. Initial questions about Ubiquitous Computing.

paper (Peixoto *et al.*, 2016). These answers guided us in the design of LUCy, giving priority to the integration with real devices to better explain the concepts of Middleware and Context-Awareness.

2.2. *Teaching Ubiquitous Computing: Experience Reported in our University*

Two authors of this document are professors and have been teaching a course about Mobile and Ubiquitous Computing (MUC) since 2011. Along all these years, they performed nine editions of this subject, with the participation of undergraduate and graduate students. Each year, the discipline has been redesigned to incorporate innovative approaches to improve the exchange of information with students. Lucy is one of these initiatives.

Each edition of our MUC course had a very heterogeneous audience. These editions had students from three different graduations: Computer Science, Computer Engineering and Digital Media Engineering. Graduate Students are either from Master and Doctorate courses of the Computer Science program. Although MUC has a robust multidisciplinary appeal (Girouard *et al.*, 2018), the course focused on UbiComp Support Platforms (e.g., middleware, frameworks) and the software development process of ubiquitous applications on mobile devices.

In the first editions of the course, we spent a reasonable amount of classes for teaching programming for mobile devices (Android development) and a review of Distributed Systems concepts that are relevant to Ubiquitous Computing, such as Communication Models, Middleware, and Event-Based Communication. This approach allowed first-semester students to join our course. In some course editions, we also included extra classes to help students on their programming skills.

However, this approach was not effective once students failed to handle an excessive amount of classes (extra and regular ones). This scenario has led to a wrong understanding of Ubicomp concepts due to students focused on mobile devices programming tasks.

We changed this approach since 2016. From this year, both Mobile Device Programming and Distributed Systems disciplines were defined as requirements to attend our MUC course. It is why we need tools focusing on teaching Ubicomp to improve class understanding. Currently, MUC course follows a sequence with five major parts:

- **Ubiquitous Computing overview.** This topic introduces the presentation of Weiser seminal papers, the evolution of Ubiquitous Computing and its basic concepts (i.e., calmness, natural interaction, disappearance), and the intersections of Ubicomp with the Internet of Things, Human-Computer-Interaction, Wearable Computing, and Cyber-Physical Systems. We also analyze the advance of Ubicomp regarding what is a reality now and what is still very difficult to implement. To do that, we read and discuss some newest papers such as Caceres and Friday (2012); Oliveira *et al.* (2018).
- **Context-awareness and system adaptation.** In this part of our Ubicomp course, we discuss Dey and Abowd's articles, such as Dey and Abowd (1999). We introduce their classical definitions for both context and context-sensitive systems. Besides, we also show other definitions, such as those proposed by Viana Maia *et al.* (2013) and Jadwiga Indulska (Bettini *et al.*, 2010). Then students are introduced to the evolution of the state of the art in the representation, modeling, and inference of contextual information, based on surveys such as Yurur *et al.* (2016). This topic includes the understanding of approaches such as key-value pairs, ontologies, and inference of higher-level contextual information using Machine Learning techniques. After that, we present heterogeneity and adaptation challenges related to adaptation requirements in mobile and ubiquitous applications. Then, we present current solutions for adaptation in context-sensitive systems (Krupitzer *et al.*, 2015), including the simplest solution such Alternate Resources from Android[10] to more elaborate frameworks such as MAPE-K (Krupitzer *et al.*, 2015). Last but not least, we introduce middleware for context-aware and ubiquitous applications using tools such as the Context Toolkit Framework (Dey and Abowd, 1999) and Google Awareness.
- **Ubicomp practices.** To complement the theoretical classes, we intercalate them with practical activities. These activities may include experimenting with new technologies and Ubicomp concepts. LUCy simulation is an example of those experimentation classes. Also, we use HP Reveal to experience Augmented Reality features of mobile devices and the Node-RED to illustrate sensor-actuator behavior. Other practices include programming activities in Android, such as Alternative Resources and Google Awareness classes. Also, we give to students' applications that use MQTT and LoCCAM middleware (Maia *et al.*, 2013) to be completed.

[10]`https://developer.android.com/guide/topics/resources/providing-resources`

- **Research seminars on advanced topics on UbiComp.** In this part of the course, students form groups (3 or 4 students), and they have to find and present a recent academic paper related to the Ubicomp concepts. For instance, they frequently chose Pervasive Games, Mobile Cloud Computing, Ubiquitous Learning Systems, Ubiquitous Computing and Accessibility technologies, and Context-aware middleware.
- **Ubicomp application project.** At the end of MUC, students must present a prototype of a Ubicomp system. They are free to choose the thematic. We only establish that the system should have context-aware behavior and that smartphones should be involved in the system (Chavoshi and Hamidi, 2019).

3. LUCy – Learning Ubiquitous Computing Easily

LUCy assists the professor in three distinct ways. First, LUCy may serve as a repository for theoretical lesson materials on the subject of UbiComp. Second, students and professor can perform the simulation of theoretical UbiComp concepts during in-class activities. Finally, LUCy can also assist professors in programming practices related to the subject being taught.

LUCy architecture is divided into two main components: (*i*) a Web module and (*ii*) a mobile module. In the Web module, students and professors have access to reading material and the simulator manager. So, in a web browser, users set and manage the simulation configuration. In the mobile module, users run their environmental sensing app on their smartphones, using the device sensors. Users also interact with the running simulation, according to the previous configuration settings defined in the Web module.

We designed LUCy simulator targeting in-class activities based on the concept of Microlabs (Kurtz *et al.*, 2012). Microlabs approach, adopted by Kurtz *et al.* (2012), advocates that actions performed in the classroom must be short (5–15 minutes), and they must occur during class time. Students may form groups do perform the activities, and also, they should answer short tests about the topics under study right after the exercises. This assessment must give constructive feedback to students about the provided answers.

To use LUCy, each participant creates a user and password to access the system. After accessing the environment, users find the concepts related to UbiComp available in their Home screen. Upon accessing one of the available concepts, each student or professor is taken to an area divided into three sections: (*i*) Lesson, (*ii*) Simulation, and (*iii*) Practice[11]. LUCy was implemented using HTML and CCS for its front-end, Node.js as a backend server, and MongoDB as its database.

Currently, LUCy implements only the concept of Context-Awareness. However,

LUCy can be extended to implement other topics related to ubiquitous computing. The next subsections present more details of this first implementation.

[11] A video in URL `https://1drv.ms/v/s!AnfTpOzL2MeOiOpnBbtiU5Gbr_bI_Q` shows these sections.

3.1. *LUCy Prototype refinements and evaluations*

After applied the online survey, we started to design and development of our tool. Fig. 2 present LUCy's timeline. First, we designed an initial interactive Web page to explain Context-Awareness theme. We evaluated this interface in 2016, which showed some layout issues that created problems for the user's understanding of the lesson (in this case, the role of the main layers of a Context-aware middleware). We then modified this initial version to provide a Web system with a friendlier user interface and started to connect it to the student's mobile phone. We applied this new version in an undergraduate course to get information about its applicability. According to the evaluation results, we created the final version of LUCy with a more minimalist interface and fewer bugs in the connection mobile-Web tool. This version was evaluated three times between 2017–2018, whose results we describe in this paper.

3.2. *Context-Awareness Lesson*

Context-Awareness is the ability of a system to modify its current state due to changes in users context (Maia *et al.*, 2013) (e.g., user location, user mood, execution environment), without their direct intervention. Context-awareness is a key feature in ubiquitous computing and, as we identified in our online survey, also is a relevant topic in UbiComp lessons. Teaching Context-Awareness frequently includes: (*i*) to explain the definition of "context", (*ii*) to show how to model and represent it, and (*iii*) to study some middleware platforms that support the development of context-aware systems. The task of gathering contextual information can be made easily by adopting a context management middleware, which deals with most of the challenges in this domain (e.g., sensor configuration/ access, data aggregation, inference mechanisms, geofencing). Google Awareness is an example for this kind of middleware.

In Context-Awareness domain, Dey and Abowd's papers represent the seminal work (Dey and Abowd, 1999). Their definition of Context and Context-Awareness is the most cited among others found in the literature. They also proposed the Context Toolkit Framework, a classical architecture to support Ubiquitous Systems, which is the base for

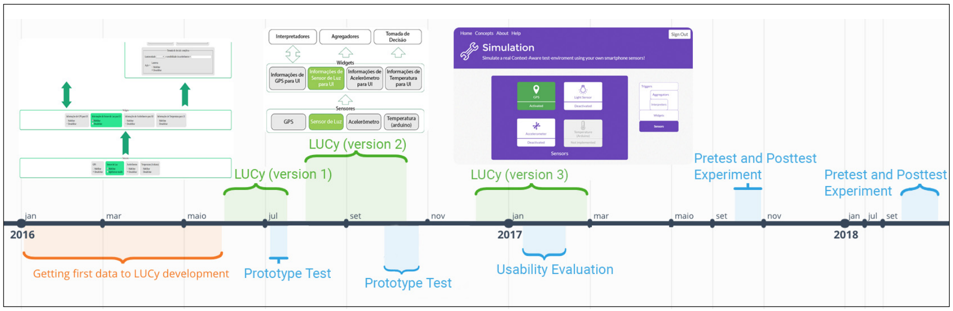

Interpretadores
Agregadores
Tomada de Decisão
Widgets
Informações de GPS para UI
Informações de Sensor de Luz para UI
Informações de Acelerômetro para UI
Informações de Temperatura para UI
Sensores
GPS
Sensor de Luz
Acelerômetro
Temperatura (arduino)
Simulation
Simulate a real Context-Aware test-enviroment using your own smartphone sensors!
Sensors
LUCy (version 1)
LUCy (version 2)
LUCy (version 3)
Pretest and Posttest Experiment
Pretest and Posttest Experiment
jan
mar
maio
jul
set
nov
jan
mar
maio
set
nov
jan
jul
set
2016
2017
2018
Getting first data to LUCy development
Prototype Test
Prototype Test
Usability Evaluation

Fig. 2. Development steps of LUCy.

other middleware platforms, such as Google Awareness and Aware (Ferreira *et al.*, 2015). We adopted the Context Toolkit as a start point for the Context-Awareness lesson. It is important to introduce the concepts of Sensors, Widgets, Interpreters, Aggregators, and Triggers to better understand the Context Toolkit framework. In the Lesson section of Context-Awareness, there is an explanation of the Context Toolkit. There are also links that lead to the previous documents of the authors responsible for building the Context Toolkit (Dey and Abowd, 1999). Professors can also insert other documents and projects related to the theme and divide the section into main and complementary materials.

We also extend the definitions of Dey and Abowd with the work of Coutaz (Coutaz *et al.*, 2005) to better illustrate the layered architecture of Context-Aware systems. There are various Ubicomp works developed based on those architectures, which makes studying them even more critical. Some of these works include Veiga *et al.* (2014); Duarte *et al.* (2015); Ferreira *et al.* (2015); Wendt and Julien (2016).

3.3. *Context-Awareness Simulation*

Our simulation is an interactive layered architecture. It has similar layers of the Context Toolkit framework and the reference architecture presented by Coutaz *et al.* (2005). We suppose that these abstractions help the understanding of context-awareness topics (e.g., architecture, principal components, rule-based execution) by the students.

Fig. 3 shows the structure proposed in this work. *Sensors* make the most basic layer. When selected, the tool shows which sensors are available. On top of it, we have the *Widgets* layer. It encapsulates information regarding the user context, such as his location or activity. Widgets layer also generates an interface for the application, hiding the implementation details. On top of it are the *Interpreters*, which gather information from the low-level context, such as luminosity data in lx (lux), and transform them into high-level context data, such as a horizontal bar that is filled and emptied according to luminosity changes. There is also the *Aggregator* layer, which merges contextual information, such as location and luminosity, to generate new contextual information (e.g.,

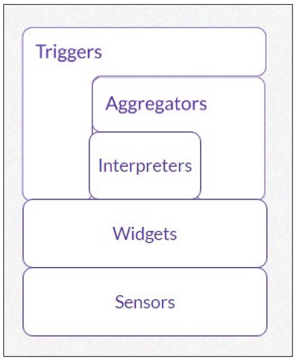


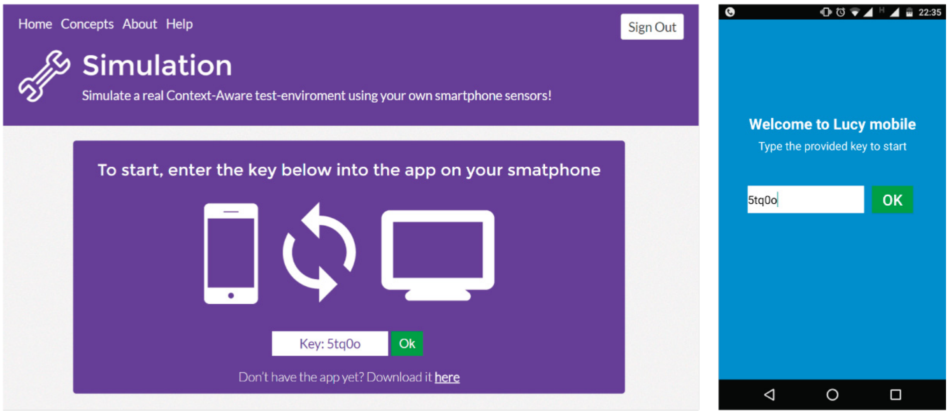
Fig. 3. Layered architecture proposed by LUCy.

user's location safety level). Lastly, are the *Triggers*. In this layer, users define a context situation to trigger an event or action. The Context-Awareness simulation requires some configuration to simulate the behavior that a context-aware system might have. In other words, choices are made in the Web tool to make the mobile application capture and use the user's current context to adapt itself according to actions specified by the user in the web tool. This simulation has two parts: the Web site, which is responsible for configuring the information and creating contextual rules, and the mobile side, which presents results according to settings made in the Web interface.

The user must insert the access key provided by the simulation (Fig. 4 (a)) into the text box on the mobile app (Fig. 4 (b)) to connect the mobile application with the Web tool. The simulation Web interface has a layered structure, in which users enable, disable, and configure sensors according to their objectives.

Also, users can insert parameters to change the context-aware decision-making in this structure. In this part of the environment, users can also create new contextual rules. With all settings made in the Web simulation, users can view and test their effects in the LUCy mobile application. Fig. 6 shows one example of a user interaction. When the Sensor layer is selected, the tool offers the lower level sensors connected to the simulation. Students or professor can activate or deactivate them. If any of these sensors of the Sensor layer is activated, then the sensors of the next layer (Widget layer) will be available to be activated.

As we stated, the simulator has five layers in this Web version. The *Sensors* layer shows the available sensors for running the simulations, which can be enabled or disabled by the user. For instance, in Fig. 5, three different sensors are available: GPS, light sensor, and accelerometer. We did not yet implement the temperature sensor, and consequently, it was disabled. The Web interface is only showing a place for its future implementation. Initially, GPS captures the user's current location. Users can modify it by inserting new coordinates into the Web simulation interface. The *Widgets* layer is responsible for dis-



Simulate a real Context-Aware test-enviroment using your own smartphone sensors!

To start, enter the key below into the app on your smatphone

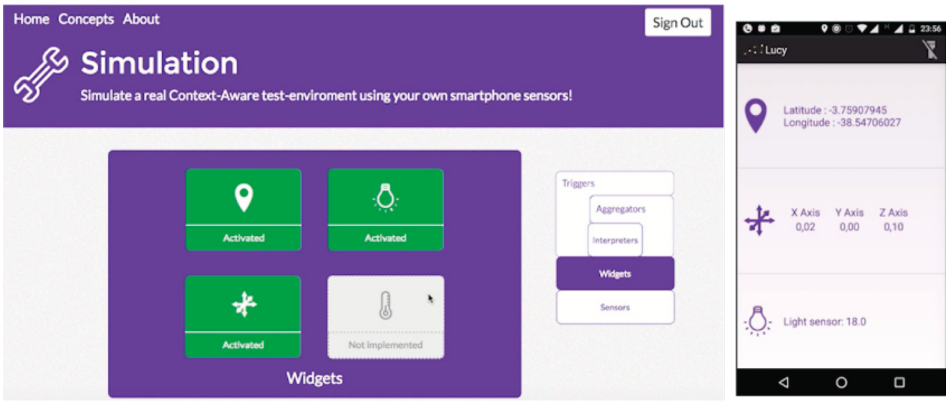
Don't have the app yet? Download it here

Welcome to Lucy mobile

Type the provided key to start

(a) (b)

Fig. 4. Connecting the mobile application to the Web simulation tool.



(a) (b)

Fig. 5. Enabling sensor information in the widget layer.

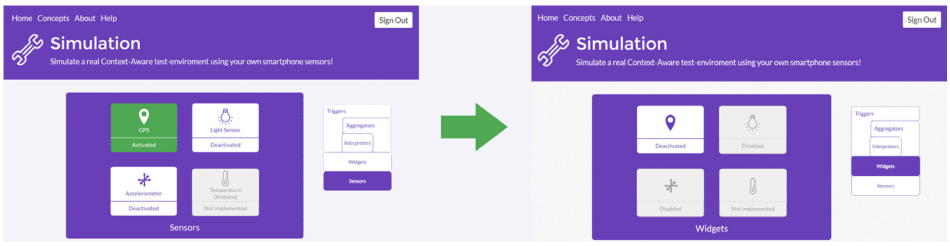


Fig. 6. Example of user interaction.

playing, in the user's mobile application, data acquired by the enabled sensor. *Aggregators* merge data from different sensors to create higher-level contextual information (e.g., combine accelerometer data and GPS location). The *Interpreters* layer uses low-level data and produces a piece of high-level information based on these raw data (e.g., the street name where the user is). In the *Triggers* layer, low-layers' data can be combined (e.g., data from Interpreters and Aggregators layers) to create contextual rules. For instance, a student can establish that if a user is at a particular street, then he is at an unsafe place. In this case, the smartphone must turn on the flashlight to show the rule was satisfied.

Mobile LUCy is an Android application used by the students to interact with the simulation. We chose this platform since it has no costs for mobile apps development, and most of our students have Android-based smartphones. Some professors who answered our online survey also use Android as the primary platform for the practical projects they propose in their Ubicomp classes.

In this part of the LUCy environment, users observe the behavior of a context-aware system according to the settings they made in the Web environment. The mobile system initially has no information in the sensors area (Fig. 7 (a)). Fig. 7 (b) depicts that the system displays the enabled sensors when the user activates them in the Web tool. However, no data is shown, since the widgets, which make the connection between the sensors and the application are not activated. Thus, for instance, if the user enables the brightness sensor inside the *Widgets* layer in the Web system, as shown in Fig. 5 (a), values in lumens (lx) appear in the LUCy's mobile application. The brightness values change according to the light intensity variation in the real user's environment captured by its smartphone sensor, as shown in Fig. 5 (c).

When the user enables an Interpreter in the Web tool, the mobile app will show its effects. For instance, Fig. 7 (d) has a progress bar that represents the interpretation of luminosity together with a classifier of the level of light. It varies between "no light", "dim", "low" and "bright". The image also displays an animation that interprets the accelerometer when the device is shaken, in addition to the user location map.

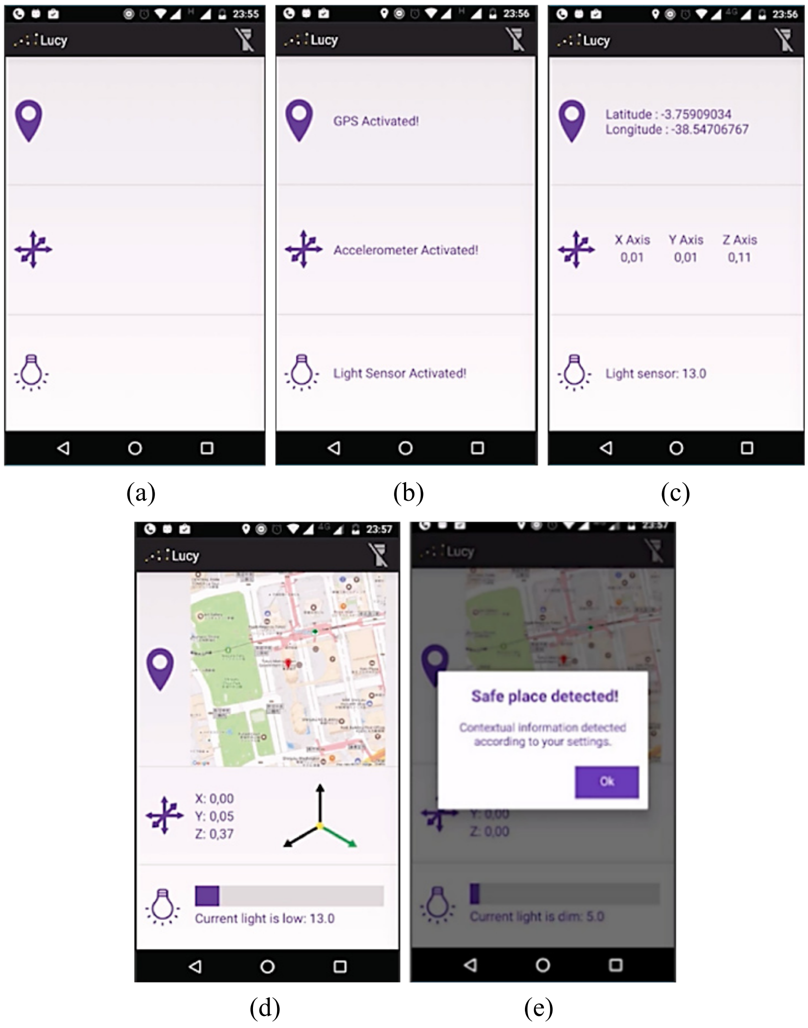


Fig. 7. Mobile LUCy screens. (a) Initial sensor area. (b) Sensors enabled. (c) Sensor data in the Widgets layer. (d) Sensor data in the Interpreters layer. (e) Safe context recognition.

In the Fig. 7 (e), there is an alert indicating if the user is at a safe or unsafe place – this classification depends on both the Aggregator that the user set in the Web tool and the rule the user defined in the Triggers layer. The user can also set a contextual rule to turn on the device's flashlight, in the absence of light or when the user is at an unsafe place. The mobile application will execute this action if the contextual rule is satisfied. We provide a demonstration video[12] where this behavior is shown.

3.4. *Context-Aware Practice*

In the area called Practice, there are links to papers with related subjects and questionnaires that guide both the simulation and the practical programming activity. The programming activity is an incomplete Android project, which contains a similar structure of the Context Toolkit framework. Students must find bugs in a Trigger implementation of this project and adds a new implementation for three elements to it: a Sensor, an Interpreter, and an Aggregator.

4. Evaluation

4.1. *Evaluation Goals*

As we mentioned in the Methodology section, we evaluated LUCy to improve its design, fix its bugs, and get students' feedback in the course of Mobile and Ubiquitous Computing, in each semester since 2016. Table 1 describes some information about the formal evaluations we made it since Lucy prototype (version 1). In the first two (P1 and P2) assessments, our goal was to get insights to redesign the tool. The other four assessment sessions (S1, S2, S3, and S4) used the same LUCy version, which we developed after the first two evaluations. The primary goal of these last four sessions was to measure the learning impact when using LUCy in practical lessons, especially its simulation feature. The assessments also measured LUCy usability and gather insights about its applicability in other UbiComp classes (i.e., beyond the Context-Awareness lesson). In this paper, we describe the procedures and results of the last four sessions.

4.2. *Sample Users*

We carried out three evaluation sessions with students on the two semesters of 2017 (S1 and S2) and the second semester of 2018 (S3) in the classes of the "Mobile and Ubiquitous Computing – MUC". This course is taught in the Computer Science program and the Digital Media Engineering program, both from the same university.

[12] `https://1drv.ms/v/s!AnfTpOzL2MeOiOpnBbtiU5Gbr_bI_Q`

Table 1
Six assessments carried out with LUCy

Test	Type	Period	Users	Methods
P1	Prototype Test	2016.1	8 undergraduate students	Learning Object Evaluation (Reatequi *et al.* 2010)
P2	Prototype Test	2016.2	6 undergraduate and graduate students	Learning Object Evaluation (Reatequi *et al.* 2010)
S1	Usability Evaluation and Learning Impact	2017.1	5 undergraduate students	SUS, Pretest-Posttest exam
S2	Usability Evaluation and Learning Impact	2017.2	14 undergraduate and graduate students	SUS, Pretest-Posttest Quasi-Experiment
S3	Usability Evaluation and Learning Impact	2018.2	13 undergraduate and graduate students	SUS, Pretest-Posttest Quasi-Experiment
S4	Usability Evaluation and Applicability	2017.1	6 professors	SUS, semi-structured interview

The first session (S1) was composed of five undergraduate students from the Digital Media Engineering course. This session lasted for approximately 1 hour and 45 minutes. The second one (S2) was held in a mixed class of undergraduate and graduate students, composed of five undergraduate students from the Computer Science program and nine graduate students from the masters and doctoral program in Computer Science (MDCC). This evaluation lasted about 1 hour and 55 minutes. The third session (S3) was composed of three undergraduate students from the Digital Media Engineering program and ten graduate students from MDCC in the same class. It lasted for about 1 hour and 40 minutes.

We also assessed LUCy with Ubiquitous Computing professors (S4). In total, six professors from five different universities tested our tool. All professors have a research background on Ubiquitous Computing. Some of them teach classes with UbiComp subjects. All of them have much experience with the subject matter. They evaluated LUCy taking into consideration its usefulness, applicability, usability, and user interface.

4.3. *Materials and Methods*

Students performed the tests on iMac computers (Intel Core i5 2.3 GHz processor, 8GB RAM, and 1TB hard disk). They accessed the Web module using either Safari, Chrome, or Firefox browsers. The students used the LUCy mobile application with their own devices (Android smartphones running version 4.0 or later). We also used a projector to display the professor's computer screen, so that students could see the Web module while the professor set up the rules of the simulation.

We used Google Forms tool to collect data about the students' profile and previous knowledge about Context-Awareness. We also used another Google Form to evaluate LUCy and the acquired knowledge after the practice session.

Regarding the tool evaluation, the students assessed usability, user interfaces, and pedagogical aspects present in the system. We used the System Usability Scale – SUS (Brooke, 2013) to measure LUCy's usability and the instruments proposed by PETESE (Coomans and Lacerda, 2015) to evaluate the pedagogical aspects of our approach. In both cases, students used a Likert scale to indicate their level of agreement with the statements given with options ranging from 1 to 5, in which one means "Strongly Disagree" and five means "Strongly Agree".

Assessing "learning impact" is notoriously difficult. In our study, we implemented a quasi-experimental design to get insights concerning the contribution of LUCy simulation in student learning about Context-Awareness. We used the One-Group Pretest-Posttest (Allen, 2017) design to observe the learning impact of our Context-Aware simulation. In the last class before the experiment, MUC professors gave a lecture about Context-Awareness. In the next class, students answered the pretest and followed the simulation as we explain in the next section. The pretest and posttest had the same three open questions concerning Context-Awareness concepts (i.e., the questions concern Dey's Context Toolkit components and their relation to the user's context gathering and processing). Both two professors of the MUC course corrected the students' tests, who answered the tests anonymously. Professors did not know if the answers were from the pretest or posttest. We conducted these actions to provide anonymization to the answers provided by the students. We calculated the average of these answers to perform the experiment analysis.

4.4. *Procedure*

We used LUCy to assist in practical classes approaching the Context-Awareness topic of the Mobile and Ubiquitous Computing – MUC course. The practices occurred after professors addressed this topic in a lecture class (i.e., in the same week or a week after the theory class). Before we started the practice with LUCy, the students were required to fill an online quiz about their personal information. Fig. 8 shows the sequence of steps we performed in the student assessment.

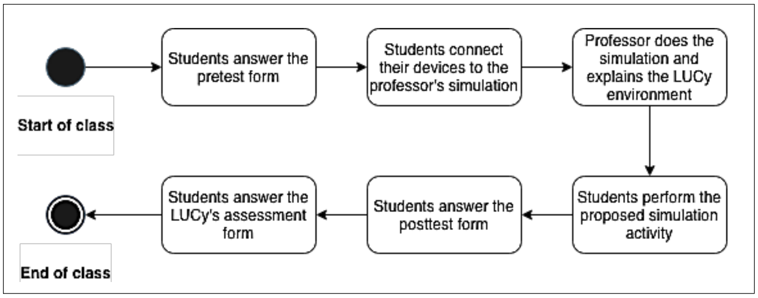

Start of class
Students answer the pretest form
Students connect their devices to the professor's simulation
Professor does the simulation and explains the LUCy environment
Students perform the proposed simulation activity
Students answer the posttest form
Students answer the LUCy's assessment form
End of class

Fig. 8. Stages performed in practical classes with LUCy.

First, students answer the pretest; then the professor connects his computer to a projector to present what is happening in his Web module of the learning tool and how they can configure it. After that, students connect their mobile devices with the professor's Web module, using a unique shared key provided by the system for synchronization. Hence, when the professor changes Lucy configuration, all students will receive these changes in their devices.

In the following activity, the professor demonstrates how to use the tool. Next, the students answer the simulation activity with LUCy, individually. After completing the simulation report, the students answered the same test applied at the beginning of the class (i.e., in a posttest form). Lastly, students also fill LUCy's assessment questionnaire.

The evaluation with professors was either performed in person or remotely through Skype when professors were from universities located outside our city. In both cases, professors accessed LUCy, adjusted some settings in the Context-Awareness simulation, and watched the results on their smartphones. Afterward, they answered questionnaires to evaluate the tool.

4.5. *Results*

4.5.1. *Usability and Acceptance*

In the three sessions (S1, S2, and S3), with 32 students in total, over 70% of the students stated that they had strong programming experience at the moment of the tests and had already used Android Studio for programming activities. Also, all students agreed that they needed a tool to assist them with the practice of ubiquitous computing concepts. Regarding the challenging Ubicomp topics to understand without practice classes, about 40% of the students answered that middleware was the most difficult topic, and about 29% of the learners stated the Context-awareness topic was the most challenger. A possible reason for this result is the absence of proper background on Distributed Systems by most of the students.

In the environment assessment questionnaire, the students evaluated the usability of the system as satisfactory, measured by the score of 75.4 points obtained according to SUS procedures Brooke (2013), indicating good usability of the software. According to this assessment, LUCy is in the "B" range, which includes applications with a usability score higher than 70 and less than or equal to 80 points.

For the three affirmations of the assessment of pedagogical aspects, as shown in Fig. 9, 20 students answered that they strongly agree with "I think the LUCy environment promote debate about the UbiComp topics between students and professors" and "I think the level of difficulty presented in the activities of the environment is appropriate for me (student)". Moreover, 22 students said they strongly agree with "I think that the technological resources of LUCy fit as mediators in the process of learning the concepts worked in the course".

For the interface evaluation (Fig. 10), 27 students said that agree with "the environment has an interface that makes its use friendlier". Sixteen students answered they strongly agree with "the interactive features of the environment allow me to change its

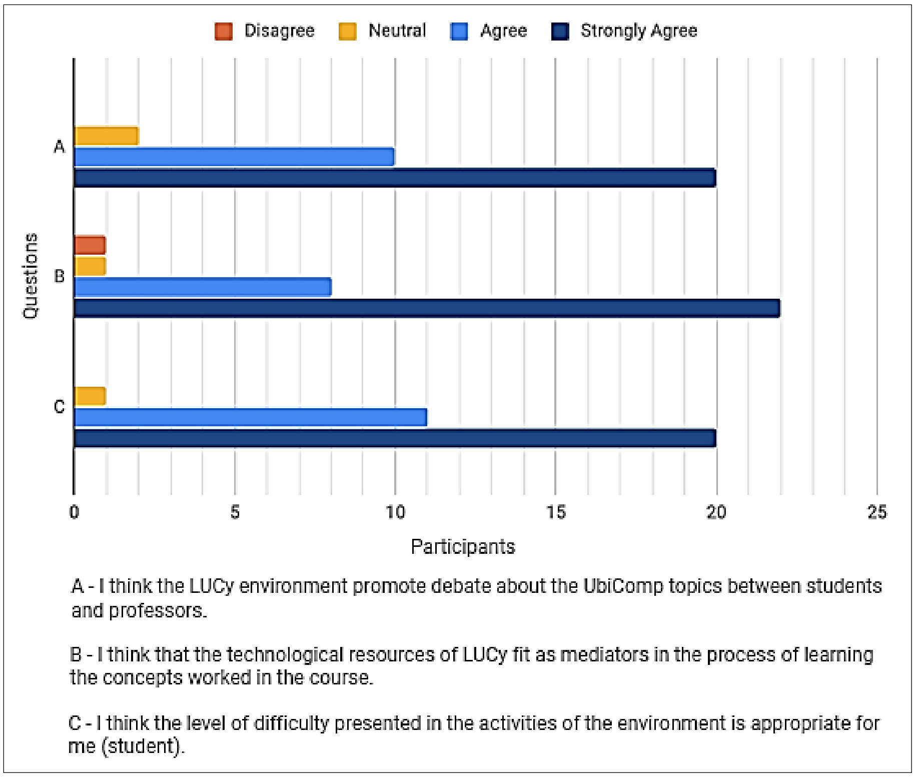


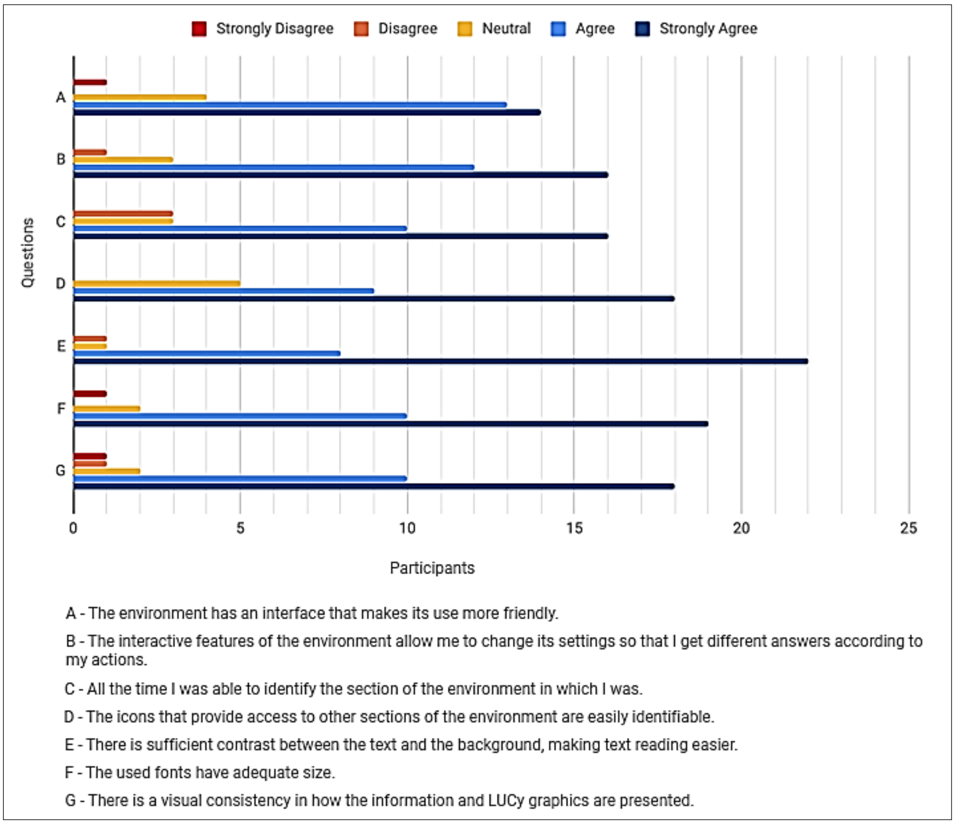
Fig. 9. Students answers about the evaluation of pedagogical aspects.

settings so that I get different answers according to my actions". The same number of learners strongly agree with "all the time I was able to identify the section of the environment in which I was". Besides, 18 students said that they also strongly agree with "the icons that provide access to other sections of the environment are easily identifiable", and "there is a visual consistency in how the information and LUCy graphics are presented".

4.5.2. *Professors Evaluation*

Regarding the results obtained with the professors, they classified the environment usability as very satisfactory, with a SUS score of 84.3 points. It means that the LUCy is on the "A" scale of SUS classification (> 80.3).

The results obtained through the satisfaction questionnaire show that all professors considered the activities proposed by Lucy acceptable to students studying Ubiquitous Computing concepts. In the results of the evaluation of pedagogical aspects, five professors responded that they strongly agree with "I think the LUCy environment promote debate about the UbiComp topics between students and professors" and "I think the technological resources of LUCy fit as mediators in the process of learning the concepts worked in the UbiComp course". Besides that, five professors also agree that the level of difficulty presented in activities of the environment is appropriate for students.



A - The environment has an interface that makes its use more friendly.

B - The interactive features of the environment allow me to change its settings so that I get different answers according to my actions.

C - All the time I was able to identify the section of the environment in which I was.

D - The icons that provide access to other sections of the environment are easily identifiable.

E - There is sufficient contrast between the text and the background, making text reading easier.

F - The used fonts have adequate size.

G - There is a visual consistency in how the information and LUCy graphics are presented.

Fig. 10. Students answers about the interface evaluation

Regarding LUCy's visual interface, Fig. 11 presents the professors' evaluation where they agree unanimously that with the statements "the environment has an interface that makes its use friendlier" and "the used fonts have adequate size". In addition, five professors trust that "the interactive features of the environment allow me to change its settings so that I get different answers according to my actions" and "there is a visual consistency in how the information and LUCy graphics are presented". Four professors also confirmed that "all the time I was able to identify the section of the environment in which I was", "the icons that provide access to other sections of the environment are easily identifiable" and "there is sufficient contrast between the text and the background, making text reading easier".

Professors said Lucy's proposal would be suitable for both undergraduate and graduate students. They considered our tool an exciting approach and suggested that LUCy should be shared with other professors. They also claimed that LUCy could be a starting point for the exploration of other UbiComp concepts. As a relevant comment, some professors reported that the LUCy is essential for teaching UbiComp concepts since there are few materials for this purpose. Besides, the proposal is simple to apply in the classroom. As a drawback, some professors reported that they were not comfortable with

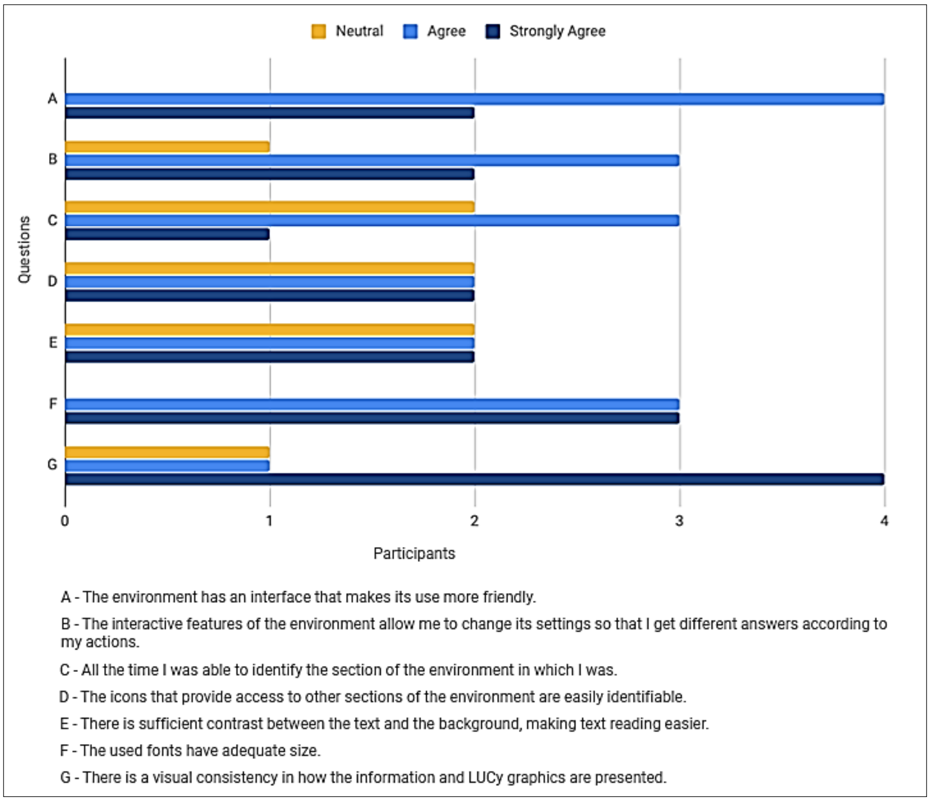


Fig. 11. Professors answers about the interface evaluation.

the colors used in the Web tool. Others suggested the configuration on the Web module should immediately affect the mobile application without having to wait for the state change of the corresponding sensors. We opt for this sensor-based change behavior to make our mobile implementation easier. After a sensor data change, the mobile system rechecks the rules defined in the Web module. Although, we can add a feature of context history, which will change the simulation according to the last data gathered without waiting changes in the sensor data.

4.5.3. *Assessment of Learning Impact*

In the session S1, we applied a pretest and posttest evaluation, but we did not make a score analysis. Our goal was to first understand this kind of quasi-experimental design and to get the first insights concerning students answers. Table 2 shows sample answers from some students to the question “How can you differentiate Interpreters over Widgets?”. In all responses, we perceive a better understanding of the Widget concept, which is confused with the Sensor definition frequently. Also, students provide an improved description of the process of interpreting low-level data (sensor data) into higher-level information.

Table 2

Sample of students' answers to one of the questions in the pretest and posttest questionnaire.

	Pretest	Posttest
Student A	The different sensors associated with them.	Widgets show low-level information, but through interpreters can be easily translated to high-level information.
Student B	Widgets receive information from the sensors (such as noise level) and transform them into meaningful information for the application (a meeting started in the room).	Widgets are interfaces for viewing sensor data. Interpreters use this data to derive information (e.g., low light, lots of light).
Student C	The first would be more sophisticated.	Widgets show raw information about the specific sensor, while interpreters refine this information and give feedback quickly recognized.
Student D	I do not know how to answer.	Widgets are interfaces to the raw data sources, which are in the lower layer. The interpreters use these interfaces and try to get some meaning from raw data.

In the session S2 and S3, we analyzed the pretest and posttest scores. As we explained in Section 4.3, both tests had no identification of its respondents. Only 23 of 27 students answered both pretest and posttest.

Fig. 12 shows a histogram of the pretest score. Only five students scored more than 7/10 in the pretest. In this case, the test measured what content students learned with the theory class. The students had a mean of 4.43 (SD = 2.568) in the pretest and 7.703 (SD = 1.913) in the posttest with a mean gain of 3.269 (SD = 2.510).

Fig. 13 compares pretest and posttest results. Only two students decreased their score: U21 and U22. U22 decreased from 5.25 to 2.5. After the score analysis, we talked with him about the tests, and he claimed he was in a hurry because he needed to leave the class early. So, he had to answer the test quickly. In the case of U21, he decreased from 8.33 to 7.83, but he still got a high score.

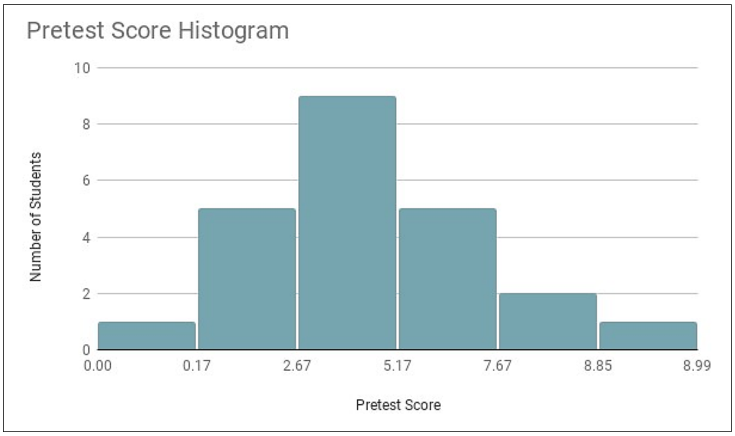


Fig. 12. Students' Performance in Pretest.

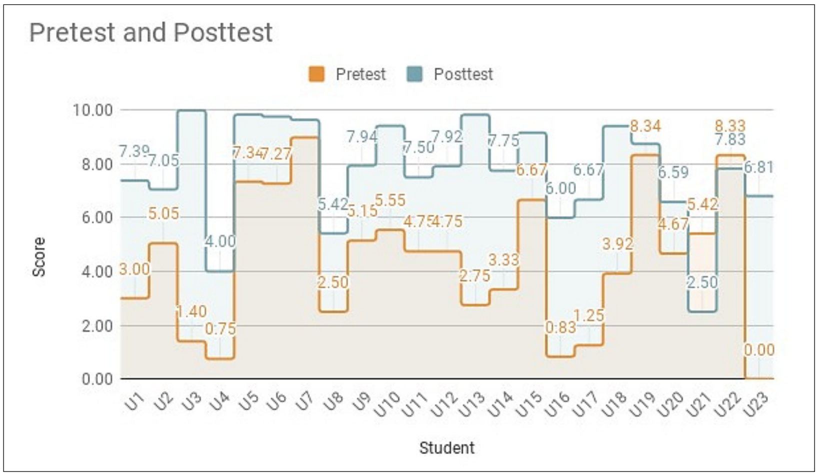


Fig. 13. Students' Performance in both tests.

For answer research question RQ2, we compared the pretest and posttest scores (Boxplots of the Fig. 14). We tested if these differences were statistically significant. Since both groups are dependent, we use a Paired T-Student test, right-tailed. As the test requires normalized data, we applied the Shapiro-Wilk test, which confirmed the normality (0.8709 with p = 0.05). Our Null hypothesis (H_0) was: $\mu_{Posttest} <= \mu_{Pretest}$, that is, the posttest score ($\mu_{Posttest}$) *is not* significantly *greater than* of the pretest score ($\mu_{Pretest}$). With the test results ($p - value = 0.00000188834$, $\alpha < 0.05$), H_0 was rejected with t equals 6.109631. In other words, the difference between the average of the Posttest minus Pretest is large enough to be statistically significant.

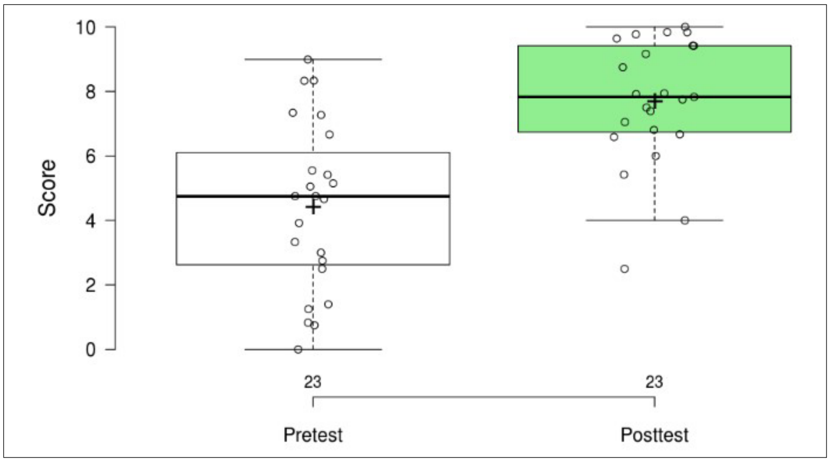


Fig. 14. BoxPlot comparing Pretest and Postest results.

5. Discussion

According to our research study, it was possible to understand what are the difficulties raised by students and professors in Ubiquitous Computing courses. In this sense, based on results presented in previous sections, we assert that simulations using students' smartphones sensors help them to better understand UbiComp concepts, such as Middleware and Context-Awareness.

Regarding research question RQ1, LUCy approach and its user interface were evaluated positively by students and professors according to SUS scores. The usability assessment demonstrated that users had pleasant interactions with the system, and the level of difficulty found for understanding the interactions in LUCy was not high. Students felt control over the tool, using interactive ways to change the configuration of the LUCy simulator.

The results of four sessions indicate that the pedagogical aspects were also considered positive. The students found the activities appropriated to that kind of topic and considered their level of knowledge compatible with the proposed activity.

According to the results of pretest and posttest quasi-experiment, we observe LUCy improves students' understanding of the Context-Awareness topic regarding the concepts covered by the tests (answering RQ2 positively). In our study, students had both theory and practical classes about Context-Awareness, with the last class contributing significantly to improve their understanding of Dey's Context Toolkit. Students understood better the functions played by the middleware layers of a context-aware application. Our simulation helps students to understand clearly the role of each context-aware layer since they observe the result in their smartphones when each layer is activated, and they can affect their behavior using the Web module.

5.1. *Difficulties in the Tool Usage and User Suggestions*

Students and professors had given very positive feedback about the environment in general, regarding its operation, its interface, and its usability. However, some evaluators had suggested adjustments and changes to refine further the proposal.

Some students faced problems to adjust the brightness sensor, as well as understanding the concept of brightness or darkness presented by the tool. Some students had troubles to keep their session active after the mobile application failed because when they restarted, they had to restart the simulation process from the beginning. This behavior suggests that the system needs to be modified to better deal with session states.

Some suggestions were made regarding a requirement for supporting other languages than English (e.g., Portuguese, Spanish). Students also suggested creating a tutorial video explaining the tool usage. This video will be accessed during the practice to avoid repeated explication of the professor. Students highly praised the system for practicing Context-Aware concepts. Although some students faced system errors during the practice (due to bugs in the mobile module), all of them finished the simulations required.

5.2. *Threats to Validity*

Some factors might cause bias in our study. One factor is the sample size of professors in our tests. Only six professors took part in the trial. Although these professors were from different universities, the sample size should be bigger for more definitive conclusions. Also, these professors did not use LUCy in their regular classes, which could give us a better perspective of the approach advantages and drawbacks.

Concerning the learning impact assessment, both pretest and posttest had the same open questions. In this case, a possible threat to validity is that students might have discussed these questions among each other and might have used LUCy with them in mind. Although they were not aware of having to do a posttest at the end of the evaluation session, this situation may have made it easier to answer the posttest. We decided to use open questions for minimizing this threat, trying to require from students a more in-depth description of their understanding. At last, we could have proposed more questions to check the knowledge of students regarding context-awareness concepts, but our experiment is already requiring two hours of the class.

6. Related Work

The related work exemplified in this section were extracted from a wider bibliographic search, made on Google Scholar[13], Springer[14], IEEE EXplore[15] and ACM Digital Library[16] databases, with the following keywords: "computing teaching environment"; "ubiquitous computing teaching environment"; "ubiquitous computing software"; "teaching environment for context awareness"; "teaching ubiquitous computing"; "context awareness teaching".

The searches with the more specific keywords about environments or software for teaching ubiquitous computing did not bring results similar works to LUCy proposal. In view of this, more general proposals of classes, methodologies, and materials used for the teaching of ubiquitous computing concepts were considered. Also, the search became more limited to the context awareness concept, with terms such as "context awareness teaching", for example. Some works from this search are described below.

Table 3 presents a summary of the tools used on the related work we found. We compare them, highlighting the characteristics relevant to this research. Complementary, Table 4 presents a comparison of the proposals and approaches of courses related to ubiquitous computing. Both tables are important to get an overview about how solutions and methods have been used in ubiquitous computing courses around the world.

In Richards *et al.* (2012), there is a description of a module, "My Digital Life (TU100)", designed to assist novice undergraduates in online education on the devel-

[13]`https://scholar.google.com.br/`
[14]`http://www.springer.com/br/`
[15]`http://ieeexplore.ieee.org/Xplore/home.jsp`
[16]`http://dl.acm.org/`

Table 3
Related Work Comparison

Study	Support multiple UbiComp concepts	Programming knowledge	Advantages	Disadvantages
SenseBoard (Richards et al., 2012)	Yes	Any Level	• It does not limit the type of UbiComp concepts supported. • It does not require prior knowledge in programming.	• SenseBoard hardware is no longer available. • The "Sense" language is unstable.
LEGO (Silvis-Cividjian, 2015)	Yes	Any Level	• It does not limit the type of UbiComp concepts supported. • It does not require prior knowledge in programming.	• Medium/high cost for acquisition and maintenance. • Limitation of the sensors and actuators types used.
Phidgets (Chalmers, 2015)	Yes	Required some programming skills	• It does not limit the type of UbiComp concepts supported. • It provides APIs in various programming languages.	• Medium/high cost for acquisition and maintenance. • There is no environment/ system that relates theoretical and practical classes.
Simulator of context-aware systems (Martin and Nurmi, 2006)	No - just context-awareness	Java programming knowledge	• It allows the illustrated visualization about studied concept. • It enables to student to create a simulation.	• It was not created for teaching-learning of ubiquitous concepts. • There is no interaction with the agents in the created simulation interface.

opment of ubiquitous programs. They playfully experiment with the concepts of ubiquitous computing using hardware called SenseBoard, which is programmable with the Sense language (a block language similar to Scratch[17]). The SenseBoard device, cited in Richards *et al.* (2012), is a board with sensors based on the Arduino microcontroller.

We observe that Richards *et al.* (2012) enables the teaching of ubiquitous computing through practical experimentation without requirements of previous knowledge in programming. The concepts covered in the practical activities are closely related to how location-based structures work in relation to the use of sensors and services. However, this work was discontinued since the SenseBoard hardware is no longer available for sale, and the Sense language is very unstable.

In Silvis-Cividjian (2015), authors approached the teaching of Pervasive Computing to students of the first year of a Computer Science course. Authors promote the teaching of Pervasive Computing through the development of pervasive projects, where students' context is inferred by recognition of standards, for example. Students are challenged

[17]`https://scratch.mit.edu/projects/editor/`

Table 4
Related Work Comparison (cont.)

Study	Local	Target audience	Division of classes	Resources used	Practical classes structure
Ubiquitous Computing Course (Chalmers, 2015)	University of Sussex, England	Undergraduate and graduate students	Theoretical classes, seminars, and laboratory practices	Phidgets	Development of targeted experiments using Phidgets.
Distance Learning ubiquitous computing Course (Goumopoulos et al., 2017)	Hellenic Open University, Greece	Distance graduate students	Theoretical and practical activities	Android, LEGO Mind-storms, Phidgets, Arduino, 123d circuits, Wireshark and remote lab with Arduino	Applications development with mentioned technologies and problems solving.
Environmental Intelligence Course (Corno et al., 2016)	Politecnico di Torino, Italy	Undergraduate student	Theoretical and practical activities	Raspberry Pi, Android, Web protocols and smart home systems	Practical activities in the classroom, guided assignments in laboratory and group work in the laboratory.

with the methodology of problem-based learning to find solutions that improve people's quality of life. In addition to designing and developing pervasive systems, students are led to demonstrate the features of their projects and how they have done so. In order to support these practical projects, there are laboratories equipped with LEGO Mindstorms NXT 2.0. This material (LEGO) was used in the construction of the physical part of the project to be controlled through programs and algorithms elaborated in MATLAB mathematical environment. Although it is robust, easy to program, mobile and scalable hardware, LEGO has a high cost and presents some limitation concerning available sensors and actuators. There is also a substantial limitation on the variety of pervasive systems or designs that can be built with this platform.

The work of Martin and Nurmi (2006) consists of a simulator that exemplifies the execution of contextual rules created for a specific environment with several differentiated and unique agents. The simulator for that work is part of a Java-based toolkit called Siafu[18]. This simulator was not specifically created for context-awareness teaching and learning. However, it has examples of simulations that are ready to be executed and visualized, as well as the possibility of extending the Java code to create other environments, agents, and rules. For creating new simulations, three classes of the original code need to be extended and defined: BaseAgentModel, BaseWorldModel, and BaseContextModel. A professor could use the ready-made examples to illustrate in the classroom what may be considered context, the elements that make up the context, and what happens in the face of the variation of this context. Despite this, there would be some limitation on the

[18]`http://siafusimulator.org/`

use of this material in class practices for other ubiquitous concepts, since the simulator in question is limited to context-awareness.

In Chalmers (2015), a scenario is described about an organization of disciplines that have concepts and practices of ubiquitous computing at the University of Sussex, in the United Kingdom. These classes are for undergraduate and graduate students and are organized in three parts: theoretical classes, seminars, and laboratory practices (Chalmers, 2015). In the theoretical classes, ubiquitous computing fundamental concepts and related state of the art works are presented. In the seminars, the objective is to critically analyze that theories and existing solutions to ubiquitous computing problems.

In laboratory practice, the goal is to program various devices to exercise the understanding of what has been seen in theory. The focus is on the design, execution, and analysis of experiments, with the subsequent presentation of the results obtained. Practical lab sessions last an average of two hours. The negative point of this work is the absence of an environment to support the practical classes to assist in the visualization and assimilation of the presented concepts. No tool concentrates theoretical and practical activities in the same space with the objective of relating them directly.

The work developed by Goumopoulos *et al.* (2017) presents a syllabus of a ubiquitous computing course created for the distance graduate of Hellenic Open University in Greece. Because it is a distance course, this curriculum has only five face-to-face meetings each academic year. In these meetings, theoretical problems and concepts related to ubiquitous computing are discussed. The proposal of Goumopoulos *et al.* (2017) includes the use of several technologies in the performance of theoretical and practical activities. Because it is a distance learning course, the use of an environment to concentrate the operations of the teaching-learning process is indispensable, but this work does not relate the links between theoretical and practical exercises within this environment.

The work of Corno *et al.* (2016) presents a course that, although it is not directly and specifically about ubiquitous computing, uses many concepts related to this area in its projects and definitions of intelligent environments. The course addresses several concepts of ubiquitous computing and has a clear structure for theoretical and practical moments, with a higher workload for practical activities, which is equivalent to 65% of the entire course. The proposal of a project development based on problem-solving by teams used in this course could also be applied to other programs that focus on teaching-learning concepts of ubiquitous computing.

The way we currently structure our MUC course is very similar to that proposed by Chalmers (2015). Students spend considerable time on the project to implement the ubiquitous system. The most significant difference is that we leave them free to use equipment and technologies in the design, and these materials may not have been addressed in the course (e.g., iOS, Arduino, Raspberry). The purpose of constructing LUCy is more in the sense of fill the gap that exists between theoretical classes and code implementation of UbiComp concepts. Being able to experience, configure, and view the outcome of contextual changes allow students to better understanding the Context-awareness before implementing it.

7. Conclusion and Future Works

The technological and research potential of UbiComp turn teaching its concepts a trend in both undergraduate and graduate courses. In this paper, we presented some insights about teaching UbiComp in a Computer Science perspective. Mainly, we introduced the design, implementation, and evaluation of LUCy, a tool to assist professors in teaching UbiComp concepts. At the current state, LUCy better contemplates the topic of Context-Awareness.

According to the conduction of the experiment and looking at the suggestions presented by the students on the assessment of LUCy, some improvement points were identified. For example, it is required to identify and give the key that is being used in the Web system after the simulation starts, so that if the cell phone disconnects from Web Module, the user can return to the session using the previous key. Regarding the tool usage, we need to improve its help and support material to be available to the students during the simulation, as well as the tool should support other languages (e.g., Spanish, Chinese, Portuguese) to enable students to interact more naturally with LUCy.

As future work, we intend to implement other interactive simulations to assist the understanding of ubiquitous concepts mentioned by professors and students, such as middleware and system adaptation.

References

Allen, M. (2017). The SAGE Encyclopedia of Communication Research Methods. SAGE Publications.

Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A., and Riboni, D. (2010). A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6(2),161–180. `https://doi.org/10.1016/j.pmcj.2009.06.002`

Brooke, J. (2013). Sus: A retrospective. *Journal of Usability Studies*, 8:29–40.

Caceres, R. and Friday, A. (2012). Ubicomp systems at 20: Progress, opportunities, and challenges. *IEEE Pervasive Computing*, 11(1),14–21. `https://doi.org/10.1109/MPRV.2011.85`

Cárdenas-Robledo, L. A. and Peña-Ayala, A. (2018). Ubiquitous learning: A systematic review. *Telematics and Informatics*, 35(5),1097–1132. `https://doi.org/10.1016/j.tele.2018.01.009`

Chalmers, D. (2015). Pervasive computing as a classroom-based course. *Pervasive computing – IEEE*, pp. 70–78.

Chavoshi, A. and Hamidi, H. (2019). Social, individual, technological and pedagogical factors influencing mobile learning acceptance in higher education: A case from Iran. *Telematics and Informatics*, 38(September 2018),133–165. `https://doi.org/10.1016/j.tele.2018.09.007`

Coomans, S. and Lacerda, G. S. (2015). Petese, a pedagogical ergonomic tool for educational software evaluation. *Procedia Manufacturing,* 3, 5881–5888.

Corno, F., Russis, L. D., and Bonino, D. (2016). Educating internet of things professionals: The ambient intelligence course. *IT Professional*, 18(6), 50–57.

Coutaz, J., Crowley, J. L., Dobson, S., & Garlan, D. (2005). Context is key. *Communications of the ACM*, 48(3), 49–53.

Dey, A. K. and Abowd, G. D. (2000, June). The context toolkit: Aiding the development of context-aware applications. In: *Workshop on Software Engineering for wearable and pervasive computing* (pp. 431–441).

Duarte, P. A., Silva, L. F. M., Gomes, F. A., Viana, W., & Trinta, F. M. (2015, October). Dynamic deployment for context-aware multimedia environments. In: *Proceedings of the 21st Brazilian Symposium on Multimedia and the Web* (pp. 197–204).

Ferreira, D., Kostakos, V., and Dey, A. K. (2015). Aware: mobile context instrumentation framework. *Frontiers in ICT*, 2, 6.

Girouard, A., Kun, A. L., Roudaut, A., Shaer, O., and Kun, A. L. (2018). Pervasive computing education. *IEEE Pervasive Computing*, 17(4),9–12.
`https://doi.org/10.1109/MPRV.2018.2878999`

Goumopoulos, C., Nicopolitidis, P., Gavalas, D., & Kameas, A. (2017). A distance learning curriculum on pervasive computing. *International Journal of Continuing Engineering Education and Life Long Learning*, 27(1–2), 122–146.

Krupitzer, C., Roth, F. M., VanSyckel, S., Schiele, G., & Becker, C. (2015). A survey on engineering approaches for self-adaptive systems. *Pervasive and Mobile Computing*, 17, 184–206.
`https://doi.org/10.1016/j.pmcj.2014.09.009`

Kurtz, B. L., Fenwick Jr, J. B., & Meznar, P. (2012, February). Developing microlabs using google web toolkit. In: *Proceedings of the 43rd ACM technical symposium on Computer Science Education* (pp. 607–612).

Lima, F. F. (2011). *Syssu – um Sistema de Suporte Para Computação Ubíqua* (Master's Thesis), Universidade Federal do Ceará, Fortaleza, Ceará, Brasil.

Maia, M. E. F., Fonteles, A., Neto, B., Gadelha, R., Viana, W., and Andrade, R. M. C. (2013). Loccam – loosely coupled context acquisition middleware. In: *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13, pp. 534–541, New York, NY, USA. ACM.
`https: //doi.org/10.1145/2480362.2480465`

Martin, M., & Nurmi, P. (2006, July). A generic large scale simulator for ubiquitous computing. In *2006 Third Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services* (pp. 1–3). IEEE.

Nieuwdorp, E. (2007). The pervasive discourse: an analysis. *Computers in Entertainment (CIE)*, 5(2), 13.

Oliveira, L. B., Pereira, F. M. Q., Misoczki, R., Aranha, D. F., Borges, F., Nogueira, M., Wangham, M., Wu, M., and Liu, J. (2018). The computer for the 21st century: present security & privacy challenges. *Journal of Internet Services and Applications*, 9(1), 24.
`https://doi.org/10.1186/s13174-018-0095-2`

Peixoto, M. J. P., Sarmento, W. W. F., and Viana, W. (2016). Ferramenta de apoio ao ensino prático de computação ubíqua. v 12, (pp. 333–343).

Reategui, E., & Finco, M. D. (2010). Proposta de diretrizes para avaliação de objetos de aprendizagem considerando aspectos pedagógicos e técnicos. *RENOTE-Revista Novas Tecnologias na Educação*, 8(3).

Richards, M., Petre, M. and Bandara, A. K. (2012, February). Starting with Ubicomp: using the SenseBoard to introduce computing. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education* (pp. 583–588).

Silvis-Cividjian, N. (2015, June). Teaching pervasive computing to cs freshmen: A multidisciplinary approach. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 195–200).

Veiga, E. F., Melo, G., & Neto, R. D. F. B. (2014, November). Apoio ao desenvolvimento de aplicações de tempo real sensíveis a contexto semântico. In *Anais Estendidos do XX Simpósio Brasileiro de Sistemas Multimídia e Web* (pp. 9–12). SBC.

Weiser, M. (1991). The computer for the 21st century. *Scientific American*, 265(3),66–75.

Wendt, N. and Julien, C. (2016). Mason: An open development contextual sensing framework enabling reactive applications. *In Proceedings of the International Conference on Mobile Software Engineering and Systems, MOBILESoft* '16, pp. 100–110, New York, NY, USA. ACM.
`https://doi.org/ 10.1145/2897073.2897099`

Wood, S., & Romero, P. (2010, November). User-centred design for a mobile learning application. In *Proceedings of the 3rd Mexican Workshop on Human Computer Interaction* (pp. 77–84).

Yürür, Ö., Liu, C. H., Sheng, Z., Leung, V. C., Moreno, W., & Leung, K. K. (2014). Context-awareness for mobile sensing: A survey and future directions. *IEEE Communications Surveys & Tutorials*, 18(1), 68–93.
`https://doi.org/10.1109/COMST.2014.2381246`

M.J.P. Peixoto is a Computer Engineering Ph.D student at Ontario Tech University, Canada. She holds a master's degree in Computer Science from the Federal University of Ceará, Brazil (2017) and a bachelor's degree in Digital Media Engineering from the Federal University of Ceará, Brazil (2014), with honors (Magna Cum Laude). She has experience in Computer Science, with emphasis on software development, multimedia systems, mobile and ubiquitous computing and game-based learning.

P.A. de S. Duarte is a doctoral student in computer science from the Federal University of Ceará currently. He holds a bachelor's degree in Computer Engineering from Federal Center of Technological Education of Ceará (2012) and a master's degree in Computer Science from the Federal University of Ceará (2014).

P.T. de Araújo is graduated in Computer Science from the Federal University of Ceará (2014) and is a master student in Computer Science from the Federal University of Ceará currently.

P.I. da C. Pinto is graduated in Digital Media Engineering from the Federal University of Ceará (2018).

W.W.F. Sarmento is adjunct professor at the Federal University of Ceará. Master in Teleinformatics Engineering from the Federal University of Ceará (2007) and graduated in Telematics from the Federal Center of Technological Education of Ceará (2003).

F.A.M. Trinta holds a PhD degree from Federal University of Pernambuco (2007). He is an adjunct professor at the Computer Science Department - Federal University of Ceará since 2011. He works mainly in the areas of Multimedia, Software Engineering and Distributed Systems. Currently, his research is focused on Mobile Cloud Computing, Context-Aware Computing and Fog Computing.

W. Viana is an associate Professor of UFC Virtual Institute and member of the Master and Doctorate in Computer Science (MDCC) program of the Federal University of Ceará (UFC). He holds a bachelor's degree in Computer Science from the Federal University of Ceará (2002), a master's degree in Computer Science from the Federal University of Ceará (2005) and a PhD (specialité Informatique) from the Université de Grenoble, France (2010). He has experience in the area of Computer Science, with an emphasis on Mobile and Ubiquitous Computing, Multimedia, Serious Games, and Software Engineering.