Received July 26, 2018, accepted August 30, 2018, date of publication September 24, 2018, date of current version November 9, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2871768

A Systematic Map for Improving Teaching and Learning in Undergraduate Operating Systems Courses

SONIA PAMPLONA[1], NELSON MEDINILLA[2], AND PAMELA FLORES[3]
[1]Department of Computer Science, Madrid Open University, 28400 Madrid, Spain
[2]Department of Information Systems and Languages and Software Engineering, Universidad Politécnica de Madrid, Campus de Montegancedo, 28660 Madrid, Spain
[3]Faculty of Systems Engineering, Escuela Politécnica Nacional, Quito 170517, Ecuador

Corresponding author: Sonia Pamplona (sonia.pamplona@udima.es)

ABSTRACT Operating Systems (OS) is an important area of knowledge included in virtually all undergraduate computing curricula and in some engineering curricula as well. Teaching and learning an OS undergraduate course have always been a challenge. Several different approaches have been used for OS teaching and learning. Nevertheless, it is not easy for a teacher to choose one of them. No guidelines are available on how to choose one of them to match the specific objectives of each OS course. The objective of this paper is to analyze the approaches that have been used to improve OS teaching and learning by applying a systematic map. In particular, we consider the following dimensions: learning objectives, assessment, empirical study, methodology, and mode (face-to-face, online, or blended). The systematic map devised in this paper is focused on the time span from 1995 to 2017 and considered six of the major publications on the Computer Science Education. We considered three journals (the *Journal of Engineering Education*, the IEEE TRANSACTIONS ON EDUCATION, and the *International Journal of Engineering Education*) and three conferences (the ACM Technical Symposium on Computer Science Education—SIGCSE, the Conference on Computing Education Research—ITiCSE, and the International Conference on Computing Education Research—Koli). A total of 55 papers were included in the study after performing a search based on the inclusion/exclusion criteria. Nine approaches to improve OS teaching and learning were identified and analyzed. Furthermore, the implications for OS instructors and for research in this field are discussed.

INDEX TERMS Computer science education, operating systems, systematic map.

I. INTRODUCTION

Operating Systems (OS) is an important area of knowledge included in the curriculum guidelines established by the ACM and IEEE Computer Society for undergraduate programs in computing [1]. Accordingly, an OS course is included in virtually all computing curricula (see [2]–[4]) and in some engineering curricula as well (see [5], [6]).

An OS is a software that manages resource sharing and provides programmers with abstraction to control the hardware. Primary OS concepts include concurrency, process management, scheduling, memory management, device management, file systems, and security. These concepts impart an understanding to students about the issues that influence the design of contemporary OS [7].

OS is a core course in computer science, and the understanding of other main subjects in this field heavily relies on the understanding of OS concepts and functionalities [8]. Moreover, many of the ideas and techniques used by OS designers can be applied to the field of software development. Therefore, the study of OS, which identifies the type of problems that designers face and their solutions, provides students with both the experience and knowledge required for their future careers as software engineers.

Teaching an undergraduate OS course has always been a challenge [9]–[11]. Furthermore, students have traditionally found it difficult to grasp OS concepts [12], [13], such as concurrency [11], [14], [15], virtual memory [16], and context switching [17].

Several approaches have been used to improve teaching and learning in OS courses and to overcome the aforementioned difficulties. Nevertheless, it is not easy for a teacher to choose one of them. No guidelines are available on how to choose one of them to match the specific objectives of each OS course. The present study intends to offer a comprehensive overview of the available information on OS education by creating a systematic map.

A systematic map study is a systematic review method that describes the nature of a field of research [18]. Systematic reviews help build theory and avoid recurrence of existing issues [19]. In particular, such studies can benefit engineering education because they synthesize prior work, inform practice, and identify new directions for research [20]. To the best of our knowledge, no systematic mapping study has been published in the field of OS education to date. One review was published, but it only referred to one approach to teach OS, namely the use of instructional OS [21].

The key contribution of the present paper is the map of approaches, contexts, methodologies, and results it offers that can be used by teachers and researchers to fulfill the following objectives: (1) to understand the current state of research in OS education, (2) to improve teaching and learning in this field, and (3) to identify gaps and new directions for research in this area.

II. RESEARCH METHODOLOGY

The aim of the present paper was to describe the nature of the research on OS education. In particular, the overall objective of our study was to identify the range and type of existing research studies addressing the improvement of OS teaching and learning. Therefore, we carried out a systematic mapping study. This type of review allows to gain an understanding of the purpose and extent of research activity in a given area [18]. The boundaries and purpose of our map were informed by both the review questions and the selection criteria defined below.

A. RESEARCH QUESTIONS

In order to obtain a detailed view of the topic under study, the systematic mapping study raised the following five research questions (one primary question and four secondary questions):

- Primary question: What approaches have been used to improve OS teaching and learning?
- Sub-Question 1. Does the study specify learning objectives?
- Sub-Question 2. What kind of assessment has been used in the study to evaluate the approach (no assessment, attitude and opinion assessment, or learning assessment)?
- Sub-Question 3. Does the study describe any kind of empirical research?
- Sub-Question 4. If an empirical study has been conducted, what research methodology has been used?
- Sub-Question 5. What kind of teaching and learning mode has been used (face-to-face, online, or blended)?

The primary question allowed us to categorize the selected studies and to identify future areas of research in the field. The sub-questions helped characterize each study.

Sub-question 1 aimed to determine whether the studies specify the learning objectives. Sub-questions 2, 3, and 4 sought to evaluate the effectiveness of the approach. Sub-question 5 identified the learning context of the studied course (face-to-face, online, or blended).

Summarizing, the answers to the research questions may be useful for both teachers and researchers engaged in engineering education. The answers may help teachers identify the level of suitability of the approach for a particular case, and researchers may be able to identify research trends and gaps in the research on OS education.

B. SEARCH STRATEGY

In order to include only the most relevant studies, we decided to restrict the body of literature for this map to six of the major publications on Computer Science Education, namely three conferences and three journals. The selection was carried out based on the CORE Computer Science Conference Ranking [22] for conferences and Journal Citation Reports (JCR) [23] for journals. The selected conferences were SIGCSE (ACM Technical Symposium on Computer Science Education), ITiCSE (Conference on Innovation and Technology in Computer Science Education), and Koli (International Conference on Computing Education Research). The selected journals were Journal of Engineering Education, IEEE Transactions on Education, and International Journal of Engineering Education.

In addition, we focused on a particular time span from 1995 to 2017. The first reason for selecting this time span was that the research published before 1995 refers almost exclusively to one approach to teach OS, namely instructional OS. In 1995, a greater number of approaches began to be used to teach OS. The second reason for selecting this time span is that technology has changed so much in the past 20 years that approaches prior to 1995 might not act as a reference for current teachers.

To construct the search string, we considered the need to find a balance between searching sensitively and searching precisely [18]. A sensitive search aims to identify as much relevant material as possible. However, usually, such searches retrieve large numbers of irrelevant studies. A precise search aims to identify a high proportion of studies that meet the inclusion criteria. However, this type of search can also result in exclusion of relevant studies.

First, we constructed a search string with a list of keywords to be found in the titles of some papers we already knew from previous research. Next, we included synonyms and alternative words. Thus, the first search string was as follows.

“operating system” AND (teaching OR learning OR course OR courseware OR simulator OR educational OR instructional OR approach OR academic OR project).

The limitation of this search string was that relevant studies may have been excluded because the main goal was to perform a precise search. Eventually, in order to increase sensitivity and given that the publications we searched pertained to engineering education, we simplified our search string and used the keyword "operating systems" to search in the title, abstract, or keywords.

C. STUDY SELECTION

The selection process aimed to identify papers relevant to the objective of this mapping study. Each article was retrieved by one author, and two others authors determined whether if it should be included. The inclusion and exclusion criteria are listed below.

1) INCLUSION CRITERIA

To be included in the review, the papers had to present an approach to improve teaching or learning in an undergraduate OS course in accordance with the main concepts presented in the ACM/IEEE curricula [1]: concurrency, process management, scheduling, memory management, device management, file systems, and security.

2) EXCLUSION CRITERIA

Papers were excluded if the main objective of the discussed OS course was to learn a non-OS such as Computer Security or Software Engineering. Papers were also excluded it they were not a full paper. Therefore, we excluded poster summaries, summaries of panel and demo sessions, and similar types of reports.

The final results of the searching process are presented in Table 1.

TABLE 1. Number of articles by publication.

Name of the publication	Search results	Included in the study
SIGCSE proceedings	239	39
ItiCSE proceedings	11	7
Koli proceedings	1	1
IEEE Transactions on Education	63	3
Journal of Engineering Education	2	1
International Journal of Engineering Education	7	4
TOTAL	**323**	**55**

D. DATA EXTRACTION STRATEGY

The papers were analyzed by using the ATLAS.ti software.[1] ATLAS.ti is a computer program which allows systematically code unstructured data in order to facilitate further analysis. The data extraction strategy was based on providing the set of possible answers to research questions.

Primary question: What approaches have been used to improve OS teaching and learning?

To answer this question, we analyzed the different approaches to improve OS teaching and learning used by the studies, then we gathered those into different categories.

[1]https://atlasti.com/

Sub-Question 1. Does the study specify learning objectives?

We analyzed whether the selected study specified educational objectives. We used a broad definition of objectives: what teachers want students to learn [24]. We consider that the study establishes the objectives if the purpose of the assignments or projects is explicit, regardless of whether the authors have used another term than "objectives."

Stating objectives explicitly is particularly important because we cannot always infer the objectives from an assignment. For example, in a software development assignment, we cannot know whether the main purpose of the assignment is to learn to develop software or to understand concepts.

The learning objectives of the assignments pertaining to each approach are necessary for teachers to decide if the approach is appropriate for their particular case. In addition, the objectives provide guidance for evaluating the approach.

Sub-Question 2. What kind of assessment has been used in the study to evaluate the approach (no assessment, attitude and opinion assessment, or learning assessment)?

We found the following kinds of assessments in the selected studies:

- No assessment: No assessment has been carried out.
- Attitude and opinion assessment. The study discovers the perception of students regarding the implemented teaching and learning approach.
- Learning assessment: The study verifies whether learning objectives have been achieved.

The most desirable method to evaluate the approach is assessing the learning because it may then provide evidence about the effectiveness of the approach.

Sub-Question 3. Does the study describe any kind of empirical research?

We checked whether an empirical research has been described in the selected paper.

Sub-Question 4. If an empirical research has been conducted, what research methodology has been used?

Instead of classifying the methodology into quantitative and qualitative research, we decided to provide more information by using a more precise classification scheme based on two dimensions: the purpose of the research and the nature of the research data.

- The purpose of the research. We follow the classification made by [25] with respect to the purpose of a research. These authors distinguished two kinds of purpose in education research. (1) confirmatory, when the main goal of the research is to confirm a hypothesis; (2) discovery, when the objective of the research is to discover new information.
  When we applied this classification scheme to our data we found that there were studies whose purpose did not match with a confirmatory purpose or a discovery purpose. These studies had merely a descriptive purpose. Hence, we created a third category called descriptive.
- The nature of the research data. In the studies we reviewed, there are educational experiences which are

converted into words or into numbers. Hence, we distinguished two types of data: (1) qualitative data, which refers to data in the form of words; and (2) quantitative data, which refers to data in the form of numbers.

Sub-Question 5. What kind of teaching and learning mode has been used? (face-to-face, online, or blended)

In this question, we identified the learning mode described in the study. The mode of learning can be classified into the following categories:

- face-to-face. A face-to-face course occurs ''in person'' and in real time between teachers and students.
- online. An online course is held over the Internet with the teacher and student separated geographically, using a web-based learning management system. It may be synchronous, when the participants interact in real time (e.g. web conference), or asynchronous when the participants' communications are separated by time (e.g., online forums) [26].
- blended. A blended course includes face-to-face interactions and online interactions.

III. RESULTS

This section describes the results to the formulated research questions. Tables 2–10 contains the data extracted from each of the 55 articles analyzed. We designed a table for each approach containing the information about the papers following that specific approach. In turn, the different fields considered according to the data extraction strategy described earlier were: aim of the study, specifies learning objectives, approach evaluation, empirical study, research methodology, teaching and learning mode.

1. Primary question: What approaches have been used to improve OS teaching and learning?

We categorized the selected studies into nine groups based on the different approaches reported in the research (Figure 1). The approaches are listed according to the time they started to be used to show their evolution

Real OS

In the real OS approach, students are involved in developing kernel code for an actual OS. The following have been proposed: UNIX, Linux, iPodLinux, and Android. The authors of the studies supporting this approach argued that the exposure to a real-world system helps students understand OS concepts [27], imparts an overall picture of an OS to the students, improves programming skills [28], improves students' motivation [29], increases their confidence in their abilities [28], and helps students get a job [29].

Instructional OS

The studies included in the Instructional OS approach propose the use of kernel programming projects that avoid the complexity of real OS. The objective is to provide students with an actual and easy-to-understand OS that they can study and modify. To achieve this goal, instructional OS have been developed, namely small operating systems designed specifically to teach rather than to function as an operating system [21]. Instructional OS is further divided into two types: those that are fully developed by students and those that are totally or partially developed so that students can analyze or modify them. Moreover, an instructional OS can run over real or simulated hardware. An instructional OS running on simulated hardware is easier for students to understand because it avoids the complex details of real hardware.

Programming Projects

The Programming Projects approach does not involve kernel-level programming. Instead, it is centered on smaller, user-level projects. The main principle underlying this approach is that the most important part of an OS course is the understanding of the concepts. Accordingly, all the studies using this approach agree that the main learning objective of the assignments is to help students grasp OS concepts, and not to teach them to program an OS.

Graphical Simulators

The main principle underlying the Graphical Simulators approach is that students never see the inside of an OS, which makes it difficult for them to understand its functioning [30]. The solution proposed by these studies is to provide a graphic representation that allows students to understand how an OS works by observing the events at each moment.

Learning Environments

The aim of the Learning Environments approach is to use software applications to teach and learn OS. Although there are four papers in the group, only three tools are described because two papers discuss the same tool. These tools are Exploratory Operating Systems (EOS), WebgeneOS, and Computer-Aided Learning Operating Systems (CALOS). The CALOS tool was so successful that it led to the development of the web tool WebCT, which is known today as Blackboard Learn, a Learning Management System widely used to teach not only OS but other disciplines as well.

Collaborative Learning

In the Collaborative Learning approach, studies use collaborative learning techniques to improve learning in OS. The term ''collaborative learning'' has been interpreted in several ways, but the common aspect is the emphasis on student interactions rather than on learning as a solitary activity [31].

Clickers

Studies employing this approach use clickers in order to improve learning. Clickers are individual response devices that allow students to answer questions in-class and summarize results immediately [32]. They are also called Classroom Response Systems or Personal Response Systems [9].

Games

The Games approach uses games as teaching and learning tools. This method implements educational games in order to motivate students and provides them opportunities to practice skills that the transmissions model classroom may not [33].

Conceptual Knowledge

Studies using this approach presented several methods to assess conceptual knowledge and/or discover alternative conceptions. Understanding conceptual knowledge is critical

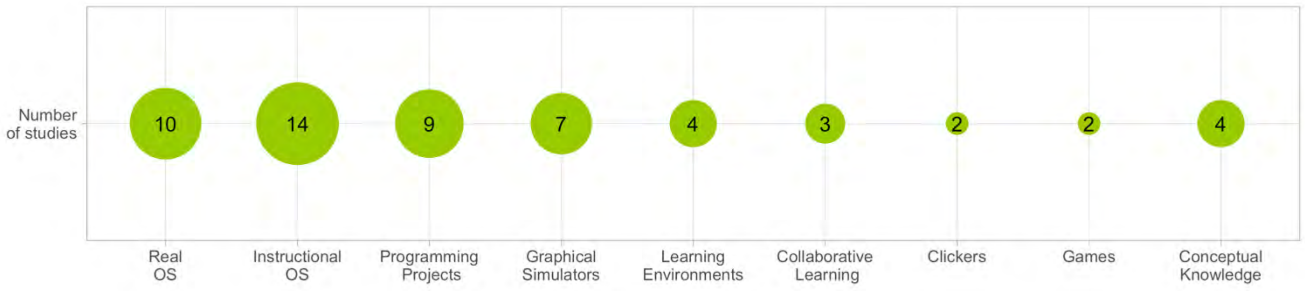

FIGURE 1. Categorization of studies based on teaching and learning approach.

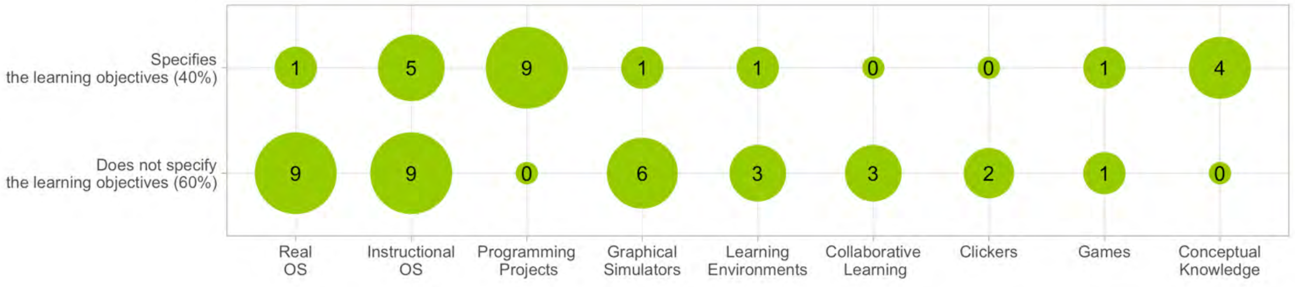

FIGURE 2. Number of articles specifying the learning objectives.

to develop the competence of engineering students and for practicing professionals [34]. We used the term "concept" broadly, including ideas, objects, or events that aid in understanding our environment [35]. An alternative conception is a conception that is clearly incompatible with the accepted conceptions, but is maintained persistently, even after instruction [36]. We prefer this general term "alternative conception" over more specific terms from the literature (e.g. misconceptions, misunderstandings, and mistakes).

The categorization of studies in terms of the selected approach to improve learning is indicated in Figure 1.

The results for the primary question revealed that 40 out of 55 studies (73%) employed traditional approaches. The studies using these approaches describe the different types of assignments that have been used in OS courses for a long time. The assignments described are specific to OS courses. These traditional approaches are Real OS (10 studies), Instructional OS (14 studies), Programming Projects (9 studies), and Graphical Simulators (7 studies).

The remaining studies, 15 out of 55 (27%), used approaches based on recent research trends in education. These trends are not specific to OS courses. Thus, they could be applied to different knowledge areas. Moreover, these trends can be combined with the different types of assignments described in the traditional approaches. These innovative approaches are Learning Environments (4 studies), Collaborative learning (3 studies), Clickers (2 studies), Games (2 studies), and Conceptual Knowledge (4 studies).

We have described the answers to the sub-questions below. We first present the answers in general. We then discussed the answers obtained corresponding to each approach.

Sub-Question 1. Does the study specify learning objectives?

Most of the studies (60%) did not explicitly specify learning objectives for the OS courses they implemented. Those which effectively did so (40%), as indicated below, consistently indicated the objective of the course as facilitating understanding of OS concepts. Figure 2 summarizes the number of articles specifying learning objectives according to each approach.

When the approaches were individually analyzed, we noticed the following outcomes. First, two approaches, namely Programming Projects and Conceptual Knowledge, were distinct because they differed from the general trend. Unlike in the other categories, in these two approaches, all the studies specified the learning objectives of the OS courses. In addition, all the studies consistently stated that their objective was to facilitate understanding of OS concepts.

Regarding the approaches using kernel programming projects (Real OS and Instructional OS), six studies (25%) advocated for learning OS concepts. The rest (18 studies – 75%) did not explicitly state learning objectives. Thus, we do not know whether the main objective of these approaches is to learn OS concepts or to learn how to program an OS.

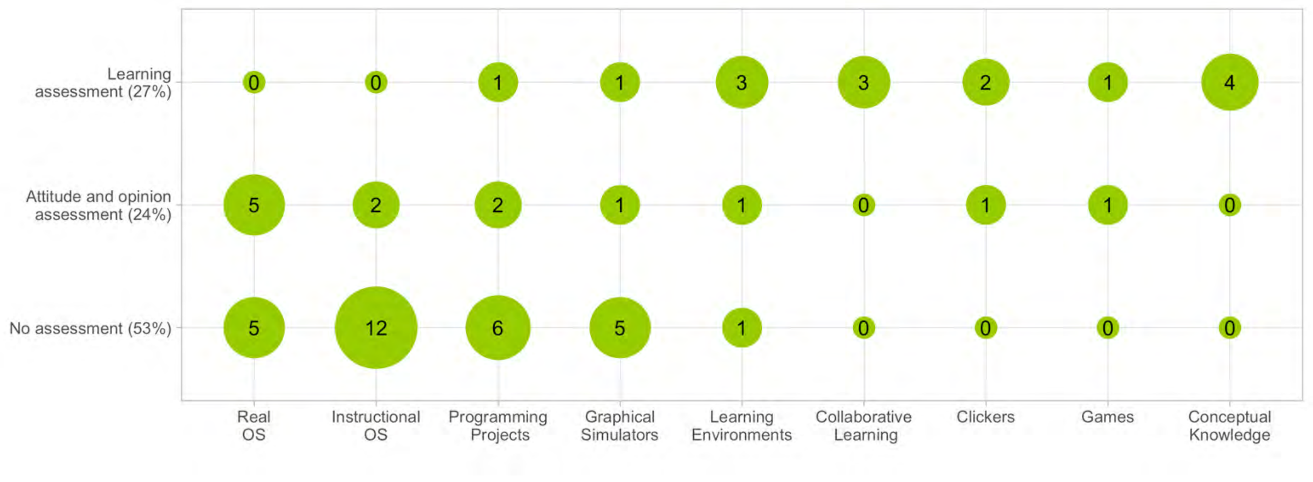

FIGURE 3. Type of assessment.

Among the papers employing the Graphical Simulators approach, only 1 out of 8 (12.5%) explicitly stated the learning objectives. In particular, the scheduling simulator created by [37] aimed to help the students understand process scheduling algorithms and how performance is influenced by system parameters. Although the remaining papers do not specify the learning objectives, it could be inferred from each paper that the main learning objective of the experiences described was understanding OS concepts.

In papers using the Learning Environment approach, only one paper explicitly stated the learning objectives explicit. In the study by [38], students had to learn how an OS works and what its main components were, how an OS interfaces between the computer and the user, and how an OS manages resources such as memory or storage devices. Therefore, the objective of this study is to understand OS concepts, and not to program an OS.

Among the papers employing the Collaborative Learning and Clicker approaches, none explicitly specified the learning objectives. Thus, we do not know the exact objective of approach. Surprinsingly, the study by Kvadsheim [9] suggested that in order to prepare for effective clicker-supported training, the learning objectives must be stated clearly. Therefore, we supposed that they established their learning objectives, but they have not made them explicit in the paper.

Finally, among the papers using the Games approach, only the study by [39] stated the course objectives. This study presented two different games. The first game, called *BattleThreads*, was designed to help students understand some important concepts about threads. The second game, called *Process State Transition*, aimed to facilitate understanding of process management, particularly the following concepts: transitions and data structures to manage processes and scheduling.

Summarizing, 40% of the analyzed studies confirmed that the objective of the proposed approach was to understand OS concepts. Moreover, authors of one study employing the Real OS approach [28] stated that they felt that the students benefited owing to improved programming skills after the approach was implemented, which can be considered as another learning objective for this approach. Therefore, we identified two different kinds of learning objectives: understanding concepts and improving programming skills.

Sub-questions 2, 3, and 4 are closely related. Therefore, the results for each approach are discussed together for these sub-questions.

Sub-Question 2. What kind of assessment has been used in the study to evaluate the approach (no assessment, attitude and opinion assessment, learning assessment)?

Among the different assessments, learning assessment is the most desirable option since it is the only one that can really verify the extent of effectiveness of the approach. Another noteworthy issue is that ''learning assessment'' and ''attitude and opinion assessment'' are not exclusive, since the same study may have carried out both kinds of assessment.

On reviewing the assessment used in each study (Figure 3), we found that 29 papers (53%) did not implement any kind of assessment, 13 (24%) used an attitude and opinion assessment, and 15 (27%) conducted a learning assessment.

The approaches on the right side of Figure 3 and those on the left are clearly distinguished. Most of the studies employing Real OS, Instructional OS, Programming Projects, and Graphical Simulators approaches did not perform any type of assessment. This implies that we do not have any information about their potential effectiveness. However, studies employing the five approaches on the right (Learning Environments, Collaborative Learning, Clickers, Games, and Conceptual Knowledge) conducted some kind of assessment. Therefore, these studies collected data that might allow us to assess the efficiency of the approaches.

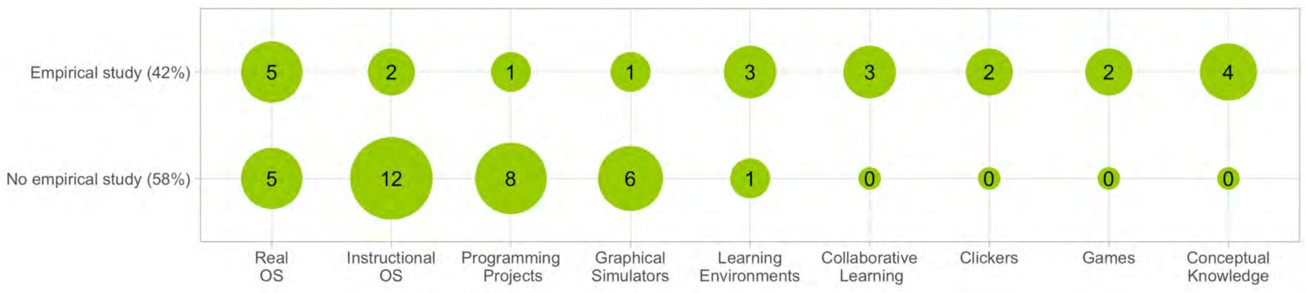

FIGURE 4. Number of articles describing an empirical study.

Sub-Question 3. Does the study describe any kind of empirical research?

We categorized the studies employing each approach into two exclusive groups based on the presence or absence of the description of an empirical study (Figure 4).

Out of the 53 analyzed papers, only 23 (42%) were empirical in nature. Figure 4 indicates that all the studies that employed Collaborative Learning, Clickers, Games, and Conceptual Knowledge approaches conducted an empirical study to test the effectiveness of the approach. This was also true for the Learning Environments category, which had four papers, but only three types of learning environments because two papers referred to the same type of learning environment. In the remaining categories, only a small percentage of the studies conducted empirical research (50% in Real OS, 17% in Instructional OS, 13% in Programming Projects, and 17% in Graphical Simulators).

Sub-Question 4. If an empirical study has been conducted, what research methodology was used?

As we explained in the methodology section we characterized the methodology used in each study using two dimensions: the purpose of the research and the type of gathered data. In particular, we considered three purposes (discovery, confirmatory, and descriptive) and two types of data (quantitative and qualitative). Therefore, there are six possible combinations of purpose and type of data. However, only four combinations are indicated in Figure 5 because the combination ‘‘confirmatory perspective and qualitative data’’ and the combination ‘‘discovery perspective and quantitative data’’ did not appear within the selected papers. This is not surprising, since the confirmatory perspective is usually associated with quantitative data and the discovery perspective with qualitative data [25].

Moreover, it is necessary to clarify that a study can have only a single purpose (descriptive, confirmatory, or discovery). However, a study can collect both types of data, quantitative and qualitative.

The methodology used by the studies that described empirical research is represented in Figure 5. Out of the 23 (52%) empirical studies, 12 applied a methodology with a descriptive purpose and quantitative data. Two of these papers (9%) collected qualitative data as well. A total of 8 studies (35%) used a methodology with a confirmatory purpose and quantitative data, and only 3 (13%) applied a methodology with a discovery purpose and qualitative data.

Figure 5 indicates that the studies using the Learning Environments, Collaborative Learning, Clickers, and Games approaches were the only ones to have a confirmatory purpose. On the other hand, only studies using the Conceptual Knowledge approach had a discovery purpose. Finally, all the studies using the Real OS, Instructional OS, Programming Projects, and Graphical Simulators approaches had a descriptive purpose.

We discuss Sub-Questions 2, 3, and 4 below for each approach.

No information was available about the effectiveness of the Real OS approach because none of the studies employing this approach conducted a learning assessment. Out of 10 papers (50%), 5 did not have any kind of assessment. The other 5 papers (50%) conducted a survey on attitudes at the end of the course and performed a quantitative analysis of the survey results. The results indicated that the use of the approach had positive results.

Similarly, no evidence was available about the effectiveness of the Instructional OS approach. The studies employing this approach appeared to focus on describing the Instructional OS and detailing some assignments. Hence, they focused on the technology rather than on the educational value of the provided solution. Only 2 out of 14 papers (14.29%) conducted an empirical study to evaluate the approach. Both studies evaluated the approach by asking students about their perceptions, and none of performed a real learning assessment.

Only two out of the eight studies employing the Programming Projects approach performed an assessment. In [13], an attitudes survey was conducted, and the results indicated that the students perceived that the approach supported the learning of the concepts. However, [17] argued that assessing the student comprehension of behavioral concepts is not easy in programming assignments. Programming assignments when successfully completed result in code which when executed will reproduce the behavior in question, but do not necessarily prove understanding of the behavior by the student. Instead of programming, in their assignments, they

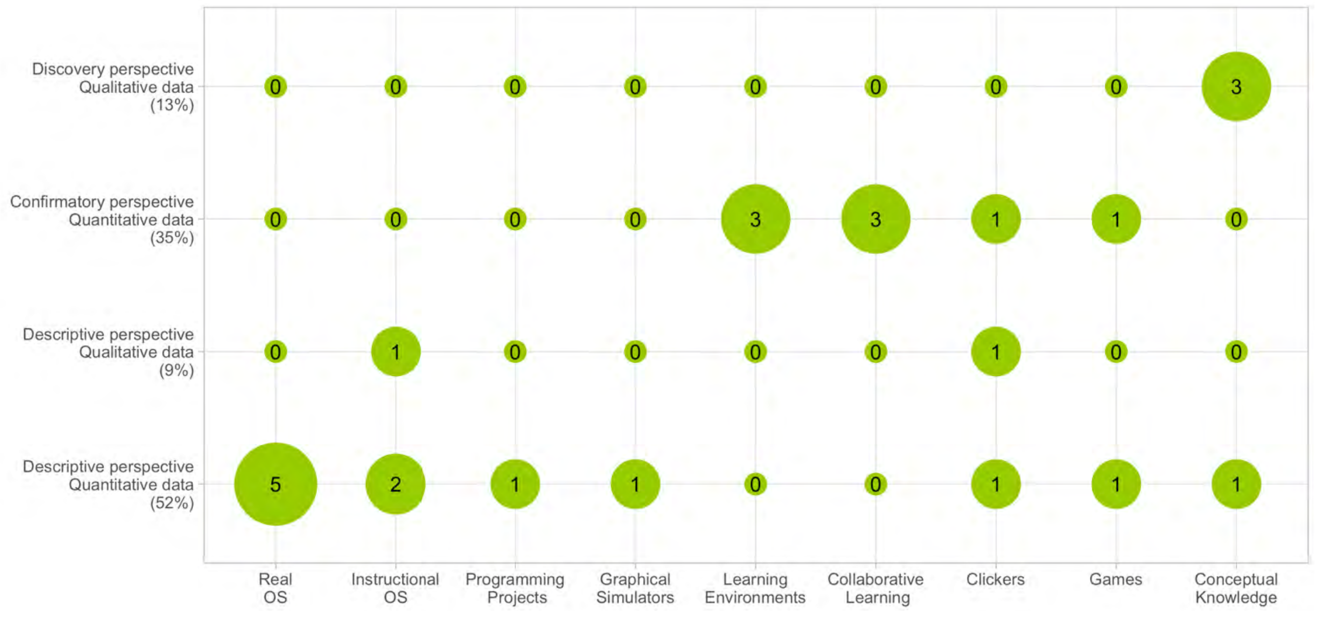

FIGURE 5. **Research methodology.**

asked for a description of the sequence of actions occurring in an OS after the introduction of a stimulus. Although this study described a learning assessment, no empirical research results were reported. In conclusion, as none of the studies performed an actual learning assessment, there is no evidence for the effectiveness of the Programming Projects approach.

Only two out of the seven papers (28.6%) employing the Graphical Simulators approach referred to any type of assessment. The simulator for practicing memory address translation [16] can be used for learning assessment by generating a log that will reveal how the student addressed the translations to the instructor. However, no empirical study was reported. Maia *et al.* [40] solicited feedback from the students to assess the benefits of the simulator as a valid educational tool. The results indicated that most students felt that learning with the simulator helped them understand OS concepts better. As no empirical study that performed a learning assessment had been conducted, no evidence was found for the effectiveness of the Graphical Simulators approach.

With regard to the studies using the Learning Environments approach, an empirical study of each of the applications (EOS, CALOS, and WebgeneOS) was conducted with positive results. All the studies had a confirmatory purpose; they intended to verify whether the use of a learning environment improved academic performance.

In the case of the tool EOS [41], a standardized testing instrument was used to compare class performance before and after adding the lab to the OS course. Preliminary results indicated an improvement in performance attributable to the EOS laboratory.

The evaluation of the tool CALOS [42] comprised a survey on attitudes and a learning assessment. In particular, the learning assessment was performed through four programming assignments, one midterm exam, and one final exam. The authors conducted a study by dividing the students into three groups: (1) students who had access to an online course and were not allowed to attend lectures, (2) students who both attended the lectures and had access to the online course as well, and (3) students who did not have access to the online course and attended lectures (traditional format). After comparing the academic performances of the three groups, the results indicated that those students who had access to both, lectures and the online course, performed better than either of the other groups. Students who had access only to the online course or the lectures performed roughly the same. Therefore, it appears that an online course could be as effective as a traditional lecture-based course.

The tool WebgeneOS [38] was evaluated using a learning assessment with five intermediate assignments and a final exam. The empirical study analyzed the impact of using the tool on students' grades for three academic years. The results indicated that students using WebgeneOS improved their grades by approximately 13% with respect to those attending face-to-face laboratory sessions.

The three papers employing the Collaborative Learning approach conducted empirical research, and their results appeared to support the use of the collaborative learning technique. The methodology used in each study is summarized below.

Panetta *et al.* [43] compared the class average on the final exam (83%) with the results of the previous two years (70% and 71%) and concluded that students had gained a stronger comprehension than the previous years. In addition, they performed a multiple regression analysis to examine the relationship between the software TEAMThink performance measures and student final exam scores. Statistically significant positive correlations were found between participation level and quiz scores and the student's final exam grade. Overall, a positive relationship appeared to exist between the use of the new methodology and the academic results of the students.

The study [44] compared the level of academic success reached and the achievement of certain generic competences (''teamwork'' and ''planning and time management'') between two groups in the same academic year: students who attended traditional lecture/discussion classes (control group) and students who attended classes using cooperative learning techniques (intervention group). Results indicated that collaborative learning methodologies improved the academic performance and had a positive effect on the teamwork competence, but did not produce any significant change in the generic competence ''planning and time management.''

The study [2] compared the demographics and outcomes of two courses in 2012 and 2013. The analysis indicated that students in the intervention group performed better on a variety of measurements even while considering the impact of certain demographic factors. They also stated that other factors that had not been considered may have played a role. Therefore, they did not assert a causal link between the used techniques and the improved outcomes, but they felt that there was a significant amount of anecdotal evidence to suggest that these techniques were beneficial and contributed to improving students' outcomes in the intervention group.

Both the studies employing the Clickers approach performed a learning assessment and an empirical research. In particular, the study [45] had both a descriptive and a confirmatory purpose. The evaluation was performed with 90 students: 45 students followed their proposal with clickers, and 45 students followed the same class structure but without using electronic polls. They considered four different sources of information: teachers' personal journals, participation reports, class surveys, and exam grades. The results indicated that the use of clickers in the considered scenario increased student participation, interest in attending classes, motivation, and performance. The study [9] performed a randomized experiment with a crossover design. The design of this research had a confirmatory purpose, and the findings supported the hypothesis that the use of clickers helped the students obtain better exam scores. In conclusion, these two studies provided empirical evidence supporting the idea that clickers improve learning.

All the considered studies that used the Games approach offered evidence of empirical research supporting the idea that games improve OS learning OS. The empirical study [39] had a descriptive purpose. The participating students were asked to respond anonymously to a survey about their experience with games. Most of the students agreed that the games were a better option than covering the same material in a lecture and that games helped them better understand OS concepts.

The study [12] performed a more complete and rigorous evaluation of the approach. This study used a quasi-experimental design with a confirmatory purpose. The participants' learning motivation and achievement were evaluated before and after the experiment, and an ANOVA was conducted to analyze the differences among groups. The results indicated that game-based cooperative learning excels in motivation and learning achievement.

All the studies employing the Conceptual Knowledge approach improved the learning of OS concepts. In particular, the studies [4], [46], [47] discovered or showed alternative conceptions and the study [48] developed a method to assess conceptual knowledge in an online environment. The methodology and results of each study using this approach are summarized below.

The study [46] had a discovery purpose and aimed at inferring mental models by observing students engaged in mental activities. In particular, the study analyzed the videotaped conversation between two students as they attempted to solve a problem in concurrency. The results revealed that both students had a flawed concept of a semaphore.

The study [47] had a discovery purpose and used an assessment design where students had to answer several questions about conceptual knowledge by submitting a written explanation. The analysis of this written data led to previously unknown alternative concepts about two OS concepts: interrupt and process switching.

The study [4] developed an Operating Systems Concept Inventory to explore students' alternative conceptions. The researchers created multiple choice questions with distractor answers based on their past experiences of teaching OS courses. The results offered a discussion of the lowest-performing questions. These questions provide alternative conceptions about the following concepts: indirection, Input/Output, and synchronization.

In [48], a formative assessment about conceptual knowledge pertaining to OS was designed for an online course. The assessment was based on Bloom's revised taxonomy and had two goals: to promote deep learning and to make students aware of their learning processes. The analysis of the students' answers to this assessment indicated that both goals had been achieved.

Sub-Question 5: What kind of teaching and learning mode has been used (face-to-face, online, blended)?

Most of the studies (91%) were conducted in face-to-face scenarios. Of the studies, 20% used an online context, and only 4% employed the blended teaching and learning mode. These three categories (Figure 6) are not exclusive because a study may have used more than one learning scenario.

The Instructional OS, Programming Projects, Graphical Simulators, Collaborative Learning, and Clickers approaches were used only in face-to-face scenarios.

Studies employing the following approaches used the online mode: Real OS (5 out of 10 studies, 50%), Learning Environments (3 out of 4 studies, 75%), Games

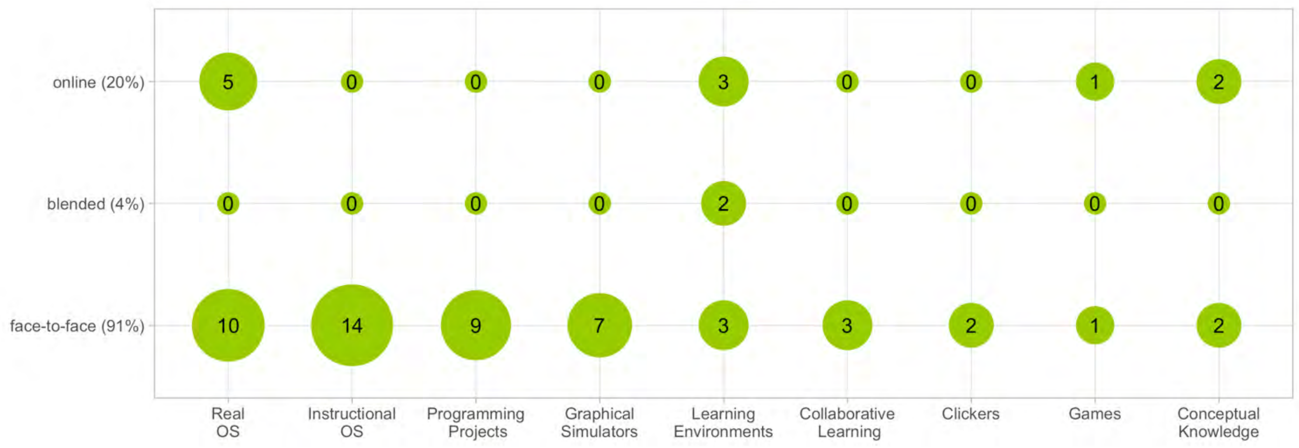

FIGURE 6. **Teaching and learning mode by approach.**

(1 out of 2 studies, 50%) and Conceptual Knowledge (2 out of 4 studies, 50%).

Blended learning was used only in studies employing the Learning Environment approach (2 out of 4 studies, 50%).

IV. DISCUSSION

This section discusses the results of the systematic mapping study. In addition, some implications for OS instructors and Engineering Education researchers are proposed.

The findings of our systematic mapping study have implications for OS instructors, since this study will allow them to identify the existing approaches in the literature on OS education. Moreover, the empirical studies presented can provide an overview of the efficiency of each approach. In addition, recommendations for future research based on the review outcomes are provided.

A. IMPLICATIONS FOR OS INSTRUCTORS

The implications of our study for OS instructors are as follows:

- Nine different approaches have been used to improve the processes of OS teaching and learning from 1995 to 2017. Four of them are traditional approaches that have been used for a long time in OS courses. The first of these traditional approaches, Real OS, involves programming the kernel of a real OS. This approach is very complex, and most students considered it too difficult. This led to the development of another approach: Instructional OS. This approach continues to advocate kernel programming but uses an OS created for educational purposes instead of a real OS. However, this approach is considered to be too complex by some teachers and students. As a consequence, and because the essential objective of an OS course is the understanding of OS concepts, a new approach called the Programming Projects approach emerged, which consists of programming or using different pieces of software to illustrate OS concepts. The fourth traditional approach, Graphical Simulators, is not related to programming. Instead, it focuses on representing the events of an OS in a graphical manner to facilitate the understanding of the main OS concepts. The remaining approaches are based on education research trends (Learning Environments, Collaborative Learning, Clickers, Games, and Conceptual Knowledge). The aforementioned traditional approaches can be combined with these innovative ones to meet the requirements of teachers.
- In terms of learning objectives, 40% of the analyzed studies indicated the objective of the proposed approach to be understanding OS concepts. These objectives are consistent with those established by ACM IEEE for OS, according to the analysis carried out in [48]. Therefore, it might be inferred that the understanding of concepts is also the objective of the remaining 60% of the studies, although this has not been mentioned explicitly.
- In addition, we have identified two kinds of learning objectives for OS in our study: understanding concepts and improving OS programming skills. These two kinds of objectives could be related to two different professional profiles and two different OS courses. The understanding of concepts might be adequate for students who need to be aware of both the issues raised in an OS and techniques for solving problems in these areas, but they are unlikely to ever write code for an OS kernel. The improvement of programming skills might be appropriate for students who will write or maintain OS kernels [41].
- We identified a debate around the approaches based on kernel programming projects. Some authors consider that kernel programming projects (Real OS and Instructional OS) are not particularly efficient means of teaching OS principles and concepts [41]. This is because most of the time spent on writing an OS kernel is not time spent on learning OS concepts, but struggling with

TABLE 2. Real OS.

Authors and year	Aim	Specifies learning objectives	Approach evaluation	Empirical study	Research methodology	Teaching and learning mode
Pérez-Dávila, 1995 [53]	To present a different approach of teaching operating systems: implementing a laboratory that allows students to experiment with a copy of the 386BSD Unix System.	No	No	No	n/a	face-to-face
Nieh & Vaill, 2005 [54]	To present a tool for developing operating systems: a virtual kernel development environment suitable for face-to-face and distance settings.	No	No	No	n/a	face-to-face and online
Lawson & Barnett, 2008 [28]	To present a project in which the students modify a real operating system kernel: the iPodLinux, a version of the Linux operating system that runs on iPods.	No	No	No	n/a	face-to-face
Hwang, 2009 [55]	To propose a course project based on Windows kernel.	No	Attitude and opinion assessment	Yes	Descriptive perspective Quantitative data	face-to-face
Hess & Paulson, 2010 [29]	To present a series of programming projects based on the Linux kernel.	No	Attitude and opinion assessment	Yes	Descriptive perspective Quantitative data	face-to-face
Laadan, Nieh, & Viennot, 2010 [27]	To describe a new approach to teaching operating systems based on virtual appliances, a distributed version control system, and live demonstrations.	No	No	No	n/a	face-to-face and online
Schmidt, Polze, & Probert, 2010 [56]	To present the Abstract Memory Management Project where students need to modify the Windows kernel, compile and run their own version of Windows.	Yes	Attitude and opinion assessment	Yes	Descriptive perspective Quantitative data	face-to-face
Laadan, Nieh & Viennot, 2011 [57]	To present a series of Linux kernel programming projects focused on five core topics: system calls and processes, synchronization mechanisms, processor scheduling, virtual memory and file systems.	No	No	No	n/a	face-to-face and online
Andrus & Nieh, 2012 [58]	To present a series of Android kernel programming projects focused on five core topics: system calls and processes, synchronization mechanisms, processor scheduling, virtual memory and file systems.	No	Attitude and opinion assessment	Yes	Descriptive perspective Quantitative data	face-to-face and online
Dall & Nieh, 2014 [59]	To present GradeBoard, a web-based code review system designed to simplify grading and help students to understand and learn from their errors.	No	Attitude and opinion assessment	Yes	Descriptive perspective Quantitative data	face-to-face and online

low-level programming. On the contrary, other authors [8] argue that having hands-on source code experience is a minimal requirement for appropriate understanding of OS concepts. In [49], it is even stated that students who have completed their kernel OS projects have no doubts at all about how an OS works by the end of the course.

- It was found that the Real OS and Instructional OS approaches can be used to improve both OS programming skills and the learning of OS concepts. The remaining approaches appeared to be more focused on facilitating the understanding of OS concepts.
- No evidence was found for the effectiveness of the Real OS, Instructional OS, Programming Projects, and Graphical Simulators approaches because the studies that advocated these approaches did not conduct any empirical research with a learning assessment.

TABLE 3. Instructional OS.

Authors and year	Aim	Specifies learning objectives	Approach evaluation	Empirical study	Research methodology	Teaching and learning mode
Morsiani & Davoli, 1999 [60]	To present an experimental project on operating system development (TINA) and a hardware emulator (MPS).	No	No	No	n/a	face-to-face
Dickinson, 2000 [49]	To describe a simple hardware simulator that allows students to create their own operating system in stages and discover the fundamental issues involved with operating systems.	Yes	No	No	n/a	face-to-face
Donaldson, 2001 [61]	To describe a project for a course on operating systems consisting on building a rudimentary operating system kernel from the bottom up based on the Intel architecture.	Yes	No	No	n/a	face-to-face
Nicholas & Barchanski, 2001 [62]	To describe an educational distributed operating system implemented in Java that is sufficiently simple to cover the core concepts of operating systems and includes many of the key features of distributed.	No	No	No	n/a	face-to-face
Holland, Lim, & Seltzer, 2002 [63]	To present an instructional operating system called OS/161 and a simulated execution environment called OS/161 to teach an undergraduated operating system course.	No	No	No	n/a	face-to-face
Atkin & Sirer, 2002 [64]	To describe PortOS, an educational operating system that focuses on peer-to-peer distributed computing on mobile devices.	No	No	No	n/a	face-to-face
Hovemeyer, Hollingsworth, & Bhattacharjee, 2004 [65]	To present GeekOS, an instructional operating system kernel which runs on real hardware.	Yes	No	No	n/a	face-to-face
Liu, Chen, & Gong, 2007 [66]	To present BabyOS, a very small educational operating system which reflect some interesting design concepts of embedded operating systems.	No	No	No	n/a	face-to-face
Cheng & Lin, 2008 [67]	To propose an instructional operating system that not depends on any hardware simulator or platform. It provides basic hardware functions such as timer interrupts and page-fault interrupts that are simulated using the Unix utility awk as a program instrumentation tool.	No	Attitude and opinion assessment	Yes	Descriptive perspective Quantitative and qualitative data	face-to-face.
Black, 2009 [68]	To describe a semester project where students design a small and simple operating system consisting of system calls, program execution, a file system, a command-line shell and support for multiprocessing.	No	No	No	n/a	face-to-face
Pfaff, Romano, & Back, 2009 [69]	To present Pintos, an instructional operating system to teach the principles of multi-programming, scheduling, virtual memory and filesystems.	No	No	No	n/a	face-to-face
Corliss & Melara, 2011 [70]	To present the VIREOS educational operating system which runs on the Lard educational architecture.	Yes	Attitude and opinion assessment	Yes	Descriptive perspective Quantitative data	face-to-face
Goldweber, Davoli, & Jonjic, 2012 [71]	To present µMPS2, the first multiprocessor system emulator designed specifically for an undergraduate operating systems course.	No	No	No	n/a	face-to-face
Román Otero & Aravind, 2015 [8]	To present MiniOS, a minimal instructional embedded operating system designed to create an operating system from scratch.	Yes	No	No	No	face-to-face

TABLE 4. Programming projects.

Authors and year	Aim	Specifies learning objectives	Approach evaluation	Empirical study	Research methodology	Teaching and learning mode
Wagner & Ressler, 1997 [11]	To describe an approach to teach Operating Systems based on straightforward programming problems rather than problems that involve kernel-level programming projects.	Yes	No	No	n/a	face-to-face
Holliday, 1997 [72]	To describe a sequence of material based on two concepts (the system call interface and the interrupt vector mechanism) that can be used in an operating system course.	Yes	No	No	n/a	face-to-face
Ziegler, 1999 [13]	To present some practical experiences that employ short programs with unexpected behaviors.	Yes	Attitude and opinion assessment	Yes	Descriptive perspective Quantitative data	face-to-face
Downey, 1999 [73]	To describe an approach for teaching operating systems where students are asked to conduct a series of experiments about operating system concepts and make inferences from the results.	Yes	No	No	n/a	face-to-face
Polze & Probert, 2006 [74]	To present an approach to teach Operating System concepts based on the Windows family of operating systems.	Yes	No	No	N/a	face-to-face
Sheehan, 2007 [75]	To expose the advantages and disadvantages of using Ruby as the language for assignment work and the presentation of concepts in an Operating Systems course.	Yes	No	No	n/a	face-to-face
Donaldson, 2008 [76]	To describe a project for an undergraduate operating system course that provides an alternative approach to projects based on kernel programming.	Yes	No	No	n/a	face-to-face
Desnoyers, 2011 [17]	To present a behavioral approach to teach Operating Systems. The goal is to teach how an operating system works, that is to understand the causal chains of events which occur in response to stimuli such as user requests.	Yes	Learning assessment	No	n/a	face-to-face
O'Brien, 2017 [3]	To describe an approach where the Linux monitoring tool SystemTap is used to capture kernel-level events in order to illustrate, with concrete examples, operating system concepts in the areas of scheduling, file system implementation and memory management.	Yes	Attitude and opinion assessment	No	n/a	face-to-face

- There is some evidence of the effectiveness of the Learning Environments, Collaborative Learning, Clickers, Games, and Conceptual Knowledge approaches. These have been tested through the application of learning assessments and rigorous methodologies

B. IMPLICATIONS FOR RESEARCH

- Most of the analyzed papers do not specify the learning objectives of the described OS courses. Making the learning objectives explicit is important due to the following reasons. The first is that we cannot always infer the learning objectives from an assignment. In the particular case of OS education, programming assignments are common. Unfortunately, it is difficult to know whether the main purpose of such assignments is to improve the students' programming skills or to impart an understanding of OS concepts to them. Secondly, if the objectives are not established, it cannot be verified whether they have been achieved. In addition, each approach might be appropriate for facilitation a certain

TABLE 5. Graphical simulators.

Authors and year	Aim	Specifies learning objectives	Approach evaluation	Empirical study	Research methodology	Teaching and learning mode
Robbins & Robbins, 1999 [37]	To present a process scheduling simulator written in Java that has two objectives: to help students understand operating system concepts and to incorporate an experimental approach into the course.	Yes	No	No	n/a	face-to-face
Robbins, 2000 [15]	To present a simulator for experimenting with a classical synchronization problem: the bounded buffer problem.	No	No	No	n/a	face-to-face
Robbins, 2002 [77]	To examine the use of a simulator for exploring process interaction in UNIX by showing how the processes and pipes are connected.	No	No	No	n/a	face-to-face
Robbins, 2004 [78]	To describe a disk head scheduling simulator for exploring traditional disk scheduling algorithms.	No	No	No	n/a	face-to-face
Robbins, 2005 [16]	To discuss a simulator for practising address translation in a traslation lookaside buffer (TLB) and single or 2-level page tables.	No	Learning assessment	No	n/a	face-to-face
Maia, Machado, & Pacheco Jr., 2005 [40]	To propose a pedagogical approach for teaching and learning operating systems based on constructivism. The framework presented uses a graphical simulator called SOsim where students can visualize how an operating system works.	No	Attitude and opinion assessment	Yes	Descriptive perspective Quantitative data	face-to-face
Robbins, 2006 [79]	To describe a simulator that allows students to explore the mechanism of concurrent I/O in UNIX.	No	No	No	n/a	face-to-face

type of objective, and both teachers and researchers need to know the learning objectives are. Finally, without learning objectives, instructors cannot know whether they can be applied to the approach to facilitate the learning objectives of their own OS courses. In conclusion, learning objectives should be stated clearly for the learning experiences described in engineering education research.

- Only 27% of the studies performed a learning assessment. Hence, we do not know whether the approaches used in the remaining studies (73%) were effective. Thus, in order to assess the reliability of a teaching and learning approach, it is desirable to design and apply an appropriate learning assessment.
- Less than half of the studies (42%) conducted empirical research. Consequently, more empirical studies are needed to test the efficacy of the approaches used for OS teaching and learning. In particular, the percentage of empirical studies (23%) on traditional learning approaches (Real OS, Instructional OS, Programming Projects, and Graphical Simulators) is even lower. Therefore, empirical studies are particularly necessary in these categories.
- We observed a lack of rigor in the methodologies applied by the studies employing the Real OS, Instructional OS, Programming Projects, and Graphical simulators approaches. All the methodologies used in these approaches had a descriptive purpose. None of the studies had a confirmatory or a discovery purpose.
- A variety of reasons account for the lack of learning assessments and rigorous methodologies in the aforementioned approaches. One of them is that authors of the studies might take the effectiveness of the traditional approaches for granted in contrast with the new approaches that require evidence for their adoption. Another reason might be the background of the researchers. The authors' list and affiliations for the considered studies suggest that research had been carried out by engineers who teach OS, and not by researchers of engineering education. This issue has already been noted by previous reviews on engineering education [33], [50].

TABLE 6. **Learning environments.**

Authors and year	Aim	Specifies learning objectives	Approach evaluation	Empirical study	Research methodology	Teaching and learning mode
Erickson, 1996 [41]	To describe a laboratory environment for teaching operating systems that is based on three approaches: individual programming projects, simulators and writing portions of a kernel.	No	Learning assessment	Yes	Confirmatory perspective Quantitative data	face-to-face
Goldberg, 1996 [80]	To describe CALOS, a web application to learn Operating Systems. The course consists of interactive exercises, interactive simulations, instructor and student communications mechanisms, students evaluations, a glossary and a bibliography.	No	No	No	n/a	face-to-face, blended, and online
Goldberg, 1997 [42]	To evaluate CALOS, a www-based course delivery both in terms of academic performance and students acceptance.	No	Attitude and opinion assessment and learning assessment	Yes	Confirmatory perspective Quantitative data	face-to-face, blended, and online
Buendía & Cano, 2006 [38]	To present and to evaluate WebgeneOS, a experimental learning system based on web technologies and generative learning methods.	Yes	Learning assessment	Yes	Confirmatory perspective Quantitative data	online

TABLE 7. **Collaborative learning.**

Authors and year	Aim	Specifies learning objectives	Approach evaluation	Empirical study	Research methodology	Teaching and learning mode
Panetta, Dornbusch, & Loomis, 2002 [43]	To discuss a collaborative learning methodology to develop a deeper understanding of the material in an operating systems course.	No	Learning assessment	Yes	Confirmatory perspective Quantitative data	face-to-face
Pérez, García, Muñoz, & Sierra, 2010 [44]	To present the experience of applying two active learning methodologies (Cooperative Learning and Problem Based Learning) to an Operating System Course of 159 students.	No	Learning assessment	Yes	Confirmatory perspective Quantitative data	face-to-face
Kirkpatrick & Prins, 2015 [2]	To describe two techniques to increase the amount of active learning in an Operating Systems course.	No	Learning assessment	Yes	Confirmatory perspective Quantitative data	face-to-face

- Approximately a third of the studies (35%) used a confirmatory methodology, and only a few studies (13%) applied a methodology with a discovery purpose. A study with a confirmatory purpose usually aims to confirm that a new approach is better than the approach in use. In such studies, an experimental group of students is treated with a new approach of teaching and/or learning, and the performance of this group is compared to that of a control group that is not treated with the new approach. A statistically significant improvement in the performance of the experimental group is taken as evidence for the new approach being better [46]. In sum, a confirmatory study provides information about which approach is better, but it does not explain why and how. In order to discover the answers to these questions, studies with a discovery purpose are needed. Such studies use methods that presumably enable the investigator to ask questions and discover answers that are based on the events under study than the investigator's preconception [25]. Consequently, more studies with a discovery purpose are needed. Such studies are naturally associated with the analysis of qualitative data [51].
- Only 20% of the analyzed studies were conducted in an online scenario, and the percentage is even lower for the studies tested in a blended context. Provided that the

TABLE 8. **Clickers.**

Authors and year	Aim	Specifies learning objectives	Approach evaluation	Empirical study	Research methodology	Teaching and learning mode
Such, Criado, & García-Fornes, 2015 [45]	To describe an active learning technique based on electronic polls that increase participation, interest in attending classes and learning outcomes achievement.	No	Attitude and opinion assessment and learning assessment	Yes	Descriptive perspective Quantitative and qualitative data	face-to-face
Kvadsheim, Haugerud, Hammer, Bratterud, & Habib, 2015 [9]	To report a study on the effects of clicker use on student learning carried out within an undergraduate course in Operating Systems	No	Learning assessment	Yes	Confirmatory perspective Quantitative data	face-to-face

TABLE 9. **Games.**

Authors and year	Aim	Specifies learning objectives	Approach evaluation	Empirical study	Research methodology	Teaching and learning mode
Hill, Ray, Blair, & Carver Jr., 2003 [39]	To assess the effectiveness of two games used in an undergraduate operating system course. The goal of the games is to help students understand process and thread management	Yes	Attitude and opinion assessment	Yes	Descriptive perspective Quantitative data	face-to-face
Jong, Lai, Hsia, Lin, & Lu, 2013 [12]	To investigate the differences in motivation and achievement between game-based cooperative learning and traditional face-to-face cooperative learning. For this study, an online game was developed for students in the Operating systems course.	No	Attitude and opinion assessment and learning assessment	Yes	Confirmatory perspective Quantitative data	online

number of online and blended courses is increasing [52], research needs to pay more attention to the online and blended learning modes.

C. LIMITATIONS

Several factors need to be considered about the results of this study:

- Identification of primary studies. We carefully defined a search strategy that balanced sensitiveness and precision. In addition, we informed criteria for deciding which studies were to be included in the review. Nevertheless, alternative terms might have altered the final list of papers. Moreover, the search was performed by using six publications. Consequently, the papers that were not published in these forums were not considered in the study.
- Data extraction and analysis. The first two authors carried out both the data extraction and the classification of the primary studies. The results were then discussed with the third author. When a disagreement arose, a discussion was conducted until an eventual agreement was reached.
- Conclusion validity. The process followed to perform the selection of the papers and the data extraction and analysis were clearly described in the previous sections. The traceability between the extracted data and the obtained conclusions was strengthened through the generation of bubble plots based on the data. The conclusions drawn were valid for the 55 papers that were selected and analyzed. Nevertheless, the search string and the classification scheme presented may serve as a starting point for researchers in engineering education to identify and categorize additional papers accordingly.

V. CONCLUSIONS

This paper presented a systematic mapping study that summarizes the main research on OS education. A total of 55 studies published between 1995 and 2017 were selected and analyzed according to five criteria: learning objectives,

TABLE 10. Conceptual knowledge.

Authors and year	Aim	Specifies learning objectives	Approach evaluation	Empirical study	Research methodology	Teaching and learning mode
Kolikant, Ben-Ari, & Pollack, 2000 [46]	To obtain cognitive information of the process of understanding concurrency using qualitative methods.	Yes	Learning assessment	Yes	Discovery perspective Qualitative data	face-to-face
Pamplona, Medinilla & Flores, 2013 [47]	To discovery students' misconceptions in an online operating systems course.	Yes	Learning assessment	Yes	Discovery perspective Qualitative data	online
Webb & Taylor, 2014 [4]	To describe the development of an Operating Systems concept inventory in order to explore students' misconceptions.	Yes	Learning assessment	Yes	Descriptive perspective Quantitative data	face-to-face
Pamplona, Medinilla, & Flores, 2015 [48]	To explore the effects of a formative assessment on operating systems at an online university.	Yes	Learning assessment	Yes	Discovery perspective Qualitative data	online

approach evaluation, existence of an empirical study, research methodology applied, and mode of the course (online, blended, face-to-face). We have discussed our results and presented the implications for instructors and for research in this field.

Regarding the implications for instruction, the results of the present study may enable OS teachers to identify different ways to improve the effectiveness of their courses. Nine approaches to improve OS teaching and learning were identified. Four of them are traditional approaches that have been used for a long time in OS courses: Real Operating Systems, Instructional Operating Systems, Programming Projects, and Graphical Simulators. The other five are based on new trends on educational research: Learning Environments, Collaborative Learning, Clicker Games, and Conceptual Knowledge. The traditional approaches can be combined with the innovative ones to meet the requirements of the teachers.

Furthermore, two different kinds of learning objectives were identified: understanding concepts and improving OS programming skills. The improvement of programming skills might be appropriate for students who will write or maintain OS kernels, whereas the understanding of OS concepts might be enough for students who are unlikely to write code for an OS kernel.

It appears that both Real OS and Instructional OS approaches can be used to improve students' programming skills, whereas the remaining identified approaches are more focused on facilitating the understanding of OS concepts.

Moreover, a debate around the approaches based on kernel programming projects was detected. It is not clear whether low-level programming is an efficient means of teaching OS principles and concepts.

Regarding the implications for research, there are several shortcomings in the analyzed studies. First, most of the studies do not specify the learning objectives. Consequently, teachers cannot detect quickly whether the described learning experiences fit their needs. Second, few studies carried out a learning assessment and applied a rigorous methodology. Thus, there is little evidence to assess their effectiveness. Therefore, in order to strengthen future research on OS education, we recommend clearly stating the learning objectives, in order to design and conduct a proper learning assessment and to use a rigorous methodology.

Moreover, most studies that use rigorous methodologies have a confirmatory purpose. Consequently, they help to identify the more suitable approach, but do not explain why or how. In order to shed light on these questions, more studies employing a qualitative methodology are needed.

Finally, most of the studies were carried out in face-to-face scenarios. Therefore, research needs to pay more attention to emerging scenarios such as online and blended learning.

In conclusion, with this systematic map, we document both the research done so far and the current gaps in order to improve OS teaching and learning processes.

REFERENCES

[1] *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*, ACM/IEEE, New York, NY, USA, 2013.

[2] M. S. Kirkpatrick and S. Prins, "Using the readiness assurance process and metacognition in an operating systems course," in *Proc. ACM Conf. Innov. Technol. Comput. Sci. Educ. (ITiCSE)*, 2015, pp. 183–188.

[3] D. O'Brien, "Teaching operating systems concepts with SystemTap," in *Proc. ACM Conf. Innov. Technol. Comput. Sci. Educ.*, 2017, pp. 335–340.

[4] K. C. Webb and C. Taylor, "Developing a pre- and post-course concept inventory to gauge operating systems learning," in *Proc. 45th ACM Tech. Symp. Comput. Sci. Educ.*, 2014, pp. 103–108.

[5] C. Yang, "Computer operating systems in electrical engineering curriculum," *IEEE Trans. Educ.*, vol. 36, no. 1, pp. 177–180, Feb. 1993.

[6] S. Pamplona, I. Seoane, J. Bravo-Agapito, and N. Medinilla, "Insights into students' conceptual understanding of operating systems: A four-year case study in online education," *IEEE Commun. Mag.*, vol. 55, no. 11, pp. 170–177, Nov. 2017.

[7] *Computer Science Curriculum 2008: An Interim Revision of CS 2001, Report from the Interim Review Task Force*, ACM/IEEE, New York, NY, USA, 2008.

[8] R. R. Otero and A. A. Aravind, "MiniOS: An instructional platform for teaching operating systems project," in *Proc. 46th ACM Tech. Symp. Comput. Sci. Educ. (SIGCSE)*, 2015, pp. 430–435.

[9] R. Kvadsheim, H. Haugerud, H. L. Hammer, A. Bratterud, and L. Habib, "Does clicker use improve exam scores? A controlled randomized experiment in a bachelor-level course in software engineering," *Int. J. Eng. Educ.*, vol. 31, no. 2, pp. 505–520, 2015.

[10] A. S. Tanenbaum, "A UNIX clone with source code for operating systems courses," *ACM SIGOPS Oper. Syst. Rev.*, vol. 21, no. 1, pp. 20–29, 1987.

[11] T. D. Wagner and E. K. Ressler, "A practical approach to reinforcing concepts in introductory operating systems," in *Proc. 28th SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 1997, vol. 29, no. 1, pp. 44–47.

[12] B. S. Jong, C. H. Lai, Y. T. Hsia, T. W. Lin, and C. Y. Lu, "Using game-based cooperative learning to improve learning motivation: A study of online game use in an operating systems course," *IEEE Trans. Educ.*, vol. 56, no. 2, pp. 183–190, May 2013.

[13] U. Ziegler, "Discovery learning in introductory operating system courses," in *Proc. 13th SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 1999, pp. 321–325.

[14] S. J. Lincke, "Creating interest in operating systems via active learning," in *Proc. Frontiers Edu. 35th Annu. Conf.*, 2005, pp. S3C-7–S3C-10.

[15] S. Robbins, "Experimentation with bounded buffer synchronization," in *Proc. 31st SIGCSE Tech. Symp. Comput. Sci. Educ.*, 2000, pp. 330–334.

[16] S. Robbins, "An address translation simulator," in *Proc. 36th SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 2005, vol. 37, no. 1, pp. 515–519.

[17] P. J. Desnoyers, "Teaching operating systems as how computers work," in *Proc. 42nd ACM Tech. Symp. Comput. Sci. Educ.*, 2011, pp. 281–286.

[18] D. Gough, S. Oliver, and J. Thomas, *An Introduction to Systematic Reviews*. London, U.K.: SAGE, 2012.

[19] M. Borrego, M. J. Foster, and J. E. Froyd, "What is the state of theArt of systematic reviewin engineering education?" *J. Eng. Educ.*, vol. 104, no. 2, pp. 212–242, 2015.

[20] M. Borrego, M. J. Foster, and J. E. Froyd, "Systematic literature reviews in engineering education and other developing interdisciplinary fields," *J. Eng. Educ.*, vol. 103, no. 1, pp. 45–76, Jan. 2014.

[21] C. L. Anderson and M. Nguyen, "A survey of contemporary instructional operating systems for use in undergraduate courses," *J. Comput. Sci. Colleges*, vol. 21, no. 1, pp. 183–190, Oct. 2005.

[22] *CORE Conference Portal*, Comput. Res. Educ., Washington, DC, USA, 2013.

[23] *Journal Citation Reports*, Thomson Reuters, Toronto, ON, Canada, 2016.

[24] L. W. Anderson *et al.*, *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives, Abridged Edition*. Boston, MA, USA: Allyn & Bacon, 2000.

[25] B. J. Biddle and D. S. Anderson, "Theory, methods, knowledge and research on teaching," in *Handbook of Research on Teaching: A Project of the American Educational Research Association*, M. C. Wittrock, Ed. New York, NY, USA: Macmillan, 1986, pp. 230–252.

[26] J. Watson, A. Murin, L. Vashaw, B. Gemin, and C. Rapp, "Keeping pace with K-12 online learning: An annual review of policy and practice, 2011," Evergreen Educ. Group, Durango, CO, USA, Tech. Rep., 2011.

[27] O. Laadan, J. Nieh, and N. Viennot, "Teaching operating systems using virtual appliances and distributed version control," in *Proc. 41st ACM Tech. Symp. Comput. Sci. Educ.*, 2010, pp. 480–484.

[28] B. Lawson and L. Barnett, "Using iPodLinux in an introductory OS course," in *Proc. 39th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 2008, pp. 182–186.

[29] R. Hess and P. Paulson, "Linux kernel projects for an undergraduate operating systems course," in *Proc. 41st ACM Tech. Symp. Comput. Sci. Educ.*, 2010, pp. 485–489.

[30] D. A. Cañas, "Graphos: A graphic operating system," in *Proc. 18th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 1987, pp. 201–205.

[31] M. Prince, "Does active learning work? A review of the research," *J. Eng. Educ.*, vol. 93, no. 3, pp. 223–231, 2004.

[32] M. E. Lantz, "The use of 'clickers' in the classroom: Teaching innovation or merely an amusing novelty?" *Comput. Hum. Behav.*, vol. 26, no. 4, pp. 556–561, Jul. 2010.

[33] C. A. Bodnar, D. Anastasio, J. A. Enszer, and D. D. Burkey, "Engineers at play: Games as teaching tools for undergraduate engineering students," *J. Eng. Educ.*, vol. 105, no. 1, pp. 147–200, Jan. 2016.

[34] R. A. Streveler, T. A. Litzinger, R. L. Miller, and P. S. Steif, "Learning conceptual knowledge in the engineering sciences: Overview and future research directions," *J. Eng. Educ.*, vol. 97, no. 3, pp. 279–294, Jul. 2008.

[35] P. Eggen and D. Kauchak, *Educational Psychology: Windows on Classrooms*, 9th ed. London, U.K.: Pearson, 2012.

[36] I. O. Abimbola, "The problem of terminology in the study of student conceptions in science," *Sci. Educ.*, vol. 72, no. 2, pp. 175–184, Apr. 1988.

[37] S. Robbins and K. A. Robbins, "Empirical exploration in undergraduate operating systems," in *Proc. 13th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 1999, pp. 311–315.

[38] F. Buenda and J.-C. Cano, "Webgene$_{OS}$: A generative and Web-based learning architecture to teach operating systems in undergraduate courses," *IEEE Trans. Educ.*, vol. 49, no. 4, pp. 464–473, Nov. 2006.

[39] J. M. D. Hill, C. K. Ray, J. R. S. Blair, and C. A. Carver, Jr, "Puzzles and games: Addressing different learning styles in teaching operating systems concepts," in *Proc. 34th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 2003, pp. 182–186.

[40] L. P. Maia, F. B. Machado, and A. C. Pacheco, Jr., "A constructivist framework for operating systems education: A pedagogic proposal using the SOsim," in *Proc. 10th Annu. SIGCSE Conf. Innov. Technol. Comput. Sci. Educ.*, 2005, pp. 218–222.

[41] C. Erickson, "The EOS laboratory environment for a course in operating systems," in *Proc. 27th SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 1996, pp. 353–357.

[42] M. W. Goldberg, "CALOS: First results from an experiment in computer-aided learning for operating systems," in *Proc. 28th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 1997, pp. 48–52.

[43] K. Panetta, C. Dornbush, and C. Loomis, "A collaborative learning methodology for enhanced comprehension using TEAMThink," *J. Eng. Educ.*, vol. 91, no. 2, pp. 223–229, Apr. 2002.

[44] J. E. P. MartÃnez, J. M. García, I. F. Muñoz, and A. A. Sierra, "Active learning and generic competences in an operating systems course," *Int. J. Eng. Educ.*, vol. 26, no. 6, pp. 1484–1492, Dec. 2010.

[45] J. M. Such, N. Criado, and A. García-Fornes, "An active learning technique enhanced with electronic polls," *Int. J. Eng. Educ.*, vol. 31, no. 4, pp. 1048–1057, 2015.

[46] Y. B.-D. Kolikant, M. Ben-Ari, and S. Pollack, "The anthropology semaphores," in *Proc. 5th Annu. SIGCSE/SIGCUE ITiCSE Conf. Innov. Technol. Comput. Sci. Edu. (ITiCSE)*, 2000, vol. 32, no. 3, pp. 21–24.

[47] S. Pamplona, N. Medinilla, and P. Flores, "Exploring misconceptions of operating systems in an online course," in *Proc. 13th Koli Calling Int. Conf. Comput. Edu. Res.- Koli Calling*, 2013, pp. 77–86.

[48] S. Pamplona, N. Medinilla, and P. Flores, "Assessment for learning: A case study of an online course in operating systems," *Int. J. Eng. Educ.*, vol. 31, no. 2, pp. 541–552, 2015.

[49] J. Dickinson, "Operating systems projects built on a simple hardware simulator," in *Proc. 31st SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 2000, vol. 32, no. 1, pp. 320–324.

[50] J. Randolph, G. Julnes, E. Sutinen, and S. Lehman, "A methodological review of computer science education research," *J. Inf. Technol. Educ. Res.*, vol. 7, pp. 135–162, Oct. 2008.

[51] S. B. Merriam, *Qualitative Research and Case Study Applications in Education*, 2nd ed. San Francisco, CA, USA: Jossey-Bass, 1998.

[52] M. Kurucay and F. A. Inan, "Examining the effects of learner-learner interactions on satisfaction and learning in an online undergraduate course," *Comput. Educ.*, vol. 115, pp. 20–37, Dec. 2017.

[53] A. Pérez-Dávila, "O.S. bridge between academia and reality," in *Proc. 26th SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 1995, vol. 27, no. 1, pp. 146–148.

[54] J. Nieh and C. Vaill, "Experiences teaching operating systems using virtual platforms and Linux," in *Proc. 36th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 2005, pp. 520–524.

[55] S. Hwang, "Blended learning for teaching operating systems with windows," in *Proc. 14th Annu. ACM SIGCSE Conf. Innov. Technol. Comput. Sci. Educ.*, 2009, p. 380.

[56] A. Schmidt, A. Polze, and D. Probert, "Teaching operating systems: Windows kernel projects," in *Proc. 41st ACM Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 2010, pp. 490–494.

[57] O. Laadan, J. Nieh, and N. Viennot, "Structured Linux kernel projects for teaching operating systems concepts," in *Proc. 42nd ACM Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 2011, p. 287.
[58] J. Andrus and J. Nieh, "Teaching operating systems using Android," in *Proc. 43rd ACM Tech. Symp. Comput. Sci. Educ. (SIGCSE)*, 2012, pp. 613–618.
[59] C. Dall and J. Nieh, "Teaching operating systems using code review," in *Proc. 45th ACM Tech. Symp. Comput. Sci. Educ.*, 2014, pp. 549–554.
[60] M. Morsiani and R. Davoli, "Learning operating systems structure and implementation through the MPS computer system simulator," in *Proc. 13th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 1999, pp. 63–67.
[61] J. L. Donaldson, "Architecture-dependent operating system project sequence," in *Proc. 32nd SIGCSE Tech. Symp. Comput. Sci. Educ. (SIGCSE)*, 2001, vol. 33, no. 1, pp. 322–326.
[62] T. Nicholas and J. A. Barchanski, "TOS: An educational distributed operating system in Java," in *Proc. 32nd SIGCSE Tech. Symp. Comput. Sci. Educ. (SIGCSE)*, 2001, vol. 33, no. 1, pp. 312–316.
[63] D. A. Holland, A. T. Lim, and M. I. Seltzer, "A new instructional operating system," in *Proc. 33rd SIGCSE Tech. Symp. Comput. Sci. Edu.*, vol. 34, 2002, pp. 111–115.
[64] B. Atkin and E. G. Sirer, "PortOS: An educational operating system for the Post-PC environment," in *Proc. 33rd SIGCSE Tech. Symp. Comput. Sci. Educ.*, 2002, pp. 116–120.
[65] D. Hovemeyer, J. K. Hollingsworth, and B. Bhattacharjee, "Running on the bare metal with GeekOS," in *Proc. 35th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 2004, pp. 315–319.
[66] H. Liu, X. Chen, and Y. Gong, "BabyOS: A fresh start," in *Proc. 38th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 2007, pp. 566–570.
[67] Y. P. Cheng and J. M. C. Lin, "Awk-Linux: A lightweight operating systems courseware," *IEEE Trans. Educ.*, vol. 51, no. 4, pp. 461–467, Nov. 2008.
[68] M. D. Black, "Build an operating system from scratch: A project for an introductory operating systems course," in *Proc. 40th ACM Tech. Symp. Comput. Sci. Educ.*, 2009, pp. 448–452.
[69] B. Pfaff, A. Romano, and G. Back, "The pintos instructional operating system kernel," in *Proc. 40th ACM Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 2009, vol. 41, no. 1, pp. 453–457.
[70] M. L. Corliss and M. Melara, "VIREOS: An integrated, bottom-up, educational operating systems project with FPGA support," in *Proc. 42nd ACM Tech. Symp. Comput. Sci. Educ. (SIGCSE)*, 2011, pp. 39–44.
[71] M. Goldweber, R. Davoli, and T. Jonjic, "Supporting operating systems projects using the μMPS2 hardware simulator," in *Proc. 17th ACM Annu. Conf. Innov. Technol. Comput. Sci. Edu. (ITiCSE)*, 2012, pp. 63–68.
[72] M. A. Holliday, "System calls and interrupt vectors in an operating systems course," in *Proc. 28th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 1997, pp. 53–57.
[73] A. B. Downey, "Teaching experimental design in an operating systems class," in *Proc. 13th SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 1999, vol. 31, no. 1, pp. 316–320.
[74] A. Polze and D. Probert, "Teaching operating systems: The windows case," in *Proc. 37th SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 2006, vol. 38, no. 1, p. 298.
[75] R. J. Sheehan, "Teaching operating systems with ruby," in *Proc. 12th Annu. SIGCSE Conf. Innov. Technol. Comput. Sci. Edu. (ITiCSE)*, 2007, vol. 39, no. 3, pp. 38–42.
[76] J. L. Donaldson, "Implementation of threads as an operating systems project," in *Proc. 39th SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 2008, pp. 187–191.
[77] S. Robbins, "Exploration of process interaction in operating systems: A pipe-fork simulator," in *Proc. 33rd SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 2002, vol. 34, no. 1, pp. 351–355.
[78] S. Robbins, "A disk head scheduling simulator," in *Proc. 35th SIGCSE Tech. Symp. Comput. Sci. Edu. (SIGCSE)*, 2004, vol. 36, no. 1, pp. 325–329.
[79] S. Robbins, "A UNIX concurrent I/O simulator," in *Proc. 37th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 2006, pp. 303–307.
[80] M. W. Goldberg, "CALOS: An experiment with computer-aided learning for operating systems," in *Proc. 27th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 1996, pp. 175–179.

SONIA PAMPLONA received the degree in computer engineer and the Ph.D. degree in computer science from the Universidad Politécnica de Madrid. She is currently a Ph.D. Associate Professor at the Universidad a Distancia de Madrid, UDIMA. She currently teaches undergraduate courses in operating systems and human–computer interaction, and a postgraduate course in mobile learning. Her area of research is engineering education, with a special interest in online learning.

NELSON MEDINILLA received the degree in electrical engineer from the Universidad de la Habana and the Ph.D. degree in information technology from the Universidad Politécnica de Madrid. He worked for 20 years as a Professor of electrical engineering. He currently teaches software design courses at the Information Technology Faculty, Universidad Politécnica de Madrid. He is also a Ph.D. Associate Professor at the Universidad Politécnica de Madrid. His areas of research include software design and engineering education.

PAMELA FLORES received the master's degree in information technologies from the Universidad Politécnica de Madrid (UPM) in 2011, and the Engineering degree in computer systems from the Escuela Politécnica Nacional (EPN) in 2005, and the Ph.D. degree from UPM. She is currently a Ph.D. Professor at EPN. She also coordinates the Doctorate in informatics and the Master in software at the EPN. Her research area is object oriented approach; in addition, she has worked on Qualitative Research in computer science.