Received: 26 May 2020 | Accepted: 3 July 2020

DOI: 10.1002/cae.22304

RESEARCH ARTICLE

WILEY

The effects of game-based learning in the acquisition of "soft skills" on undergraduate software engineering courses: A systematic literature review

Ivan Garcia[1] | **Carla Pacheco**[1] | **Francisco Méndez**[1] | **Jose A. Calvo-Manzano**[2]

[1]Division de Estudios de Posgrado, Universidad Tecnologica de la Mixteca, Oaxaca, Mexico

[2]Escuela Técnica Superior de Ingenieros Informáticos, Universidad Politécnica de Madrid, Madrid, Spain

Correspondence
Ivan Garcia, Universidad Tecnologica de la Mixteca, Carretera a Acatlima Km. 2.5., Huajuapan de León, Oaxaca 69000, Mexico.
Email: ivan@mixteco.utm.mx

Abstract

The software industry is becoming an increasingly important part of the economy in countries that have realized that the education of future software engineers is crucial to having a vibrant software industry. However, software engineering education is still influenced by traditional teaching impeding the acquisition of soft skills. Game-based learning (GBL) combines learning with different known resources, such as games, to support and improve the teaching/learning process and/or student evaluation through active learning. This study presents a systematic literature review on the use of GBL for teaching software engineering at the undergraduate level, from 2001 to 2020, by addressing four research questions: What kinds of games have been developed for software engineering education? Which software engineering areas have been addressed by these games? Which soft skills have been promoted by using these games? How have these skills been evaluated? The study found 96 studies to answer these four questions. The findings provided evidence on the development of digital games focused on teaching the fundamentals of software engineering defined by software engineering body of knowledge. Moreover, these games have been able to promote the acquisition of more than one soft skill which is beneficial for undergraduate students.

KEYWORDS

game-based learning, software engineering education, systematic literature review

1 | INTRODUCTION

Over the past few years, the software industry's demand for young engineers with competent knowledge and skills in software engineering has increased. However, the effectiveness of traditional teaching in software engineering education is still in question because it has not been able to produce efficient and qualified graduates. Therefore, according to Calderón et al. [22], there is a need to continue exploring modern alternatives with the aim of improving the future of software engineering education. The acquisition of soft skills, such as teamworking and communication, should be addressed at the undergraduate level to create young practitioners equipped with better skills [138]. In this regard, the software industry of the 21st century demands graduates skilled in the development of modern software systems, who are also aware of the importance of non-technical skills for achieving effective performance in the workplace. Although there is not a single widely recognized definition for the term "soft skills," most studies define it as the different social skills that software engineering students

must develop such as leadership, writing, negotiation, cooperation, and more. In fact, Itani and Srour [58] are more specific and state that many universities have recognized the need to expose their students to soft skills at the undergraduate level.

From this perspective, we are witnessing an increase in the implementation of not only diverse paradigms to improve learning in software engineering education but also in promoting soft skills among students. Game-based learning (GBL), for example, is defined as a pedagogical method of learning that takes advantage of digital and non-digital games to support the students' knowledge acquisition and cognitive development. Martí-Parreño et al. [86] state that GBL has the capacity to improve the way learning content is taught because students have a better control of their learning process while playing games. Therefore, by following a GBL approach, the concepts learned by students in the classroom can be put into practice, and consequently, their knowledge of these concepts can be enhanced.

With the aim of integrating the GBL approach into the educational context of software engineering, several studies have been conducted to provide evidence of its benefits as a complement to lecture-based courses. However, the authors are not aware of studies focused on exploring which specific soft skills are promoted by GBL, what kinds of games are more suitable for promoting each soft skill, and how these skills are evaluated at the undergraduate level. Therefore, we aim to present and discuss our experiences of applying the systematic literature review (SLR) to gather and evaluate the available evidence to help software engineering teachers and/or students to select the proper game for strengthening a specific soft skill.

The rest of the paper is organized as follows: Section 2 examines other related SLRs that explore the role of GBL in software engineering education at the undergraduate level. Section 3 describes the methodology used to perform our systematic review. Section 4 presents the results obtained by answering the defined research questions (RQs). Section 5 introduces some additional findings obtained from the studies. Section 6 presents the threats for validating our results, whereas Section 7 summarizes the main conclusions of this study and some future work. Finally, at the end of the sections, references for this study are included.

2 | RELATED WORK

The use of games in engineering education has been increasingly investigated by experts over the years. Although the motivating and immersive characteristics of GBL have also been widely studied, its implementation in specific educational contexts, such as software engineering education, requires more research [1]. With this aim in mind, some systematic reviews have been carried out to gather evidence on how GBL has been used to support software engineering education at the undergraduate level. These studies are briefly described in the following section.

Connolly et al. [30] presented a systematic review of empirical studies about the use of games for supporting undergraduate courses on software engineering. The SLR aimed to explore some latent problems related to requirements elicitation, analysis, and, in general terms, software engineering by addressing one RQ: How have technology and games changed during the last decades? The most significant results can be summarized as follows: (a) most of the computer games designed under a GBL approach are used for supporting teaching in project management and requirements analysis; (b) four of these games have a certain maturity level as they incorporate at least one type of evaluation; and (c) the impact of GBL on education is limited and, in some cases, nonexistent. Taking into account these results, this study concludes that there are not enough longitudinal studies to confirm and/or evaluate the effectiveness of GBL in software engineering education.

Moreover, Caulfield et al. [26] conducted a systematic review to analyze the contribution of games in improving software engineering education. The SLR gathered evidence about the effectiveness of existing games in providing a solid background which enables students to understand the challenges that any software project involves. The systematic review aimed to answer the following RQs: Which games already exist for the educational context of software engineering and how effective are they? The SLR identified 36 studies to produce the following findings: (a) most of the games are focused on addressing concepts from two software engineering body of knowledge (SWEBOK) knowledge areas, management process and software development; (b) most studies are conducted in Europe and North America; (c) there is a growing trend for using computer games instead of the board or card games; and (d) most games establish learning objectives by taking into account the first rung of Bloom's taxonomy (knowledge). This study states that, according to the analyzed evidence, games can help students to strengthen the theoretical concepts learned by traditional teaching. However, more rigorous research is required to properly validate the games' efficacy as most studies followed a nonexperimental design, and many dealt with very small sample sizes.

Research by Kosa et al. [70] introduced an SLR to identify empirical studies focused on the use of games in

software engineering education. This study identified 53 studies to answer the following RQs: For which kinds of learning purposes are games used in software engineering education? Are they mostly used in the sense that students play games or students develop games in the learning process? What kinds of games are used in software engineering education? Are they digital or non-digital? How positive is the outcome when using games in this context? Do the studies being carried out provide empirical results? Are there any design guidelines specific to the software engineering domain? What is the distribution of studies that have been carried out over a specified time period? Is there an increasing or decreasing trend? The obtained results of the SLR suggest that the reviewed studies can be classified into five main categories: games that are played by students; games that are developed by students as capstone projects; curriculum proposals; development of new approaches, tools, frameworks, or suggestions; and studies that do not fit into any of the above categories. Moreover, the obtained results state that (a) there is an ever-increasing adoption of games in software engineering education; (b) it is necessary to conduct more research on this approach due to the lack of empirical evidence; and (c) it is also necessary to establish a methodology or guideline for designing and evaluating games in this particular domain. Finally, this study supports the notion that there are no solid longitudinal studies that show that learning is more appropriate with game-based approaches with regard to traditional methods.

Finally, Souza et al. [137] presented the results of a systematic mapping study over 156 empirical studies on game-related methods for software engineering education. This study selected and extracted information to evaluate how GBL, game-development-based learning (GDBL), and gamification have supported the teaching process in specific areas of software engineering. With this aim in mind, the following five RQs were addressed: What game-related methods have been proposed for supporting software engineering education? What software engineering education knowledge areas are supported by the existing game-related methods? What types of games have been proposed or used to support software engineering education? What technologies have been used to support GDBL approaches? What game elements have been used for the gamification of software engineering education? The findings are summarized in two main statements: (a) software engineering education has been challenging since the early 1970s and games have represented a worthwhile complement to traditional teaching. However, more research is required to correctly identify the benefits of GBL, GDBL, and gamification; and (b) 69% of the primary studies are published in conferences, 29% in specialized journals, and 7% in workshops. From this evidence, the research states that although GBL involves different game genres and formats, the predominant style within digital games is the simulation-based game genre. Meanwhile, in the context of non-digital games, the importance of card games, board games, role games, and group activities is highlighted. It is common for digital games to be played individually, whereas non-digital games contribute to teamwork because they usually involve several players or groups.

In summary, all of the abovementioned works provide evidence of systematic reviews focused on analyzing empirical studies that identify different experiences from the use of GBL for enhancing software engineering education at the undergraduate level. However, in these studies, we are presenting and discussing our experiences of applying the SLR with the aim of evaluating which specific soft skills are promoted by the implementation of a GBL approach in undergraduate classrooms, as well as the games genres and formats that have more influence on their positive assimilation. Therefore, our efforts are focused on providing teachers, students, and/or game developers with a guide for selecting the appropriate type of game for teaching/learning a specific software engineering area while also effectively promoting certain soft skills. It is our premise that these aspects of GBL can lead to an improvement in the education of software engineering students, as well as acquiring some soft skills that may be considered desirable when students seek employment in the software industry.

3 | RESEARCH METHODS

An SLR aims to undertake and present an evaluation of the literature related to a specific research topic through the application of a rigorous methodology [13]. According to Kuhrmann et al. [71], this kind of review enables researchers to gather and evaluate all the available information on an RQ. With this in mind, our study followed the guidelines proposed by Kitchenham [66] and Kitchenham and Charters [67] to carry out SLRs in the software engineering context. The following subsections summarize the steps used to conduct our SLR.

3.1 | Identifying the necessity for an SLR

The general benefits that GBL brings into the educational context of software engineering have been widely explored, documented, and reported by several researchers

[31,77,128,134,136]. However, the effects of GBL on the acquisition or strengthening of soft skills on undergraduate software engineering courses, for example, have not been previously explored. According to Yilmaz et al. [159], soft skills are currently an important factor for the software industry because they are considered when assigning appropriate roles to different team members, the success rate of a project automatically increases. Students frequently receive training by learning numerous concepts and activities focused on improving technical skills, but while this kind of knowledge is appropriate, not every student can perform all task types in software development due to a lack of social skills. Moreover, there is no single soft skill that enables software practitioners to fulfill all the roles within a software team or that a student can learn to gain better employment.

While it is true that there will be students whose personalities are more suited to analyzing requirements or producing code, all individuals must be able to work in teams, communicate with people, or lead an external group [25]. Therefore, the importance of teaching both technical and soft skills during lecture-based courses in software engineering is crucial for creating competent software engineers [75]. As previously mentioned, GBL has been useful in improving the technical knowledge of software engineering methodologies and techniques within educational contexts. However, software engineers also need soft skills to be successful in the workplace [24].

The research carried out by Ahmed et al. [4] analyzed the lack of understanding of the relevance of soft skills in software development. An analysis of 500 advertisements for IT positions found communication, interpersonal, analytical and problem-solving skills, and team skills to be the most critical soft skills. On the contrary, a study conducted by Jones et al. [60] revealed three important soft skills when applying for software industry positions, such as software developers: willingness to learn, critical thinking, and attitude. Unfortunately, the same study stated that software engineering curricula at the undergraduate level often emphasize on hard skills over soft skills. Finally, research by Florea and Stray [45] analyzed 400 job advertisements for software testers from 33 countries, 64% of which required soft skills. The study determined that the most important soft skills are communication, analytical and problem-solving skills, teamwork, fast learning, independent-working, and adaptability. One of the main conclusions of the study stated that there is a clear need for introducing soft skills into the undergraduate curricula to help new graduates rapidly adapt to their new positions.

The abovementioned studies are evidence of an increasing interest in exploiting the benefits that soft skills provide within academia and the software industry. However, soft skills must be enhanced to perform a specific role in a software team? How can these skills be promoted at the undergraduate level? Can GBL help with this task? Unfortunately, most of the published results are presented in such a way as to make them difficult to use by any party interested in improving soft skills through the use of GBL in real educational contexts. Therefore, the SLR enables us to (a) identify studies for collecting evidence on the type of games developed under a GBL approach to support undergraduate software engineering courses; (b) analyze the software engineering areas covered by these games; (c) identify which soft skills were enhanced by using these games; and (d) explore how the soft skills are evaluated in educational contexts.

3.2 | Defining the RQs

It was necessary to perform a systematic search of the literature to answer the following research questions:

RQ1: What types of games are developed under a GBL approach to support software engineering education at the undergraduate level?

RQ2: Which software engineering areas are covered by these games?

RQ3: Which soft skills are promoted among students by using these games?

RQ4: How are these soft skills evaluated?

Regarding RQ3, according to Kamin [62], soft skills can be seen as the combination of the abilities, attitudes, habits, and personality traits that allow people to perform better in the workplace, complementing the technical skills required to do their jobs and influencing the way they behave and interact with others. Moreover, current literature frequently relates the term "soft skills" to other accepted terms such as "interpersonal skills" and "social skills" that were also considered in the SLR as the following section explains.

3.2.1 | Establishing the search terms

The following keywords were derived from the previous RQs: "*GBL*," "*game*," "*soft skills*," "*education*," "*students*," "*undergraduate level*," "*area*," and "*evaluation*." The search strings were constructed using relevant terms based on the four RQs. Moreover, a list of synonyms for these keywords was obtained, for example in RQ1, which contains the keywords "*games*," "*GBL*," "*education*," and "*undergraduate level*":

Keywords (game OR play* OR amusement*) AND (GBL* OR game-based learning* OR*

game based learning OR educational game* OR serious game*) AND (education* OR learning* OR learnedness* OR study*) AND (undergraduate level* OR engineering level* OR undergraduate ship* OR university level* OR tertiary education*).*

In addition, the search terms were expanded using Word Net version 3.0 [113], Thesaurus and Soule's dictionaries of English synonyms. Therefore, the search terms have been adapted to match each of the RQs.

3.3 | Defining the searching strategy

Kitchenham [66] and Kitchenham and Charters [67] suggest exploring different electronic sources. Therefore, the following research databases were used:

- ACM Digital Library.
- IEEE Xplore.
- Springer Verlag.
- Google Scholar.
- ScienceDirect.
- Wiley-InterScience.

With the aim of identifying research material and locating potentially relevant studies, the search strategy of our SLR was focused on identifying published studies from journals, conference proceedings, reports, and/or books in the abovementioned electronic databases. Additionally, trial searches were performed using search strings constructed by the combination of the keywords and synonyms previously mentioned.

3.4 | Selecting studies

The selection of studies for this SLR was based on the following criteria and procedures.

3.4.1 | Establishing the criteria for studies selection

It is worth mentioning that a paper was considered as a primary study as long as it met the criterion of presenting empirical or practical data on the use of GBL to support software engineering education at the undergraduate level and enhancing students' soft skills. Moreover, the studies included in the SLR were selected according to the following inclusion criteria:

- The study directly answered one or more of the RQs.
- The study had been published between 2001 and 2020. The year 2001 was chosen as it was the year that the game-based learning concept was formally defined by Prensky [111].
- The study was focused on supporting the teaching of software engineering at the undergraduate level.
- The study contained data about the improvement of undergraduates' soft skills.

Since the focus of this SLR was the use of GBL to support software engineering education and the enhancement of soft skills at the undergraduate level, the following material was excluded:

- Slide presentations.
- Workshops (editorials and papers).
- Personal opinions, points of view, or anecdotes.
- Studies lacking empirical evaluation.
- Software engineering in non-educational contexts (e.g., industrial experiences).

3.4.2 | Defining the procedure for selecting studies

A preliminary selection of candidate studies was performed by analyzing the title, abstract, and keywords of each study. Furthermore, this analysis was extended to include the conclusions section when the title, abstract, or keywords did not provide enough information. Afterward, all the selected studies were analyzed against the inclusion criteria to obtain a set of primary studies. To avoid any repetition, it was necessary to carefully review all of the studies to find repeated publications; thus, if a similar study was presented in different publications (even with different authors), only the most recent and broadest study was included in the review. Similarly, if two studies had the same publication date and the same extension regarding the topic scope, only one study was considered.

3.5 | Evaluating the quality of the selected studies

According to Kitchenham and Charters [67], the evaluation of quality over the selected studies can be used to lead the synthesis' interpretation of the achieved findings and to determine the strength of the elaborated inferences. Therefore, in the context of our SLR, the quality of each accepted study was evaluated considering the three criteria shown in Table 1.

The first criterion provided evidence for evaluating whether the study had clearly achieved the purposes and

TABLE 1 Quality evaluation

Evaluation criterion	Answer option for scoring (field in endnote)	Score
C1. Is the aim of the research clearly explained?	Yes = 1/ Moderately = 0.5/ No = 0	
C2. Is the presented approach clearly explained?	Yes = 1/ Moderately = 0.5/ No = 0	
C3. For this study, what is the acceptance quality rate based on the findings?	Not findings or under 20% = 0/ Over 80% = 1/ Between = 0.5	
%	Enter the percentage for the quality evaluation field in Endnote	

aims of the presented research. In the context of our SLR, the reviewed studies satisfied the criterion for the C1 question. The second criterion also provided evidence to evaluate whether the studies had provided sufficient information (either directly or by referencing other relevant literature) to establish the context and background for our SLR. The evaluation of C2 was also positive for almost all the publications (~90%). Finally, the third criterion provided evidence for evaluating if the outcome of the analyzed study was sufficient for our research purpose. The C3 question was positively answered by almost all of the publications (~82%).

It is important to mention that the evaluation measures were established by two experienced researchers from the Universidad Tecnológica de la Mixteca, México, and validated by an independent researcher who is not one of the paper's authors. Moreover, these three researchers carried out the process of evaluating the paper's quality. However, the scoring is the only heuristic that is used as a guide and no study was rejected based on its quality score. Thus, data from 96 studies were normalized by combining the percentage obtained with the quality criterion.

3.6 | Defining the strategy for data extraction

Research by Beecham et al. [13] stated that each study used to answer the defined RQs should be recorded on a separate form to correctly understand the findings. Therefore, Endnote version X9 was used for data extraction to record the references for each study. In the context of our SLR, this extraction process enabled us to identify categories for the information presented in Section 4 of this paper.

3.7 | Synthesizing the extracted data

The guidelines proposed by Kitchenham [66] and Kitchenham and Charters [67] do not provide a clear explanation about how much categorization must be done during data extraction and how much during data synthesis. Therefore, we decided to conduct a trivial extraction during the initial steps of the synthesis stage to obtain a list of minimally paraphrased quotes. Furthermore, Section 4 of this paper also shows the frequencies for each topic identified in diverse sources. It is important to mention that each occurrence received the same weighting, indicating that frequencies reflected how many times a given topic was identified in different studies as opposed to its relevance.

3.7.1 | Retrieving the documents

The searches enabled us to obtain 361 references, and, approximately, 150 of these were rejected after analyzing the title, abstract, and keywords of each study. Moreover, the number of false positives in the initial set of studies was low and included studies that seemed relevant at first, but it was decided that they were not after detailed research as they were focused on tools that supported the development of educational games within the educational context of software engineering. Thus, a careful review of 211 studies led us to obtain a final list of 96 studies. Table 2 summarizes the steps related to this selection process.

Moreover, two validation exercises were included in the selection process as follows:

- Validation 1—The *inter-rater agreement* depicts the consistency of the data extraction among the analyzed studies when two or more researchers evaluate each study [44]. Therefore, an inter-rater agreement test was conducted on the 211 studies found in the preliminary search. The primary research group, composed of experts from the Universidad Tecnológica de la Mixteca, carefully analyzed each one of these studies to accept a total of 131, whereas the independent researcher randomly analyzed 50 studies chosen among those rejected and accepted, with 92% conformity recorded. This percentage of the agreement provided certainty to

TABLE 2 Validated and reviewed studies

Selection process	Number of studies	Studies used in the validation
Studies that were extracted from electronic databases	361	n/a
Studies examined based on title, abstract, and keywords	211	n/a
Studies accepted by primary researchers	131	–
Studies rejected by the independent researcher (Validation 1)	119	10 studies rejected and 2 not found from the 50 studies that were randomly selected and that were part of the 131 accepted studies
Studies added by the independent researcher (Validation 1)	119	n/a
Studies rejected by Validation 2	96	23 rejected after Validation 1

Abbreviation: n/a, not applicable.

the original decisions to accept and reject the studies reviewed.

- Validation 2—The aim of the *independent evaluation* is that at least two researchers independently evaluate the methodological quality of each selected study [125]. Therefore, this evaluation was conducted using the 131 accepted studies and a high percentage of agreement (~82%) between the primary researchers and the independent expert was obtained. As a result of the validation, 96 studies were accepted for our research.

It is important to mention that, although Kitchenham and Charters [67] suggest applying the meta-analysis techniques, such as the data aggregation as part of the synthesis, this was impossible to do due to the unstructured sources and the high level of heterogeneity of the analyzed studies.

4 | RESULTS

The SLR found a total of 96 studies that were published during the period 2001–2020 to present different games focused on supporting software engineering education at the undergraduate level. The bibliography references for these 96 studies are provided in the references section. It is worth mentioning that some studies have answered more than one RQ or have covered diverse aspects related to the four questions. Furthermore, some of the studies address the use of games to improve the results of traditional lecture-based courses, whereas others highlighted the GBL characteristics to promote its advantages at the undergraduate level. Before presenting the data collected for answering each RQ we aim to first provide an overview of the general characteristics of the selected studies followed by the obtained results.

4.1 | Overview of the selected studies

The selected studies were directly classified by the types of games proposed without interpreting the study content. Therefore, the studies were classified by means of the platform employed to play each game as follows:

- Digital games: Studies focused on supporting software engineering education by using games designed for digital platforms such as computers, mobile phones, and/or video game consoles.
- Non-digital games: Studies focused on supporting software engineering education by using games designed to be played traditionally, such as board games and cards.

Therefore, 83.3% of studies are digital games and 16.7% of studies are non-digital games. Moreover, 100% of the empirical studies came from academic experiences.

4.2 | Effects of GBL on students' soft skills

By answering the four abovementioned RQs, we aim to provide a broad picture of what the literature has reported with regard to the use of GBL on the enhancement of students' soft skills in lecture-based courses on software engineering. With this aim, we have collected information on what types of games are developed under a GBL approach to support software engineering education at the undergraduate level (RQ1), which software engineering areas are covered by these games (RQ2), which soft skills are promoted among students by using these games (RQ3), and how these skills are evaluated (RQ4).

4.2.1 | RQ1—Types of games developed under a GBL approach to support software engineering education at the undergraduate level

A total of 96 studies were identified to answer *RQ1: What types of games are developed under a GBL approach to support software engineering education at undergraduate level?* To respond to this question, we first reviewed studies published between 2001 and 2020 to identify the games that were successfully developed under a GBL approach. Research by Qian and Clark [114] stated that successful GBL is not just providing students with a game and magically expecting to obtain increased motivation and improved knowledge acquisition. On the contrary, educational games must be designed and researched paying careful attention to contemporary learning theories (e.g., constructivism, constructionism, experiential learning, situated learning). Following this suggestion, we found 80 digital games that were successfully developed and classified as *computer games* (~71.2%), played on operating systems such as Windows, Linux, or macOS; *web games* (~17.5%) that are played on Internet browsers such as Chrome or Firefox; *mobile games* (~10%) that are played in tablets, mobile phones, iPhones, or iPads; and *tangible games* (~1.2%) that enable students to work directly with physical elements (see Table 3).

In this regard, one study was found incorporating a game-based approach based on the interaction with a Tangible User Interface (TUI) to represent programming concepts in a visible form [34]. Research by Xis et al. [157] stated that TUIs offer interactive couplings of physical artefacts with computationally mediated digital information. Therefore, we categorized the above-mentioned study as a digital game as it presents a distributed TUI to operate as a wireless sensor network connected to a computer through a compact USB radio link.

On the contrary, non-digital game-based learning has also introduced many advantages to traditional teaching at the undergraduate level (e.g., cost-effectiveness, low administrative overheads, little or no learner and teacher skills required, improved social interaction). Although these types of games may be less popular and not as fashionable their digital equivalents, they are long-established as a pedagogical practice in education [100], covering all types of physical, board, and card games. In the context on this SLR, we found 16 studies related to non-digital games developed to support lecture-based courses in software engineering at the undergraduate level. These studies were classified as traditional *tabletop games* (~93.8%) that are normally played on a table, and *toys* (~6.2%) that enable students to experiment by doing something with physical objects. Table 4 shows that the most used non-digital game in software engineering education is tabletop games.

Furthermore, the collected information enabled us to identify Supporting Information that extended the answer for RQ1. This means that categorization for the digital game genre was obtained by taking into account the classifications suggested by Arsenault [6] and Konzack [69] (see Table 5).

Therefore, we have classified the collected studies into the following genres:

- Simulation: Simulation games can represent one of many different types of situations and variables (e.g., business/trade, construction, life, sports, war, and more). Simulations are more realistic models of real-life situations and/or variables that other games do not consider. In this context, we found a total of 27 studies that provide evidence that the most used genre in digital games designed for supporting software engineering education at the undergraduate level is a simulation.
- Puzzle: Puzzle games are mainly focused on providing scenarios for solving puzzles without much narrative. The types of puzzles can test many problem-solving skills in students including logic, pattern recognition, sequence solving, answering questions, and word completion. We found a total of 14 studies for this category.
- Role-playing game (RPG): RPGs are focused on character development through narrative and obtaining

TABLE 3 Studies related to digital games

Type of game	Referenced studies	Frequency
Computer game	[2,7,9,10,16,21,23,27,28,36,38,41–43,48,49,52,56,59,65,68,78–82,85,88,94,95,99,102,103,106,107,110,112,116–118,120–124,129,130,133,139,140,147,154,158,160–163]	57
Web game	[14,15,17,33,51,57,73,87,93,98,101,141,142,148]	14
Mobile game	[5,64,74,84,127,131,155,156]	8
Tangible game	[34]	1

TABLE 4 Studies related to non-digital games

Type of game	Referenced studies	Frequency
Tabletop game	[8,39,47,53,55,91,96,97,126,132,144–146,149,150]	15
Toys	[72]	1

better benefits. It is worth mentioning that character development in RPGs does not necessarily take place as in traditional stories, but students learn new abilities or improve existing ones. We found a total of 14 studies for this category.

- Trivia: A trivia game tests the students' knowledge on specific topics or their abilities to solve challenges by answering questions to obtain some kind of reward. We found eight studies for this category.
- Strategy: A strategy game, also called a strategic game, promotes uncoerced and autonomous decision-making skills among students to achieve the desired outcome. This genre also encourages student participation through experiencing a story based on dialog and puzzle solving. We found a total of 7 studies for this category.
- Platform: Platform games, or platformers, can be normally both two-dimensional (2D) and 3D games in which the player controls a character who runs, jumps, or climbs onto platforms of various elevations, floors, ledges, and stairs horizontally or vertically depicted on the screen. We found a total of four studies for this category.
- Adventure: Adventure games involve more narrative as the student normally plays a fantasy role in an episodic adventure story. Moreover, the games in this genre revolve mostly around solving puzzles with the aim of following the outlined narrative. We found two studies for this category.
- Action: An action game frequently emphasizes physical challenges, reflexes, hand–eye coordination, and reaction time. The main mechanics of these games are accuracy, movement, quick decisions, timing, among others. In this context, we found one study for this category.
- Racing/driving: Racing/driving games allow students to race or drive simulated vehicles in a relaxed manner. Players can race in vehicles, on motorcycles, on foot, or in completely abstract graphics throughout the whole game, not just in short sequences. We found only one study for this category.
- Arcade: Arcade games are action games with a very simple gameplay interaction similar to the old coin-op arcade games. These games usually require little puzzle solving or tactical thinking and rely solely on testing the student's response time. In summary, we found one study for this category.
- Fighting: Fighting games simulate close physical combat and allow students to engage in what are usually one-on-one combats until one of the characters is knocked out. We found one study for this category.

In summary, Table 5 shows the studies found to extend the answer for RQ1 in relation to digital game genres.

TABLE 5 Genre classification for digital games

Genre	Referenced studies	Frequency
Simulation	[5,7,9,10,15,21,23,33,34,36,38,48,56,64,78,81,84,102,107,116,117,120,124,129,142,158,161]	27
Puzzle	[2,17,27,43,51,65,74,80,85,88,94,103,123,163]	14
Role playing	[14,28,42,49,52,82,101,106,112,133,140,141,148,154]	14
Trivia	[41,57,87,93,110,127,156,162]	8
Strategy	[16,59,95,99,118,121,139]	7
Platform	[68,73,130,160]	4
Adventure	[79,155]	2
Action	[147]	1
Racing/driving	[131]	1
Arcade	[122]	1
Fighting	[98]	1

Similarly, we have categorized the genre for the non-digital games by taking into account the classification suggested by Connolly et al. [29] and Battistella and von Wangenheim [12] as follows:

- Card game: A card game is normally any game that uses playing cards as its primary device. This type of game is usually fast, consisting of a number of movements that last only a few minutes each. This affords frequent opportunities for verbal socializing, and face-to-face interaction among students. We have collected evidence of eight studies that indicate that card games are the most used non-digital game genre in supporting software engineering education at undergraduate level.
- Board game: Board games involve the particular movement of small pieces on a board with a pattern following a set of rules. Some of these games are based solely on strategy, but many contain an element of chance, whereas some are based on purely chance with no element of skill. We found five studies for this category.
- Paper and pencil game: Paper and pencil games, also called paper and pen games, are games that are normally played with just paper and pencils (or any writing implement) in which nothing is allowed to be erased. In this context, we found one study for this category.
- Classic RPG: The classic RPGs involve the participants assuming the roles of characters in fictional settings by following a formal set of rules. This means that students take responsibility for acting out roles within a narrative and performing their characters' actions. We found one study for this category.
- Educational toy: An educational toy, also called an instructive toy, are objects of play normally designed to stimulate the students' learning. In fact, educational toys are a very useful way of making learning fun for students when learning very abstract concepts. We only found one study for this category

Table 6 summarizes the studies found to extend the answer for RQ1 regarding non-digital games genres.

Furthermore, from these findings, it can be stated that digital games are still in greater use than non-digital games for software engineering education at the undergraduate level. Moreover, taking into account the number of studies analyzed, computer games (~71.2%) are the most used type of digital game for this purpose. On the contrary, 93.8% of studies provide us with evidence that tabletop games are the most suitably designed type of non-digital game to support learning on undergraduate courses. In addition, most of the digital games show a marked preference for three genres: simulation (~33.8%), puzzle (~17.5%), and role playing (~17.5%). We believe that the popularity of the simulation genre is due to the fact that these games are designed to create immersive environments where students can put into practice the theoretical concepts learned in classrooms to maximize their transferability to the real industry. In this regard, computer games focused on simulation to help teachers to present software engineering topics in more effective ways than on lecture-based courses. Regarding the non-digital games, two genres have attracted the attention of developers: card games (~50%) and board games (~31.3%). In this regard, we think that, in the educational context of software engineering, the preference for both genres can be attributed to the fact that they are easy to play, encourage collaborative learning, provide immediate feedback to students, and promote face-to-face interactions.

Finally, it is worth mentioning that, although we could not find more studies related to the use of toys for software engineering education at the undergraduate level, the use of the LEGO bricks [72] showed a positive impact on students when they had to discuss complex concepts about requirements engineering area. Therefore, we believe that this toy may have the potential for teaching software engineering topics where understanding abstract concepts of the system-to-be are essential.

TABLE 6 Genre classification for non-digital games

Genre	Referenced studies	Frequency
Card game	[8,53,91,97,126,132,144,145]	8
Board game	[39,47,55,96,149]	5
Paper and pencil game	[150]	1
Classic role-playing game	[146]	1
Educational toy	[72]	1

4.2.2 | RQ2—Software engineering areas covered by the identified games

The same 96 studies were useful for answering *RQ2: Which software engineering areas are covered by these games?* According to these studies, the identified games have provided familiar and engaging contexts for undergraduate students in courses on software engineering methods, techniques, and standards. According to Mayo [92], games should have the potential to address many deficiencies of the engineering education system for five reasons: *massive reach* (i.e., games give teachers the

ability to reach students where they live), *effective learning paradigms* (i.e., games can implement instructional strategies leading to improved learning outcomes), *enhanced brain chemistry* (i.e., games are able to chemically "prepare" the brain for learning), *time on task* (i.e., games can deliver educational content and double the time spent learning at home), and *learning outcomes data* (i.e., games can increase learning outcomes when compared to lecture-based courses). Research by Braghirolli et al. [19], for example, stated that educational games are familiar to undergraduate students and they can be a viable method for engaging them in learning the concepts of any engineering topic, stimulating interest, and increasing motivation to learn and better understand theoretical content throughout a course. In this regard, the 96 studies identified by RQ1 provided us with evidence that the use of games motivates students across a range of software engineering areas, from requirements elicitation to software project planning and monitoring. Therefore, with the aim of providing a formal answer for RQ2, we first homogenized the software engineering areas by taking into account the classification proposed by the SWEBOK v3.0 [18]. This body of knowledge is composed of 15 main knowledge areas (software requirements, software design, software construction, software testing, software maintenance, software configuration management, software engineering management, software engineering process, software engineering models and methods, software quality, software engineering professional practice, software engineering economics, computing foundations, mathematical foundations, and engineering foundations) that are broken down into several topics.

Considering the abovementioned categorization, we intended to respond to RQ2 by identifying which knowledge areas from the SWEBOK were addressed by the collected studies. In this regard, research by Stevens and Norman [138] states that there has been a major emphasis on software engineering courses for developing students' technical skills. According to Hamlet and Maybee [54], technical skills are the abilities that reinforce the knowledge of the software engineer to use the methods and techniques required throughout the software lifecycle for correctly performing a specific task. Some examples include knowledge of programming languages, software tools, requirements elicitation techniques, among others. In addition, many studies have also stated that the knowledge needed to be an efficient software professional includes technical skills, expertize, and teamwork [61,75,143].

Therefore, with the aim of answering RQ2, we mapped the technical skills covered by the digital and nondigital games found in their corresponding SWEBOK knowledge areas. Thus, Table 7 shows that 36 technical skills were identified from the 80 digital games previously identified by RQ1. The results provided us with evidence that these skills could be categorized into 11 SWEBOK knowledge areas: computing foundations, software engineering process, software requirements, software engineering models and methods, software engineering management, software engineering professional practice, software construction, software quality, software design,

TABLE 7 SWEBOK knowledge areas covered by digital games

Technical skill	SWEBOK knowledge area	Referenced studies	Frequency
Formal specification	Software requirements	[110]	6
Requirements analysis		[122]	
Requirements elicitation and analysis		[52]	
Requirements elicitation		[38,48,127]	
Project management	Software engineering management	[10,81,107]	3
Software development	Software engineering professional practice, software engineering process	[14,28]	2
Mobile device programming	Software construction, software engineering professional practice	[64,156]	2
Software design concepts	Software design	[84,123]	2
Software inspection	Software construction, software quality	[112]	1
Usability (design, testing, and prototyping)	Software design, software testing	[15]	1
Preventive and corrective maintenance	Software maintenance	[121]	1

Abbreviation: SWEBOK, software engineering body of knowledge.

software testing, and software maintenance. The majority of the studies (~57.5%) have proposed digital games focused on addressing topics related to the computing foundations knowledge area, mainly for improving the knowledge related to programming concepts, data structures, object-oriented programming paradigm, and debugging. Moreover, these studies aimed to improve the way that basic computing concepts are taught by developing a better understanding among students about how to properly develop systems and software during gameplay. Knowledge areas such as software engineering models and methods (~11.3%), software engineering process (~8.8%), and software requirements (~7.5%) have been addressed to a lesser extent by the games. Considering the software engineering models and methods knowledge area, it is worth mentioning that digital games have recently become an important learning resource for teaching the fundamentals of software process standards at the undergraduate level. In fact, five games were developed during 2017–2020 for teaching the ISO/IEC 12207:1995, ISO/IEC 29110, ISO/IEC 12207:2008, ISO 21500, and ISO/IEC/IEEE 29148 standards. These studies provide evidence that a recent alternative strategy for software engineering education is to complement traditional education with serious games.

In addition, the digital games designed for the software requirements knowledge area only addressed two processes of the requirements engineering area: elicitation and analysis. Some studies stated that the use of games in requirements elicitation education has attracted the attention of developers, teachers, and students [104,119]. In fact, the same studies stated that games can be applied to requirements elicitation training as they can enhance the students' understanding by replacing traditional lecture-based courses with simulations of real-world situations. Moreover, games can provide students with a different perspective of these repetitive and time-consuming activities by combining practical understanding with theoretical knowledge. Thus, the abovementioned studies have suggested that games can be useful in providing meaningful examples in a software engineering context that mostly rely only on theory or that are hard to simulate in educational projects, such as requirements elicitation and analysis. Other knowledge areas, such as software engineering management, software engineering professional practice, software construction, software design, software quality, software testing, and software maintenance, were also addressed by educational games between 2011 and 2020.

Although evidence was collected from only one study per area, technical skills that were not identified by previous SLR's have now been encouraged by the use of games (e.g., software inspection, usability, and preventive and corrective maintenance). In this regard, we believe that the application of GBL on undergraduate software engineering courses can lead to good academic results, with practical knowledge having a positive impact on theoretical knowledge.

On the contrary, Table 8 shows that eight technical skills were identified from the 16 non-digital games previously collected. As can be seen, two knowledge areas (i.e., software engineering management and software engineering models and methods) have been mostly addressed by the non-digital games to improve practical knowledge on project management (~31.3%). Moreover, specific topics such as earned value, technical doubt, Kanban and lean thinking, the ISO/IEC 29110 standard, and Scrum were also addressed by these games. Similarly, as with digital games, in 2016 two card games were designed for enhancing learning with regard to the ISO/IEC 29110 standard.

TABLE 8 SWEBOK knowledge areas covered by the non-digital games

Technical skill	SWEBOK knowledge area	Referenced studies	Frequency
Project management	Software engineering management	[47,91,96,126,149]	5
ISO/IEC 29110	Software engineering models and methods	[144,145]	5
Scrum		[55,97,150]	
Software engineering (introductory concepts)	Software engineering process	[8,72]	2
Programming (data structures)	Computing foundations	[53]	2
Operating systems		[146]	
Requirements elicitation, analysis, validation, management	Software requirements	[132]	1
Software metrics	Software quality	[39]	1

Abbreviation: SWEBOK, software engineering body of knowledge.

These studies aimed to achieve the same goal as those related to digital games, creating educational tools as a strategy to encourage learning about the standard at the undergraduate level.

One more card game and one educational toy were found for the software engineering process knowledge area. Both of these studies were specifically focused on improving the students' understanding of some introductory software engineering concepts. In addition, ~12.5% of studies encouraged the strengthening of technical skills in computing foundations knowledge areas, primarily in topics such as programming and operating systems. The software requirements knowledge area was also addressed by one non-digital game designed to reinforce the requirements engineering practices among undergraduate students from the Institute of Technology of Cambodia. Finally, a board game for teaching software metrics and quality attributes was presented in 2019.

As can be seen, most of the digital and non-digital games identified (~50%) were designed to cover the computing foundations knowledge area, most of which had a focus on strengthening programming skills. According to Lindberg et al. [76], games can be included in curricula to motivate students when learning programming concepts during gameplay. These games have to be combined with traditional lecture-based courses to achieve better results. Furthermore, it is clear that the studies provided enough evidence to support the statement in RQ2. Moreover, an important orientation of games as a supporting tool for education in software engineering models and methods (~14.6%), software engineering process (~9.4%), software engineering management (~8.3%), Scrum (~7.3%), and software requirements (~7.3%) is also observed. Nevertheless, students must also be prepared to build software systems as part of a team with the aim of improving their social/soft skills. The following section will explore these themes.

4.2.3 | RQ3—Soft skills promoted by the identified games

Research by Matturro et al. [89] stated that the importance of soft skills can be equal to or even greater than technical skills because interaction and teamwork among professionals are needed to achieve a project's objectives in the context of software development. Therefore, the success of a software project does not only depend on the technical skills that the team members may possess. According to Ahmed et al. [3], the combination of soft skills and technical skills could increase the practitioners' efficiency when designing and deploying software projects with the aim of contributing to the organization's objectives. In other words, software development organizations can benefit from leveraging the soft skills of their employees so as to efficiently meet the objectives defined for any particular software project. However, soft skills can be promoted by a GBL approach? To collect relevant information on this issue, we used the 96 studies to answer *RQ3: Which soft skills are promoted among students by using these games?* As a first step in responding to this question, we reviewed studies published between 2001 and 2020 to identify a set of skills that, according to Matturro et al. [90], can be categorized as soft skills. More specifically, we looked for studies related to skills such as analytical skills, autonomy, change management, commitment/responsibility, communication skills (oral/written), competition, conflict management, creativity, critical thinking, customer orientation, decision-making, ethics, fast learning, flexibility, initiative, innovation, interpersonal skills, leadership, listening skills, methodical, motivation, negotiating skills, organizational/planning skills, presentation skills, problem-solving skills, results orientation, stress management, team management, teamwork, time management, and willingness to learn. In this regard, research by Kasemsap [63] stated that games developed under a GBL approach provided students with an immersive learning experience capable of capturing their attention through fun, motivating, and interesting environments that can foster collaborative learning and develop the abovementioned soft skills.

In this regard, we found that while some studies promoted only two soft skills, others were focused on promoting different combinations of these skills. Table 9 shows that from the 31 soft skills previously listed and identified by Matturro et al. [90] for the context of software engineering, 15 soft skills were identified in the collected studies.

More specifically, motivation, willingness to learn, competition, and problem-solving skills were the most promoted soft skills by the digital games with 80 occurrences (100%), 58 occurrences (~72.5%), 54 occurrences (%67.5), and 46 occurrences (~57.5%), respectively. These four soft skills were followed by communication skills with 31 occurrences (~38.8%) and decision-making which was promoted by 27 studies (~33.8%). In addition, customer orientation, interpersonal skills, and teamwork were promoted to a lesser extent at the undergraduate level with 24 studies (~30%). Moreover, time management was promoted by 22 studies (~27.5%), whereas analytical skills and leadership were addressed by 19 digital games (~23.8%) each. Commitment/responsibility and initiative were addressed by only seven (~8.8%) and four studies (~5%), respectively. Finally,

TABLE 9 Soft skills promoted by using digital games

Soft skill	Referenced studies	Frequency
Competition, motivation, problem-solving skills, willingness to learn	[5,16,17,27,34,51,56,57,59,68,74,78,85,87,94,95,103,116,117,123,124,129,140,147,161–163]	27
Analytical skills, communication, competition, customer orientation, decision-making, interpersonal skills, leadership, motivation, teamwork, time management, willingness to learn	[14,23,28,33,41,49,64,80,81,93,101,102,106,107,131,139,142,148,158]	19
Decision-making, motivation, problem-solving skills	[2,43,79,110,121]	5
Motivation, problem-solving skills, willingness to learn	[36,88,98,118,155]	5
Communication, competition, motivation, willingness to learn	[7,21,112]	3
Commitment/responsibility, motivation	[42,130,154]	3
Communication, customer orientation, interpersonal skills, motivation, problem-solving skills, willingness to learn	[38,48,127]	3
Commitment/responsibility, competition, motivation, problem-solving skills	[9,84,99]	3
Initiative, motivation	[160]	1
Competition, initiative, motivation, problem-solving skills, teamwork	[15]	1
Communication, decision-making, motivation, teamwork, time management	[120]	1
Communication, customer orientation, motivation, teamwork, time management	[52]	1
Communication, customer orientation, decision-making, motivation	[122]	1
Communication, competition, motivation, teamwork	[141]	1
Communication, initiative, motivation, problem-solving skills	[133]	1
Communication, interpersonal skills, motivation, teamwork	[82]	1
Decision-making, motivation, teamwork, time management	[10]	1
Interpersonal skills, motivation, problem-solving skills	[65]	1
Commitment/responsibility, initiative, motivation	[73]	1
Ethics, motivation, willingness to learn	[156]	1

ethics was promoted, in combination with other skills, by only one study. As can be seen with the collected studies, motivation is the most promoted soft skill by digital games on undergraduate software engineering courses. These findings seem to be in accordance with previous research by Rahman et al. [115] and Sousa and Rocha [135] which state that motivation is an intrinsic soft skill of any game created following the GBL approach. Willingness to learn, competition, and problem-solving skills were some other soft skills that have been mostly promoted by digital games during 2001–2020.

Similarly, the studies related to the non-digital games provided evidence on the soft skills promoted at the undergraduate level. Table 10 shows that 10 soft skills from the 31 identified by Matturro et al. [90] were identified in the collected studies.

In addition to the intrinsic skill of motivation, the studies showed that non-digital games have frequently promoted interpersonal skills (i.e., social interaction) and decision-making with 16 occurrences (100%). This could be due to the fact that these games usually provided mechanics that involved face-to-face interaction between students to promote social interaction. Moreover, communication skills and teamwork were promoted by 13 studies (~81.3%) each. Analytical skills and commitment/responsibility were promoted by 10 studies (~62.5%), whereas customer orientation and problem-solving skills were the least promoted skills with only three occurrences each (~18.8%).

These results do not mean that the GBL approach works better when designing digital games focused on supporting undergraduate software engineering courses. In simple terms, both digital and non-digital games can be designed to improve and develop specific skills in students depending on the course objectives (e.g., some teachers are more concerned with promoting analytical

TABLE 10 Soft skills promoted by using non-digital games

Soft skill	Referenced studies	Frequency
Analytical skills, commitment/responsibility, communication, decision-making, interpersonal skills, motivation, teamwork, time management	[47,55,91,96,97,126,144,145,149,150]	10
Communication, customer orientation, decision-making, interpersonal skills, motivation, teamwork	[8,72,132]	3
Decision-making, interpersonal skills, motivation, problem-solving skills	[39,53,146]	3

and communication skills in students during a requirements engineering course, while promoting creative and problem-solving skills might be more important during a programming course). In addition to these findings, we have classified the collected information with the aim of extending the answer to RQ3 and to provide additional information for relating the game genres with the promoted soft skills. In this regard, the studies provided us with information about which game genres could be more appropriate in developing specific soft skills than others on software engineering courses. We think that this information would be useful for a teacher who is looking to develop and use a game to promote a specific set of soft skills among undergraduate students.

The analysis was focused on the SWEBOK knowledge areas that, according to our SLR, were most explored under a GBL approach. Therefore, according to the collected evidence, simulation, puzzle, strategy, trivia, and role playing were the most widely used game genres when designing digital games to promote soft skills among undergraduate students (see Table 11).

An established pattern among the promoted soft skills enables us to identify that the simulation, puzzle, and strategy genres were effective in promoting competition, problem-solving skills, and willingness to learn in programming courses. The simulation, trivia, and role-playing genres have been useful in improving the teaching/learning process when components of software process standards or Scrum are taught. In fact, the promotion of analytical skills, communication, decision-making, teamwork, and time management was also observed. Finally, the simulation and role-playing genres were used when topics related to the software engineering process have been taught at the undergraduate level. In this case, soft skills such as competition, customer orientation, and leadership were promoted using these game genres.

Similarly, Table 12 shows the relationship between game genres and the soft skills promoted by the use of non-digital games. The card game, board game, and paper and pencil game genres were used to promote a total of seven soft skills (i.e., analytical skills, commitment/responsibility, communication, decision-making, interpersonal skills, teamwork, and time management) in undergraduate courses focused on supporting the learning of different project management topics, the ISO/IEC 29110 standard, and Scrum.

In conclusion, it is worth mentioning that the analyzed studies showed a clear tendency for exploring the benefits that different game genres can provide for promoting soft skills among undergraduate students on software engineering courses. According to the collected studies, we observed the relevance of the appropriate

TABLE 11 Digital game genres used for promoting soft skills in most explored knowledge areas

SWEBOK knowledge area (technical skills)	Game genres	Promoted soft skills	Referenced studies
Computing foundations (databases analysis and design, programming)	Simulation	Competition, problem-solving skills, willingness to learn	[5,9,33,34,36,56,78,116,117,124,129,161]
	Puzzle		[2,17,27,43,51,65,74,85,88,94,103,163]
	Strategy		[16,59,95,99,118,139]
Software engineering models and methods (ISO/IEC 29110, ISO/IEC 12207:1995, ISO/IEC 12207:2008, ISO/IEC/IEEE 29148, ISO 21500, Scrum)	Simulation	Analytical skills, communication, decision-making, teamwork, time management	[7,21,23,120]
	Trivia		[41,93]
	Role playing		[49,101]
Software engineering process (principles, processes, lifecycle, software process improvement)	Simulation	Competition, customer orientation, leadership	[102,142,158]
	Role playing		[106,140,141,148]

Abbreviation: SWEBOK, software engineering body of knowledge.

TABLE 12 Non-digital game genres used for promoting soft skills in most explored knowledge areas

SWEBOK knowledge area (technical skills)	Game genres	Promoted soft skills	Referenced studies
Software engineering management (project management)	Card game	Analytical skills, commitment/responsibility, communication, decision-making, interpersonal skills, teamwork, time management	[91,126]
	Board game		[47,96,149]
Software engineering models and methods (ISO/IEC 29110, Scrum)	Card game		[97,144,145]
	Board game		[55]
	Paper and pencil game		[150]

Abbreviation: SWEBOK, software engineering body of knowledge.

selection of game mechanics and dynamics in achieving this goal. Moreover, all of the studies reached the same conclusion on the importance of designing games that are not only capable of supporting student technical training, but also instilling soft skills among students to facilitate their subsequent entry into the software industry. In fact, we collected evidence on one digital game [156] designed to specifically promote the "ethics" soft skill among undergraduate students on software engineering. Therefore, it is possible to confirm that, according to the data collected for answering RQ3, simulation, puzzle, strategy, and role-playing genres can be suitable alternatives when looking to design a digital game which promotes the development of certain soft skills among students when teaching topics related to project management, programming, software process standards, and software engineering processes. On the contrary, if the objective is the development of non-digital games for supporting an undergraduate software engineering course, the board game, card game, or paper and pencil game genres can be useful to promote soft skills when teaching topics related to project management or software engineering process (e.g., software processes standards, Scrum).

4.2.4 | RQ4—Methods used to evaluate the soft skills promoted by the analyzed games

According to Calderón et al. [22] and Rodrigues et al. [119], the games designed for complementing software engineering education at the undergraduate level tend to be inadequately evaluated, making it difficult to determine their real impact on learning. In fact, some existing data collection methods have demonstrated their effectiveness and reliability in supporting this practice. In this regard, Petri and von Wangenheim [108] stated that the evaluation of educational games created for computing education is intended to verify results when using instructional strategies. Although this is true, not all the conducted evaluations can always provide consistent results on the matter. Moreover, the evaluation of games that were designed under a GBL approach frequently tend to involve slow and complex processes, and, for the context of our SLR, it was also necessary to explore how the proposed games and promoted soft skills have been evaluated to ensure the approach's effectiveness. Therefore, we used the 96 studies to answer *RQ4: How are these soft skills evaluated?* Research by Branch [20] established that the evaluation of an instructional strategy is usually carried out through an empirical study. Thus, with the aim of presenting relevant information on this issue, our SLR only included studies that provided empirical results derived from a systematic evaluation conducted on the proposed games. The relevance of the empirical methods in the software engineering context is that they allow for the incorporation of human behavior into the adopted research strategy [152]. In other words, these empirical methods provide a scientific basis for software engineering. Therefore, as a first step in responding to RQ4, the process established by Wohlin et al. [153] for conducting an empirical study (see Table 13) was considered. In this regard, we have identified and used the most representative issues of this process in the analysis of the collected studies:

- Research design: The research design must specify a set of methods and procedures that will be used to collect and analyze information about the variables specified in the study. Therefore, a study can be classified as *experimental* (i.e., the study involves multiple groups of students with random assignments), *quasi-experimental* (i.e., the study uses either multiple groups of students or multiple moments of measurement, but with no random assignments), *nonexperimental* (i.e., the study does not use multiple groups, but it is conducted in a systematic way, such as case studies), or ad hoc (i.e., the study is conducted in an unsystematic manner and lacks of an explicit objective).

TABLE 13 Methods used for evaluating the soft skills promoted by the games

	Research design/referenced studies	Data collection instruments	Analysis methods	Evaluation methods
Digital	Experimental (13.8%) [15,38,48,73,74,94,98,130,131,133,141]	Questionnaires (~81.8%), pretest and posttest (~72.7%), only posttest (~18.2%), interviews (~9%)	Quantitative (~18.2%) Mixed (~81.8%)	PANAS
Digital	Quasi-experimental (10%) [9,17,42,107,140,148,155,156]	Questionnaires (~75%), pretest and posttest (~87.5%), interviews (~25%), observation (~12.5%), audio recording (~12.5%)	Qualitative (~12.5%) Quantitative (~25%) Mixed (~62.5%)	
Digital	Nonexperimental (23.8%) [5,7,27,34,36,49,51,68,78,80,81,88,93,112,117,147,161–163]	Questionnaires (~89.5%), pretest and posttest (~57.8%), only posttest (~10.5%), interviews (~21%), observation (~10.5%)	Qualitative (~10.5%) Quantitative (~5.3%) Mixed (~84.2%)	FDF, MEEGA
Non-digital	Nonexperimental (31.3%) [47,55,72,132,146]	Questionnaires (100%), pretest and posttest (~40%), interviews (~20%), observation (~20%)	Qualitative (~60%) Mixed (~40%)	MEEGA
Digital	Ad hoc (52.4%) [2,10,14,16,21,23,28,33,41,43,52,56,57,59,64,65,79,82,84,85,87,95,99,101–103,106,110,116,118,120–124,127,129,139,142,154,158,160]	Questionnaires (~92.8%), pretest and posttest (~9.5%), only posttest (~11.9%), interviews (~14.3%), observation (~7.1%)	Qualitative (~61.9%) Quantitative (~4.8%) Mixed (~33.3%)	EGBL, MEEGA, UGALCO,
Non-digital	Ad hoc (68.7%) [8,39,53,91,96,97,126,144,145,149,150]	Questionnaires (~90.9%), interviews (~9.1%)	Qualitative (~100%)	MEEGA, MEEGA+

Abbreviations: EGBL, effective game-based learning; FDF, four-dimensional framework; MEEGA, model for evaluation of educational game; PANAS, positive and negative affect schedule.

- Data collection instruments: With the aim of mapping the study's evaluation goal(s) to the collected data and subsequently analyzing and interpreting such data, measures and data collection instruments must be defined. Therefore, a study can use *questionnaires*, *tests*, *interviews*, *observation*, *challenges* or *exercises*, and/or *other instruments*.
- Analysis and interpretation methods: Once the data are collected it must be analyzed, taking into account the nature of the data and the study's evaluation goal(s). Thus, a study can use *qualitative* (i.e., methods that use the teachers' and/or students' perceptions or comments) and/or *quantitative* methods (i.e., methods ranging from descriptive statistics to inferential statistics) to answer the RQs and, consequently, achieve the evaluation goal(s).

Therefore, to answer RQ4, we first analyzed the research design on the collected studies. The majority of the studies related to digital games were conducted in an ad hoc manner (~52.4%), collecting students' informal perceptions after one or more game sessions through questionnaires and observations. Case studies were conducted for 19 studies (~23.8%). In these nonexperimental studies, the game evaluation was conducted in a systematic way and, after a game session, data on the students' perceptions were also collected by applying questionnaires. Moreover, soft skills were also evaluated by interviewing and observation. It was determined that 19 studies (~23.8%) adopted a more formal research design: 11 studies (~13.8%) followed an experimental design with students randomly assigned to the experimental and control groups. Students in the experimental group usually played the game as learning support during the lecture-based course, whereas students in the control group did not. Data were collected by pre- and post-measurements of knowledge before (pretest) and after a game session (posttest), and pre- and postquestionnaires were applied to assess the understanding of soft skills. The rest of the studies (~10%) followed a quasi-experimental design without a random assignment of students to the experimental and control group. In some studies, audio recordings were used to assess the students' perceptions of soft skills.

Unlike digital games, the evaluation of soft skills in studies related to the non-digital games was informally conducted. The majority of the studies (~68.7%) were carried out in an ad hoc manner supported by game sessions, the application of postquestionnaires, and interviews to collect the students' perceptions. The rest of the studies (~31.3%) used a nonexperimental design focused on the application of a prequestionnaire, a game session, and the final application of a postquestionnaire to conclude the evaluation. In some studies (~18.8%), interviewing and observation were also used to collect students' perceptions on the soft skills promoted by the games.

As can be seen, ~55.2% of the studies did not use a well-defined research design to conduct the evaluation on digital and non-digital games and promoted soft skills. These studies were conducted in an ad hoc manner without any clearly defined objectives or measures. Nevertheless, we collected evidence from some studies that used well-defined models created to specifically evaluate the perceived learning outcomes and effectiveness of games, such as the four-dimensional framework (FDF) [40], the evaluation framework for effective game-based learning (EGBL) [32], the scale for assessing user enjoyment of E-learning games (EGameFlow) [46], the model for evaluation of educational games (MEEGA) [128], the five-dimensional framework for simulation games evaluation (UGALCO) [105], and MEEGA+ [109], an update of MEEGA. Other collected studies adapted existing evaluation models that were designed for different purposes compared to game evaluation, such as the technology acceptance model (TAM) [37] and the positive and negative affect schedule (PANAS) [151]. We also observed that four studies used the goal–question–metric (GQM) paradigm [11] to obtain analysis questions related to the expected learning outcomes.

Regarding the data collection instruments used during the game's evaluation, the vast majority of the studies related to digital games (~88.7%) used informal questionnaires frequently composed using a number of Likert-scale responses. As part of these questionnaires, five studies used standardized questionnaires for data collection from well-defined evaluation models, such as PANAS, TAM, EGameFlow, MEEGA, the Science Motivation Questionnaire II [50], and UGALCO. In addition, 12 studies also included questions on demographic information about the students such as age, gender, previous experience, educational level, and previous knowledge. In 13 studies, the students were also interviewed after the game session with the aim of capturing their perceptions. In fact, one study used interviews as the only data collection method.

Only 30 studies reported the use of testing to evaluate students' knowledge before (pretest) and after (posttest) a game session. The majority of these tests were used in studies with nonexperimental design (case studies) which did not involve multiple groups of students with random assignment (~36.6%). Moreover, 13 studies only applied tests after a game session. Fewer studies (seven) also reported the combination of observation and questionnaires to collect data on the behavior and/or skills of the students. Finally, less frequent types of data collection methods include audio recordings or discussions among students.

On the contrary, ~93.8% of the studies related to non-digital games used questionnaires as the main method to collect information from students. Furthermore, a third of these studies used standardized questionnaires from two evaluation models: MEEGA and MEEGA+. Only one study included questions on demographic information about the participating students such as age, gender, educational level, and previous knowledge. In only two studies, the students were also interviewed to collect further data of their perceptions. Moreover, one of these two studies used interviews as the only data collection method. We collected evidence from only one study that applied tests before and after a game session. Similarly, only one study reported the use of observation as an additional method for collecting information on students.

The evaluation of the last defined issue, the analysis and interpretation methods, also gave us relevant information. In most studies (~55%), a mixed-method (qualitative and quantitative) approach was used to explore the games' benefits within undergraduate educational contexts. On the contrary, ~36.2% of the studies used qualitative methods to analyze the students' perceptions and comments that were collected through the application of open-ended questionnaires. For the rest of the studies (~8.2%), quantitative analysis methods were used. Considering all the studies related to digital games, the quantitative analysis methods used were descriptive statistics (~58.7%), inferential statistics (~15%), and a combination of both methods (~26.3%). Hypothesis testing was evident in 25 studies.

Regarding the descriptive statistics methods, the mean was widely used in 34 studies as a measure of central tendency, whereas other measures include median (five studies) and quartile (one study). In addition, a standard deviation (23 studies) was also used to measure the degree of variation from the central tendency. Other measures were sometimes used to examine the dependency between variables, such as the Cronbach's alpha coefficient (four studies), the Spearman's correlation coefficient (one study), and the Pearson correlation coefficient (one study). Moreover, measures of central tendency were combined with graphic visualization techniques such as histograms (21 studies), line charts

(nine studies), scatter plots (six studies), box plots (five studies), frequency diagrams (three studies), pie charts (two studies), and treemaps (one study).

From the 25 studies that used hypothesis testing with the aim of rejecting (or accepting) a hypothesis with regard to an analysis factor, we observed that *t* test (19 studies) was the most used parametric test to compare two sample means in a one-factor—two treatments design. Four more studies used the Wilcoxon test, a nonparametric alternative to the paired *t* test, whereas Mann–Whitney U tests were used in two studies. The studies that analyzed one factor with more than two treatments used analysis of variance (four studies), Kruskal–Wallis (four studies), analysis of covariance (three studies), or multivariate analysis of variance (one study).

The analysis and interpretation methods used in the studies related to non-digital games were quite different. The majority of the studies (~87.5%) used the qualitative method approach to obtain data through open-ended questionnaires and interviews. Only two studies used a mixed-method approach to explore the games' benefits on student skills. The quantitative analysis methods used were descriptive statistics (~87.5%) and the combination of descriptive and inferential statistics (~12.5%). It is worth mentioning that inferential statistics were used in two studies that were previously classified as nonexperimental. Although these studies were presented as case studies and tried to formalize the evaluation, they did not involve multiple groups of randomly assigned students (e.g., control group and experimental group), or use multiple moments of measurement.

In relation to the descriptive statistical methods used in the studies, the mean was used in eight studies as a measure of central tendency. Other measures include median (two studies), frequency distribution (two studies), central tendency (one study), and median (one study). These studies were also complemented by the standard deviation (three studies) to measure the degree of variation from the central tendency. Other measures were sometimes used to extend the analysis, such as Peirce's criterion (one study) and the sign test (one study). Moreover, graphic visualization techniques such as frequency diagrams (four studies), histograms (two studies), and pie charts (one study) were used for supporting data interpretation in studies.

Three studies carried out data analysis without using any formal qualitative method.

In summary, considering the collected evidence, it is possible to conclude that the soft skills promoted by both types of games (digital and non-digital) are still evaluated in an ad hoc manner lacking scientific rigor. These evaluations are mainly supported by the analysis of students' perceptions and informal comments after they participated in a game session. Informal questionnaires are the most used instrument for collecting these data from students. Nevertheless, over the past few years, very little evidence has been observed for applying standardized questionnaires provided by specialized game evaluation models and models created for a different purpose. Finally, the use of qualitative method approaches predominates in studies. Moreover, some other studies also use quantitative analysis methods, among which the use of descriptive statistics is widely prevalent.

5 | DISCUSSION AND IMPLICATIONS

The studies described in Section 3 also explored the use of games in software engineering education. As a result, some of the findings from our SLR are similar to those from previous works as outlined in the following bullets:

- As Connolly et al. [30], Caulfield et al. [26], and Kosa et al. [70] concluded, our SLR also found that more rigorous research strategies are required when the GBL approach is validated in real undergraduate courses. Most of the analyzed selected studies were conducted in an ad hoc manner suggesting that it is still necessary to follow true experimental designs that enable researchers to sufficiently demonstrate the GBL's pedagogical value.
- As Kosa et al. [70] and Caulfield et al. [26] stated, our SLR also found that the tendency to design digital games for software engineering education is still prevalent at the undergraduate level. Nevertheless, the capability of non-digital games should not be disregarded for specific areas such as requirements engineering and project management.
- As Connolly et al. [30] and Kosa et al. [70] concluded, our SLR also found that the majority of studies do not provide evidence of following a formal methodology for implementing GBL into a traditional software engineering course. Therefore, more attention must be paid to investigating which guidelines or heuristics are more appropriate for the context of software engineering education.

On the contrary, our SLR has contributed to identifying additional lessons that had not been previously analyzed. More specifically, our SLR differentiates from previous works in providing an in-depth discussion on formal classifications of game types used by each area of the SWEBOK that have been taught on undergraduate software engineering courses. Furthermore, our SLR also outlines which game genres are the most suitable

with regard to specific SWEBOK areas, the soft skills promoted by GBL over the past 20 years, as well as identifying which game genres have contributed to the promotion of specific soft skills. The collected information enabled us to provide some additional suggestions:

- There are still some software engineering areas that have not yet been explored by researchers, whereas other areas, such as programming and project management, appear to be more suitable for teaching with a GBL approach. However, regardless of the area being studied, simpler game designs would shorten the time taken to implement the GBL approach as part of an undergraduate course and also allow students to focus more on content instead of spending a disproportionate amount of time understanding a game. Moreover, the combination of digital and non-digital approaches can also produce better educational results.
- The collected information provides evidence that some game genres are more suited to the promotion of specific soft skills among undergraduate students. For example, this paper recommends using a combination of simulation and role-playing genres during the development of a digital game to strengthen leadership, teamwork, time management, and communication skills in areas that demand high coordination and collaboration such as project management or requirements engineering. Furthermore, card games and classic RPGs can be combined when designing non-digital games for promoting competition, decision-making, and analytical skills when it is fundamental for students to understand abstract concepts from areas such as programming, software process standards, or even agile methodologies.

- When collecting data about the effectiveness of GBL on a software engineering course, researchers must consider using the questionnaires and models designed for this specific purpose such as FDF [40], EGBL [32], MEEGA [128], and MEEGA+ [109] to add more formality to the research design.
- Finally, the combination of descriptive and inferential statistics is highly recommended to truly demonstrate the effectiveness of GBL when supporting software engineering education at the undergraduate level.

In addition, our SLR also provides other relevant information that may be taken into account when using/evaluating the GBL approach to design a game that promotes soft skills on undergraduate software courses. These findings can be summarized as follows. The collected studies provided positive evidence that the use of games on undergraduate courses effectively improves the learning process in software engineering areas. However, the selection of proper game mechanics is still a real challenge. A proper selection process would encourage game-like student behavior and promote students' game thinking and engagement, thus motivating them to reach their goals and develop their soft skills during the learning process. For example, resource management was highly considered when designing games for improving students' knowledge on project management [10,81], whereas other mechanics such as victory points, action points, or duels were used when designing games focused on improving the students' skills on programming [42,59,73,117,139]. Furthermore, missions, quizzes, and leaderboards were game mechanics used in the design of games focused on supporting the learning of requirements engineering topics [48,49,52,110]. Therefore, it can be concluded that not all of the game mechanics are effective for teaching all the knowledge areas in software engineering. On the contrary, it was also observed that the size of the samples used in the game evaluation was not homogeneous. Approximately, 22.9% of the evaluations were conducted with small samples, ranging from 8 to 20 students. Thirty-five studies (~36.5%) were performed with samples ranging from 21 to 40 students. Several studies (~28%) reported sample sizes ranging from 41 to 100 participants, whereas a small number of evaluations (~11.5%) were conducted with more than 100 students. Moreover, the time elapsed on conducting the game sessions was fairly limited in some studies (ranging from 15 to 30 min) and more extensive in others (120 min). Therefore, we believe that these issues make it impossible to conclude that the GBL approach always generates positive effects on the acquisition of soft skills.

6 | THREATS TO VALIDITY

There were certain threats to validity that could affect the results of our SLR, with bias present in publications and inaccurate extraction probably the main examples. These are detailed as follows:

- Searches: The main studies were selected by using the search strategy described in Section 3.3, including selection criteria and quality criteria (Section 3.4.1), as well as the selection procedure (Section 3.4.2). The terms related to our RQs were used to identify studies relevant for this review. However, there is still the possibility of missing important studies because not all studies can be extracted using the terms related to the

RQs in their titles, abstracts, or keywords. As we extended our searches to include studies cited in the included studies, as well as key conferences, individual journals, and key authors, we are confident that the vast majority of key papers have been included in the SLR. Moreover, with the aim of minimizing both threats, all the selected studies were re-evaluated by an independent researcher to identify true positives (i.e., studies where the title could be relevant, but the contents do not contain answers to any of the RQs).

- Data synthesis: There is also a possibility that studies have been missed which should have been included in the set of 96 from which data were extracted. Moreover, some studies may have satisfied our assessment criteria but either failed to report what they carried out or did not report their findings in sufficient detail for us to be confident that they met the necessary criteria. Similarly, we may have missed the reporting of a detail, and a paper that should have passed a criterion may have in fact not done so. Nevertheless, we strongly believe that these risks were mitigated by our two validation exercises to assess every selected study.
- Inaccuracy of extracted data: To reduce this inaccuracy, we carried out an independent evaluation of the selected studies (see Table 1). This evaluation was performed by the primary research group and an independent researcher who randomly analyzed a set of studies. Additionally, we entered into an inter-rater agreement to solve the existing discrepancies and to obtain similarities in the ordering of ratings executed as well as the independent evaluation.

7 | CONCLUSION AND FUTURE WORK

Undergraduate education on software engineering has been stuck, for many years, in a traditional non-productive approach usually supported by lecture-based courses and the implementation of practical exercises that are often decontextualized from reality. In addition, the lack of knowledge on the software industry's current needs is frequently reflected in the universities' curricula, resulting in the lack of confidence in many graduates when solving problems in real work environments. This situation, far from motivating students, has actually complicated the acquisition of technical skills and the development of soft skills in software engineering. Moreover, it has led to some students losing interest in learning other software engineering areas that are different from programming, because they think of them as boring and difficult to learn. In this regard, the GBL approach has proven to be effective for teaching and learning various software engineering topics, given its intrinsic characteristics making learning more attractive and motivating for students. It is true to say that GBL can be an appropriate approach to transmit software engineering knowledge to undergraduate students given its ludic nature, but it also can encourage the development of students' soft skills, vital for becoming a successful practitioner within the modern software industry.

We have, therefore, presented our experiences of applying an SLR with the aim of identifying the existing empirical evidence on the use of the GBL approach in teaching software engineering at the undergraduate level. Nevertheless, unlike previous SLRs, this new review intended to explore which soft skills can be promoted using the GBL approach and which game genres can help in achieving this goal. The collected studies provided enough evidence to confirm that undergraduate students not only learned or strengthened theoretical concepts related to a topic of software engineering but also developed cognitive and soft skills as the games provided the freedom to improvise and think of creative solutions. Considering all the collected information, programming is the software engineering area that has received the most attention over a period of almost twenty years from undergraduate researchers. It is obvious that students who enroll in software engineering-related programs first learn to program. However, the majority of these students often have many difficulties in understanding complex concepts related to programming paradigms, data structures, and/or the recursion concept. This may due to the traditional approach for teaching programming which is mainly focused on the syntax and features of a computer program, rather promoting a deeper understanding of programming constructs, abstract concepts, and code quality. One could conclude that this is logical if the research by Da Silva et al. [35] is taken into account stating that the use of games motivates and enhances the teaching of programming, increasing the attention spans in students and reducing the number of students leaving courses. Similarly, according to Marín et al. [83], the use of serious games has been a thoroughly investigated method for its effectiveness in providing novice programmers with interactive and entertaining experiences during the first year of their careers. In this context, just under half of the collected studies (~44.8%) designed and evaluated games focused on supporting education on diverse programming topics. On the contrary, an increasing trend was observed for supporting education in software engineering areas with more digital games than non-digital games. Although it is true that digital games can be more attractive and interesting for students, the information collected by the SLR provides evidence that the trend for using non-digital games in software

engineering courses is growing steadily. The development and use of digital games are unquestionable at the undergraduate level, but the capability of non-digital games should not be disregarded due to their unique attributes. For example, non-digital games can be used to support undergraduate courses involving social aspects such as learning aspects of project management with Scrum where face-to-face interactions are highly recommended. Simulation has been the most used genre when designing these digital games created to support the undergraduate software engineering courses, whereas card games have been the most popular non-digital game genre.

In addition, to differentiate our SLR from previous ones, it was the aim of this study to identify the soft skills that could be promoted by the games developed under a GBL approach. In other words, the authors of this paper tried to investigate the contribution of GBL in encouraging soft skills in undergraduate software engineering courses. With this objective in mind, it was possible to determine that, although the promotion of soft skills was not the studies' main objective, they did in fact occur to some extent. This study's findings provide encouragement in believing that the GBL approach not only has the potential to strengthen the technical skills required in any area of software engineering but also in promoting the development of soft skills among students which are increasingly required by the software industry. It is also worth mentioning that non-digital games tend to promote specific skills that are rarely explored with digital games, such as the development of interpersonal skills (ie, social interaction among students). The SLR also identified the methods used for evaluating the soft skills promoted by the application of a GBL approach. The findings provided evidence that the majority of evaluations are conducted in an ad hoc manner that is widely supported by the use of descriptive statistics. The collected information also revealed that many studies used small samples to carry out game evaluation within controlled environments.

However, despite the positive findings achieved with our SLR, we strongly believe that it is necessary to continue investigating the effects of the GBL approach on soft skills to truly determine the effectiveness of this approach on undergraduate software engineering courses. Finally, future work should focus on the development of a new SLR on the type of game elements (e.g., mechanics, dynamics, esthetics) that best facilitate the promotion of soft skills among undergraduate students. Additionally, we aim to use the evidence gathered in both SLRs to apply the GBL approach in designing a serious game for training students on the use of the ISO/IEC/IEEE 15989:2017—systems and software engineering—measurement process standard.

ORCID

Ivan Garcia http://orcid.org/0000-0002-7594-6410
Jose A. Calvo-Manzano http://orcid.org/0000-0002-2864-2203

REFERENCES

1. A. I. Abdul Jabbar and P. Felicia, *How game-based learning works and what it means for pupils, teachers, and classroom learning*, Design, motivation, and frameworks in game-based learning (W. H. Tan, ed.), Information Science Reference, Hershey, PA, 2019, pp. 1–29.
2. N. Adamo-Villani, T. Haley-Hermiz, and R. Cutler, *Using a serious game approach to teach "operator precedence" to introductory programming students*, Proc. 17th Int. Conf. Inf. Vis. IEEE Comput. Soc., 2013, pp. 523–526.
3. F. Ahmed et al., *Soft skills and software development: A reflection from the software industry*, Int. J. Inf. Process. Manag. **4** (2013), no. 3, 171–191.
4. F. Ahmed, L. Capretz, and P. Campbell, *Evaluating the demand for soft skills in software development*, IT Prof. **14** (2012), no. 1, 44–49.
5. A. Akbulut, C. Catal, and B. Yıldız, *On the effectiveness of virtual reality in the education of software engineering*, Comput. Appl. Eng. Educ. **26** (2018), no. 4, 918–927.
6. D. Arsenault, *Video game genre, evolution and innovation*, J. Comput. Game Cult. **3** (2009), no. 2, 149–176.
7. U. Aydan et al., *Teaching ISO/IEC 12207 software lifecycle processes: A serious game approach*, Comput. Standards Interfaces **54** (2017), no. 3, 129–138.
8. A. Baker, E. O. Navarro, and A. Van Der Hoek, *An experimental card game for teaching software engineering processes*, J. Syst. Softw. **75** (2005), no. 1-2, 3–16.
9. T. Barnes et al., *Game2Learn: Improving the motivation of CS1 students*, Proc. ACM 3rd Int. Con. Game Dev. Comput. Sci. Educ., 2008, pp. 1–5.
10. M. O. Barros et al., *Model-driven game development: Experience and model enhancements in software project management education*, Softw. Process: Improvement Pract. **11** (2006), no. 4, 411–421.
11. V. R. Basili, G. Caldiera, and H. D. Rombach, *Goal, question metric paradigm*, Encyclopedia of software engineering (J. J. Marciniak, ed.), Wiley-Interscience, Boston, MA, 1994, pp. 528–532.
12. P. E. Battistella and C. G. von Wangenheim, *Games for teaching computing in higher education—A systematic review*, IEEE Technol. Eng. Educ. **1** (2016), no. 3, 8–30.
13. S. Beecham et al., *Motivation in software engineering: A systematic review*, Inf. Softw. Technol. **50** (2008), no. 9–10, 860–878.
14. F. B. V. Benitti, *Software engineering role-playing game: An interactive game for software engineering education*, Int. J. Adv. Res. Comput. Sci. **2** (2011), no. 2, 6–9.
15. F. B. V. Benitti and L. Sommariva, *Evaluation of a game used to teach usability to undergraduate students in computer science*, J. Usability Stud. **11** (2015), no. 1, 21–39.
16. I. Bezakova, J. E. Heliotis, and S. P. Strout, *Board game strategies in introductory computer science*, Proc. 44th ACM Tech. Symp. Comput. Sci. Educ., 2013, pp. 17–22.
17. N. Binti-Yusof and N. Binti-Che, *The effect of gamified assessment on student's achievement, motivation and engagement in database design course*, J. Tech. Vocational Educ. **4** (2019), no. 3, 78–91.

18. P. Bourque, and R. E. Fairley, *Guide to the software engineering body of knowledge (SWEBOK®) version 3.0*, IEEE Computer Society Press, Washington, DC, 2014.
19. L. F. Braghirolli et al., *Benefits of educational games as an introductory activity in industrial engineering education*, Comput. Human. Behav. **58** (2016), 315–324.
20. R. M. Branch, *Instructional design: The ADDIE approach*, Springer, Berlin, Germany, 2010.
21. A. Calderón, M. Ruiz, and R. V. O'Connor, *Coverage of ISO/IEC 29110 project management process of basic profile by a serious game*, Communications in Computer and Information Science (J. Stolfa, S. Stolfa, R. V. O'Connor, and R. Messnarz, eds.), Springer, Berlin, Germany, 2017, pp. 111–122.
22. A. Calderón, M. Ruiz, and R. V. O'Connor, *A multivocal literature review on serious games for software process standards education*, Comput. Standards Interfaces **57** (2018), 36–48.
23. A. Calderón, M. Ruiz, and R. V. O'Connor, *A serious game to support the ISO 21500 standard education in the context of software project management*, Comput. Standards Interfaces **60** (2018), 80–92.
24. L. Capretz and F. Ahmed, *A call to promote soft skills in software engineering*, Psychol. Cogn. Sci. **4** (2018), no. 1, 1–3.
25. L. Capretz, D. Varona, and A. Raza, *Influence of personality types in software tasks choices*, Comput. Hum. Behav. **52** (2015), 373–378.
26. C. Caulfield et al., *A systematic survey of games used for software engineering education*, Modern Appl. Sci. **5** (2011), no. 6, 28–43.
27. A. Chaffin et al., *Experimental evaluation of teaching recursion in a video game*, 2009 ACM SIGGRAPH Symp. Video Games, 2009, pp. 79-86.
28. W. F. Chen et al., *Work in progress—A game-based learning system for software engineering education*, Proc. 38th Annu. Front. Educ. Conf., IEEE Comput. Soc., 2008.
29. T. M. Connolly et al., *A systematic literature review of empirical evidence on computer games and serious games*, Comput. Educ. **59** (2012), no. 2, 661–686.
30. T. M. Connolly, M. Stansfield, and T. Hainey, *An application of games-based learning within software engineering*, Br. J. Educ. Technol. **38** (2007), no. 3, 416–428.
31. T. M. Connolly, M. Stansfield, and T. Hainey, *Using games-based learning to teach software engineering*, Web information systems and technologies (J. Filipe and J. Cordeiro, eds.), Springer, Berlin, Germany, 2007, pp. 304–313.
32. T. M. Connolly, M. Stansfield, and T. Hainey, *Towards the development of a games-based learning evaluation framework*, Games-based learning advancement for multisensory human computer interfaces: Techniques and effective practices (T. M. Connolly, M. H. Stansfield, and E. Boyle, eds.), Idea-Group Publishing, Hershey, PA, 2009, pp. 251–273.
33. T. M. Connolly, M. Stansfield, and E. McLellan, *Using an online games-based learning approach to teach database design concepts*, Electron. J. E-Learn. **4** (2006), no. 1, 103–110.
34. J. M. R. Corral et al., *A game-based approach to the teaching of object-oriented programming languages*, Comput. Educ. **73** (2014), 83–92.
35. T. R. Da Silva, T. Medeiros, and E. H. S. Da Aranha, *The use of games on the teaching on programming: A systematic review*, Workshop Exp. Softw. Eng. (ESELAW 2015), 2015, pp. 474–487.
36. K. Daungcharone, P. Panjaburee, and K. Thongkoo, *Using digital game as compiler to motivate C programming language learning in higher education*, 6th IIAI Int. Congr. Adv. Appl. Inform., IEEE Comput. Soc., 2017, pp. 533–538.
37. F. D. Davis, R. P. Bagozzi, and P. R. Warshaw, *User acceptance of computer technology: A comparison of two theoretical model*, Manag. Sci. **35** (1989), no. 8, 903–1028.
38. S. De Ascaniis et al., *A lifelike experience to train user requirements elicitation skills*, Design, user experience, and usability: Understanding users and contexts (A. Marcus, W. Wang, eds.), Springer, Berlin, Germany, 2017, pp. 219–237.
39. G. S. De Azevedo, V. T. Sarinho, and F. de Santana, *Metrics war: A board game proposal for teaching software metrics and quality attributes*, Proc. XVIII Simpósio Brasileiro de Games e Entretenimento Digit. (SBGames 2019), 2019, pp. 599–602.
40. S. De Freitas and M. Oliver, *How can exploratory learning with games and simulations within the curriculum be most effectively evaluated?* Comput. Educ. **46** (2006), no. 3, 249–264.
41. A. D. De Souza et al., *An experience of using a board serious virtual game for teaching the SCRUM framework*, Information technology—New generations. Advances in intelligent systems and computing In S. Latifi, ed.), Springer, Berlin, Germany, 2018, pp. 213–218.
42. M. Eagle and T. Barnes, *Wu's castle: Teaching arrays and loops in a game*, ACM SIGCSE Bull. **40** (2008), no. 3, 245–249.
43. E. M. Edirisinghe, *Teaching students to identify common programming errors using a game*, Proc. 9th ACM SIGITE Conf. Inf. Technol. Educ., 2008, pp. 95–98.
44. J. L. Fleiss, *Measuring nominal scale agreement among many raters*, Psychol. Bull. **76** (1971), no. 5, 378–382.
45. R. Florea and V. Stray, *Software tester, we want to hire you! An analysis of the demand for soft skills*, Agile processes in software engineering and extreme programming (J. Garbajosa, X. Wang, and A. Aguiar, eds.), Springer, Cham, Switzerland, 2018, pp. 54–67.
46. F. Fu, R. Su, and S. Yu, *EGameFlow: A scale to measure learners' enjoyment of e-learning games*, Comput. Educ. **52** (2009), no. 1, 101–112.
47. L. Ganesh, *Board game as a tool to teach software engineering concept—Technical debt*, Proc. 6th Int. Conf. Technol. Educ., IEEE Comput. Soc., 2014, pp. 44–47.
48. I. Garcia et al., *Experiences of using a game for improving learning in software requirements elicitation*, Comput. Appl. Eng. Educ. **27** (2019), no. 1, 249–265.
49. I. Garcia et al., *A serious game for teaching the fundamentals of ISO/IEC/IEEE 29148 systems and software engineering—lifecycle processes—requirements engineering at undergraduate level*, Comput. Standards Interfaces **67** (2020), 103377.
50. S. M. Glynn et al., *Science motivation questionnaire II: Validation with science majors and nonscience majors*, J. Res. Sci. Teach. **48** (2011), no. 10, 1159–1176.
51. U. Gulec et al., *CENGO: A web-based serious game to increase the programming knowledge levels of computer engineering students*, Systems, software and services process improvement (A. Walker, R. V. O'Connor, R. Messnarz, eds.), Springer, Berlin, Germany, 2019, pp. 237–248.

52. T. Hainey et al., *Evaluation of a game to teach requirements collection and analysis in software engineering at tertiary education level*, Comput. Educ. **56** (2011), no. 1, 21–35.
53. L. Hakulinen, *Using serious games in computer science education*, Proc. ACM 11th Koli Calling Int. Conf. Comput. Educ. Res., 2011, pp. 83–88.
54. D. Hamlet and J. Maybee, The engineering of software: A technical guide for the individual, 1st ed., Addison-Wesley, Boston, MA, 2000.
55. V. T. Heikkilä, M. Paasivaara, and C. Lassenius, *Teaching university students Kanban with a collaborative board game*, Proc. ACM 38th Int. Conf. Softw. Eng. Companion, 2016, pp. 471–480.
56. R. Hijón-Neira et al., *Improving students learning programming skills with ProGames programming through games system*, Human-computer interaction (P. Kotzé, G. Marsden, G. Lindgaard, J. Wesson, M. Winckler, eds.), Springer, Berlin, Germany, 2013, pp. 579–586.
57. R. Ibrahim et al., *Student's opinions on online educational games for learning programming introductory*, Int. J. Adv. Comput. Sci. Appl. **9** (2018), no. 6, 332–340.
58. M. Itani and I. Srour, *Engineering students' perceptions of soft skills, industry expectations, and career aspirations*, J. Prof. Issues Eng. Educ. Pract. **142** (2016), no. 1, 1–12.
59. H. C. Jiau, J. C. Chen, and K. F. Ssu, *Enhancing self-motivation in learning programming using game-based simulation and metrics*, IEEE Trans. Educ. **52** (2009), no. 4, 555–562.
60. K. Jones, L. N. K. Leonard, and G. Lang, *Desired skills for entry level IS positions: Identification and assessment*, J. Comput. Inf. Syst. **58** (2018), no. 3, 214–220.
61. E. Kalliamvakou et al., *What makes a great manager of software engineers?* IEEE Trans. Softw. Eng. **45** (2017), no. 1, 87–106.
62. N. Kamin, *Soft skills revolution: A guide for connecting with compassion for trainers, teams, and leaders*, John Wiley & Sons, San Francisco, CA, 2013.
63. K. Kasemsap, *The fundamentals of game-based learning*, Handbook of research on instructional systems and educational technology (T. Kidd and L. R. Morris, eds.), Information Science Reference, Hershey, PA, 2017, pp. 174–185.
64. I. Kazanidis, A. Tsinakos, and C. Lytridis, *Teaching mobile programming using augmented reality and collaborative game based learning*, Advances in intelligent systems and computing (M. Auer and T. Tsiatsos, eds.), Springer, Berlin, Germany, 2017, pp. 850–859.
65. C. Kazimoglu et al., *Learning programming at the computational thinking level via digital game-play*, Procedia Comput. Sci. **9** (2012), 522–531.
66. B. A. Kitchenham, *Procedures for performing systematic reviews*, Joint Technical Report Software Engineering Group, Department of Computer Science Keele University, Keele, UK and Empirical Software Engineering, National ICT Australia, Sydney, Australia. 2004.
67. B. A. Kitchenham and S. Charters, *Guidelines for performing systematic literature reviews in software engineering* EBSE Technical Report EBSE-2007-01, Software Engineering Group, School of Computers Science and Mathematics, Keele University, Keele, UK and Department of Computer Science, University of Durham, Durham, UK.
68. D. Kletenik et al., *A serious game to teach computing concepts*, Communications in computer and information science (C. Stephanidis, ed.), Springer, Berlin, Germany, 2017, pp. 146–153.
69. L. Konzack, *Video game genres*, Encyclopedia of information science and technology (M. Khosrow-Pour, ed.), 3rd ed., Information Science Reference, Hershey, PA, 2015, pp. 3070–3077.
70. M. Kosa et al., *Software engineering education and games: A systematic literature review*, J. Univ. Comput. **22** (2016), no. 12, 1558–1574.
71. M. Kuhrmann, D. M. Fernández, and M. Daneva, *On the pragmatic design of literature studies in software engineering: An experience-based guideline*, Empirical Softw. Eng **22** (2017), no. 6, 2852–2891.
72. S. Kurkovsky, *Teaching software engineering with LEGO serious play*, Proc. 2015 ACM Conf. Innov. Technol. Comput. Sci. Educ., 2015, pp. 213–218.
73. M. J. Lee, A. J. Ko, and I. Kwan, *In-game assessments increase novice programmers' engagement and level completion speed*, Proc. ACM 9th Annu. Int. ACM Conf. Int. Comput. Educ. Res., 2013, pp. 153–160.
74. E. Lee et al., *A structured approach to teaching recursion using cargo-bot*, Proc. ACM 10th Annu. Conf. Int. Comput. Educ. Res., 2014, pp. 59–66.
75. P. L. Li, A. J. Ko, and J. Zhu, *What makes a great software engineer?* Proc. 37th Int. Conf. Softw. Eng., IEEE Comput. Soc., 2015, pp. 700–710.
76. R. S. N. Lindberg, T. H. Laine, and L. Haaranen, *Gamifying programming education in K-12: A review of programming curricula in seven countries and programming games*, Br. J. Educ. Technol. **50** (2018), no. 4, 1979–1995.
77. T. Y. Liu, *Using educational games and simulation software in a computer science course: Learning achievements and student flow experiences*, Interact. Learn. Environ. **24** (2016), no. 4, 724–744.
78. C. C. Liu, Y. B. Cheng, and C. W. Huang, *The effect of simulation games on the learning of computational problem solving*, Comput. Educ. **57** (2011), no. 3, 1907–1918.
79. J. Livovský and J. Porubän, *Learning object-oriented paradigm by playing computer games: Concepts first approach*, Open Comput. Sci. **4** (2014), no. 3, 171–182.
80. R. W. Lui, P. T. Y. Lee, and K. Y. Fung, *Inquiry based learning with Kanban game*, Proc. 11th In. Conf. Eng. Educ., IEEE Comput. Soc., 2019, pp. 108–112.
81. R. W. Lui, P. T. Y. Lee, and V. T. Ng, *Design and evaluation of PMS: A computerized simulation game for software project management*, Comput. Games J. **4** (2015), no. 1–2, 101–121.
82. C. Malliarakis, M. Satratzemi, and S. Xinogalos, *CMX: The effects of an educational MMORPG on learning and teaching computer programming*, IEEE Trans. Learn. Technol. **10** (2017), no. 2, 219–235.
83. B. Marín et al., *An empirical investigation on the benefits of gamification in programming courses*, ACM Trans. Comput. Educ. **19** (2018), no. 1, 1–22.
84. B. Marín, F. Larenas, and G. Giachetti, *Learning conceptual modeling design through the Classutopia serious game*, Int. J. Softw. Eng. Knowl. Eng. **28** (2018), no. 11–12, 1679–1699.
85. B. R. Marques, S. P. Levitt, and K. J. Nixon, *Software visualisation through video games*, ACM Proc. South African Inst. Comput. Sci. Inf. Technol. Conf., 2012, pp. 206–215.

86. J. Martí-Parreño, A. Galbis-Córdova, and M. J. Miquel-Romero, *Students' attitude towards the use of educational video games to develop competencies*, Comput. Hum. Behav. **81** (2019), 366–377.
87. R. Mathew, S. I. Malik, and R. M. Tawafak, *Teaching problem solving skills using an educational game in a computer programming course*, Inform. Educ. **18** (2019), no. 2, 359–373.
88. A. Mathrani, C. Shelly, and A. Ponder-Sutton, *PlayIT: Game based learning approach for teaching programming concepts*, J. Educ. Technol. Soc. **19** (2016), no. 2, 5–17.
89. G. Matturro, F. Raschetti, and C. Fontán, *Soft skills in software development teams: A survey of the points of view of team leaders and team members*, Proc. 8th Int. Workshop Cooperative Hum. Aspects Softw. Eng., IEEE Comput. Soc., 2015, pp. 101–104.
90. G. Matturro, F. Raschetti, and C. Fontán, *A systematic mapping study on soft skills in software engineering*, J. Univ. Comput. Sci. **25** (2019), no. 1, 16–41.
91. G. V. Maturana-Gonzalez et al., *Software project pool: A game for learning software project bidding*, Proc. 46th Conf. Dev. Bus. Simul. Experiential Learn. (ABSEL), San Diego, CA, 2019, pp. 304–310.
92. M. J. Mayo, *Games for science and engineering education*, Commun. ACM **50** (2007), no. 7, 31–35.
93. A. L. Mesquida and A. Mas, *Experiences on the use of a game for improving learning and assessing knowledge*, Comput. Appl. Eng. Educ. **26** (2018), no. 6, 2058–2070.
94. M. A. Miljanovic and J. S. Bradbury, *RoboBUG: A serious game for learning debugging techniques*, Proc. ACM Conf. Int. Comput. Educ. Res., 2017, pp. 93–100.
95. P. Mohammed and P. Mohan, *Student attitudes towards using culturally oriented educational games to improve programming proficiency: An exploratory study*, Learning by playing. game-based education system design and development (M. Chang, R. Kuo, C. G. D. Kinshuk, and M. Hirose, eds.), Springer, Berlin, Germany, 2009, pp. 196–207.
96. J. S. Molléri, J. Gonzalez-Huerta, and K. Henningsson, *A legacy game for project management in software engineering courses*, Proc. ACM 3rd Eur. Conf. Softw. Eng. Educ., 2018, pp. 72–76.
97. G. G. Moreira and A. B. dos Santos Marques, *Evaluating the students' experience with the Scrum card game: An experience report in a software engineering course*, Proc. ACM 17th Braz. Symp. Softw. Qual., 2018, pp. 344–353.
98. J. Moreno, *Digital competition game to improve programming skills*, J. Educ. Technol. Soc. **15** (2012), no. 3, 288–297.
99. M. Muratet et al., *Serious game and students' learning motivation: Effect of context using prog&play*, Intelligent tutoring systems (S. A. Cerri, W. J. Clancey, G. Papadourakis, K. Panourgia, eds.), Springer, Berlin, Heidelberg, 2012, pp. 123–128.
100. N. Naik, *Non-digital game-based learning in higher education: A teacher's perspective*, Proc. 9th Eur. Conf. Games Based Learn., Acad. Conf. Publ. Int. Ltd. Read., 2015, pp. 402–407.
101. N. Naik and P. Jenkins, *Relax, it's a game: Utilising gamification in learning agile Scrum software development*, Proc. IEEE Conf. Games, IEEE Comput. Soc., 2019, pp. 1–4.
102. E. O. Navarro and A. Van Der Hoek, *Comprehensive evaluation of an educational software engineering simulation environment*, Proc. 20th Conf. Softw. Eng. Educ. Training, IEEE Comput. Soc., 2007, pp. 195–202.
103. E. Nunohiro et al., *Development of game-based learning features in programming learning support system*, Artif Life Robot. **17** (2013), no. 3-4, 373–377.
104. S. Ouhbi et al., *Requirements engineering education: A systematic mapping study*, Requirements Eng. **20** (2015), no. 2, 119–138.
105. D. C. C. Peixoto, R. F. Resende, and C. I. P. S. Pádua, *Evaluating software engineering simulation games: The UGALCO framework*, Proc. 44th ASEE/IEEE Front. Educ. Conf., IEEE Comput. Soc., 2014 pp. 1–9.
106. D. C. C. Peixoto, R. F. Resende, and C. I. P. S. Pádua, *An experience with software engineering education using a software process improvement game*, Higher education for all. From challenges to novel technology-enhanced solutions — Communications in computer and information science (A. Cristea, I. Bittencourt, and F. Lima, eds.), Springer, Berlin, Germany, 2018, pp. 157–173.
107. R. M. Pellegrini, C. E. S. da Silva, and A. D. de Souza, *Teaching communication management in software projects through serious educational games*, Information technology—New generations. Advances in intelligent systems and computing (S. Latifi, ed.), Springer, Berlin, Germany, 2018, pp. 201–205.
108. G. Petri and C. G. von Wangenheim, *How games for computing science education are evaluated? A systematic literature review*, Comput. Educ. **107** (2017), 68–90.
109. G. Petri, C. G. von Wangenheim, and A. F. Borgatto, *MEEGA +: An evolution of a model for the evaluation of educational games*, Technical Report INCoD/GQS.03.2016.E, Version 1.0, Brazilian Institute for Digital Convergence, Brazil, 2016.
110. W. Prasetya et al., *Having fun in learning formal specifications*, Proc. IEEE/ACM 41st Int. Conf. Softw. Eng.: Softw. Eng. Educ. Training, IEEE Comput. Soc., 2019, pp. 192–196.
111. M. Prensky, *Digital game-based learning*, McGraw-Hill, New York, NY, 2001.
112. H. Pötter et al., *InspectorX: A game for software inspection training and learning*, Proc. 27th Conf. Softw. Eng. Educ. Training, IEEE Comput. Soc., 2014, pp. 55–64.
113. Princeton University. *About WordNet, Word Net 3.1*, Princeton University, Princeton, NJ, available at http://wordnetprinceton.edu
114. M. Qian and K. R. Clark, Game-based learning and 21*st* century skills: A review of recent research, Comput. Hum. Behav. **63** (2016), 50–58.
115. M. H. A. Rahman et al., *Gamification elements and their impacts on teaching and learning—A review*, Int. J. Multimedia Appl. **10** (2018), no. 6, 37–46.
116. A. E. Rais, S. Sulaiman, and S. M. Syed-Mohamad, *Game-based approach and its feasibility to support the learning of object-oriented concepts and programming*, Proc. 5th Malaysian Conf. Softw. Eng., IEEE Comput. Soc., 2011, pp. 307–312.
117. S. Rajeev and S. Sharma, *Motivational game-theme based instructional module for teaching binary tree and linked list*, Proc. 28th Int. Conf. Softw. Eng. Data Eng., San Diego, CA, 2019, pp. 31–40.
118. R. Ramle, S. S. Nathan, and M. Berahim, *Digital game based learning of stack data structure using question prompts*, Int. J. Interact. Mobile Technol. **13** (2019), no. 7, 90–102.
119. P. Rodrigues, M. Souza, and E. Figueiredo, *Games and gamification in software engineering education: A survey with*

educators, Proc. 48th Annu. Front. Educ. Conf., IEEE Comput. Soc., 2018, pp. 1–9.

120. G. Rodriguez, Á. Soria, and M. Campo, *Virtual scrum: A teaching aid to introduce undergraduate software engineering students to scrum*, Comput. Appl. Eng. Educ. **23** (2015), no. 1, 147–156.
121. A. Rusu et al., *Employing software maintenance techniques via a tower-defense serious computer game*, Edutainment technologies. educational games and virtual reality/augmented reality applications (M. Chang, W. Y. Hwang, M. P. Chen, and W. Müller, eds.), Springer, Berlin, Germany, 2011, pp. 176–184.
122. A. Rusu, R. Russell, and R. Cocco, *Simulating the software engineering interview process using a decision-based serious computer game*, Proc. 16th Int. Conf. Comput. Games, IEEE Comput. Soc., 2011, pp. 235–239.
123. A. Rusu et al., *Introducing object oriented design patterns through a puzzle-based serious computer game*, Proc. Front. Educ. Conf., IEEE Comput. Soc., 2011, pp. F1H-1–F1H-6.
124. A. Sajana, K. Bijlani, and R. Jayakrishnan, *An interactive serious game via visualization of real life scenarios to learn programming concepts*, Proc. 6th Int. Conf. Comput., Commun. Netw. Technol., IEEE Comput. Soc., 2015, pp. 1–8.
125. R. F. Sampaio and M. C. Mancini, *Systematic review studies: A guide for careful synthesis of scientific evidence*, Braz. J. Phys. Ther. **11** (2007), no. 1, 83–89.
126. S. Santos et al., *Risking: A game for teaching risk management in software projects*, Proc. ACM XVIII Bra. Symp. Softw. Qual., 2019, pp. 188–197.
127. V. T. Sarinho, V. O. Gomes, and W. T. Sarinho, ERQuiz: A multiplayer multiplatform instant messaging game for the competitive assessment of requirements engineering knowledge, Proc. XVIII Simpósio Brasileiro de Games e Entretenimento Digit. (SBC), 2019, pp. 591–594.
128. R. Savi, C. G. von Wangenheim, and A. F. Borgatto, *A model for the evaluation of educational games for teaching software engineering*, Proc. 25th Braz. Symp. Softw. Eng., IEEE Comput. Soc., 2011, pp. 194-203.
129. J. L. K. Seng and E. Edirisinghe, *Teaching computer science using second life as a learning environment*, ICT, Providing Choices Learners Learn, Australas. Soc. Comput. Learn. Tertiary Educ. (2007), 583–586.
130. Á. Serrano-Laguna et al., *Building a scalable game engine to teach computer science languages*, IEEE Revista Iberoamericana de Tecnologías del Aprendizaje **10** (2015), no. 4, 253–261.
131. S. Smith and S. Chan, *Collaborative and competitive video games for teaching computing in higher education*, J. Sci. Educ. Technol. **26** (2017), no. 4, 438–457.
132. R. Smith and O. Gotel, *Gameplay to introduce and reinforce requirements engineering practices*, Proc. 16th IEEE Int. Requirements Eng. Conf., IEEE Comput. Soc., 2008, pp. 95–104.
133. M. Soflano, T. M. Connolly, and T. Hainey, *An application of adaptive games-based learning based on learning style to teach SQL*, Comput. Educ. **86** (2015), 192–211.
134. M. T. Soo and H. Aris, *Game-based learning in requirements engineering: An overview*, Proc. IEEE Conf. e-Learn., e-Manag. e-Serv., IEEE Comput. Soc., 2018, pp. 46–51.
135. M. J. Sousa and A. Rocha, *Leadership styles and skills developed through game-based learning*, J. Bus. Res. **94** (2019), 360–366.
136. M. Souza et al., *Games for learning: Bridging game-related education methods to software engineering knowledge areas*, Proc. 39th Int. Conf. Softw. Eng., Softw. Eng. Educ. Track, IEEE Comput. Soc., 2017, pp. 170-179.
137. M. Souza et al., *A systematic mapping study on game-related methods for software engineering education*, Inf. Softw. Technol **95** (2018), 201–218.
138. M. Stevens and R. Norman, *Industry expectations of soft skills in IT graduates: A regional survey*, Proc. ACM Australas. Comput. Sci. Week Multiconf. 2016, pp. 1–9.
139. J. Stigall and S. Sharma, *Usability and learning effectiveness of game-themed instructional (GTI) module for teaching stacks and queues*, Proc. SoutheastCon, IEEE Comput. Soc., 2018, pp. 1–6.
140. C. H. Su and C. H. Cheng, *3D game-based learning system for improving learning achievement in software engineering curriculum*, Turkish Online J. Educ. Technol. **12** (2013), no. 2, 1–12.
141. E. Suescún-Monsalve, J. C. S. do Prado Leite, and V. M. B. Werneck, *Transparently teaching in the context of game-based learning: The case of SimulES-W*, Proc. IEEE/ACM 37th Int. Conf. Softw. Eng., IEEE Comput. Soc., 2015, pp. 343–352.
142. E. Suescún-Monsalve et al., *SimulES-W: A collaborative game to improve software engineering teaching*, Computación y Sistemas **22** (2018), no. 3, 953–983.
143. A. Sánchez et al., *Incorporating computing professionals' know-how: Differences between assessment by students, academics, and professionals experts*, ACM Trans. Comput. Educ. **19** (2019), no. 3, 1–18.
144. M. L. Sánchez-Gordón et al., *Bridging the gap between SPI and SMEs in educational settings: A learning tool supporting ISO/IEC 29110*, Communications in computer and information science (C. Kreiner, R. V. O'Connor, A. Poth, and R. Messnarz, eds.), Springer, Berlin, Germany, 2016, pp. 3–14.
145. M. L. Sánchez-Gordón et al., *A learning tool for the ISO/IEC 29110 standard: Understanding the project management of basic profile*, Communications in computer and information science (P. Clarke, R. V. O'Connor, T. Rout, A. Dorling, eds.), Springer, Berlin, Germany, 2016, pp. 270–283.
146. D. Toth and M. Kayler, *Integrating role-playing games into computer science courses as a pedagogical tool*, Proc. 46th ACM Tech. Symp. Comput. Sci. Educ., 2015, pp. 386-391.
147. T. Uiphanit et al., *Packet warriors: An academic mobile action game for promoting OSI model concepts to learners*, Int. J. Interact. Mobile Technol. **13** (2019), no. 6, 41–51.
148. A. Vizcaíno et al., *Evaluating GSD-aware: A serious game for discovering global software development challenges*, ACM Trans. Comput. Educ. **19** (2019), no. 2, 1–23.
149. C. G. von Wangenheim, R. Savi, and A. F. Borgatto, *DELIVER!—An educational game for teaching earned value management in computing courses*, Inf. Softw. Technol. **54** (2012), no. 3, 286–298.
150. C. G. von Wangenheim, R. Savi, and A. F. Borgatto, *SCRUMIA—An educational game for teaching SCRUM in computing courses*, J. Syst. Softw. **86** (2013), no. 10, 2675–2687.
151. D. Watson, L. A. Clark, and A. Tellegen, *Development and validation of brief measures of positive and negative affect: The PANAS scales*, J. Pers. Soc. Psychol. **54** (1988), no. 6, 1063–1070.

152. C. Wohlin, M. Höst, and K. Henningsson, *Empirical research methods in software engineering methods in software engineering*, Empirical research methods and studies in software engineering (R. Conradi and A. I. Wang, eds.), Springer, Berlin, Germany, 2003, pp. 7–23.
153. C. Wohlin et al., *Experimentation in software engineering*, Springer, Berlin, Germany, 2012.
154. Y. S. Wong, M. Y. M. Hayati, and W. H. Tan, *A propriety game-based learning game as learning tool to learn object-oriented programming paradigm*, Serious games (T. Marsh, M. Ma, M. Oliveira, H. J. Baalsrud, and S. Göbel, eds.), Springer, Berlin, Germany, 2016, pp. 42–54.
155. Y. S. Wong and M. H. M. Yatim, *A propriety multiplatform game-based learning game to learn object-oriented programming*, Proc. 7th Int. Congr. Adv. Appl. Inform., IEEE Comput. Soc., 2018, pp. 278–283.
156. M. Xenos and V. Velli, *A serious game for introducing software engineering ethics to university students*, The challenges of the digital transformation in education, advances in intelligent systems and computing (M. Auer and T. Tsiatsos, eds.), Springer, Berlin, Germany, 2020, pp. 579–588.
157. L. Xis, A. N. Antle, and N. Motamedi, *Are tangibles more fun? Comparing children's enjoyment and engagement using physical, graphical and tangible user interfaces*, Proc. ACM 2nd Int. Conf. Tangible Embedded Interact., 2008, pp. 191–198.
158. E. Ye, C. Liu, and J. A. Polack-Wahl, *Enhancing software engineering education using teaching aids in 3-D online virtual worlds*, Proc. 37th Annu. Front. Educ. Conf.–Glob. Eng., Knowl. Without Borders, Opportunities Without Passports, IEEE Comput. Soc., 2007.
159. M. Yilmaz et al., *An examination of personality traits and how they impact on software development teams*, Inf. Softw. Technol. **86** (2017), 101–122.
160. J. Zhang et al., *Learning and practicing decision structures in a game*, J. Comput. Sci. Coll. **29** (2014), no. 4, 60–67.
161. D. Zhao and G. M. Muntean, *The Newton project warehouse game: A variable and data type serious game for C programming courses*, Proc. 10th Int. Conf. Educ. New Learn. Technol. (IATED Acad.), 2018, pp. 3709–3714.
162. D. Zhao, C. Muntean, and G. Muntean, *The restaurant game: A Newton project serious game for C programming courses*, Soc. Inf. Technol. Teacher Educ. Int. Conf., Assoc. Adv. Comput. Educ., 2019, pp. 1867–1874.
163. J. Zhu et al., *Programming in game space: How to represent parallel programming concepts in an educational game*, Proc. ACM 14th Int. Conf. Found. Digit. Games, 2019, pp. 1–10.

AUTHOR BIOGRAPHIES

Ivan Garcia holds a PhD degree in software and systems from the "Universidad Politécnica de Madrid" (Technical University of Madrid), Spain and is a full-time Professor at the "División de Estudios de Posgrado" (Post-Graduate Studies Division) in the "Universidad Tecnológica de la Mixteca" (Technological University of the Mixteca), Mexico. He is the author of international papers on software engineering and, more specifically, software process improvement. He has participated in more than 15 software project management and software process improvement projects in the Mexican software industry. He was an active participant of the "Research Chair for Software Process Improvement for Spain and the Latin American Region." In Latin America, he has participated in several institutional research projects in industry (IBM Global Services) and in academia. Finally, he was a member of the team that translated CMMI-DEV v1.2 and CMMI-DEV v1.3 to Spanish.

Carla Pacheco holds the PhD degree in computational languages and software engineering from the "Universidad Politécnica de Madrid" (Technical University of Madrid), Spain and is a full-time Professor at the "División de Estudios de Posgrado" (Post-Graduate Studies Division) in the "Universidad Tecnológica de la Mixteca" (Technological University of the Mixteca), Mexico. She is also author of international papers related to software engineering and, more specifically, requirements engineering. She has participated in several Mexican projects related to software requirements as a part of process to improve the Mexican software industry.

Francisco Méndez holds a MS degree in applied computing technologies from the "Universidad Tecnológica de la Mixteca" (Technological University of the Mixteca), Mexico. His research interests focus on the development of digital tools to improve software engineering education through playful and gamified approaches.

Jose A. Calvo-Manzano holds a PhD degree in computer science. He is an Assistant Professor at the "Escuela Técnica Superior de Ingenieros Informáticos" (School of Computer Engineering) in the "Universidad Politécnica de Madrid" (Technical University of Madrid), Spain, where he teaches in the area of software engineering, specifically in the domain of software process management and improvement. He has participated in more than 20 research projects (European, Spanish Public Administration as well as with private enterprises). He is co-author of more than 80 international journal papers and more than 100 conference papers. He is co-author

of more than 15 books about software process improvement and software engineering topics as well. He was the Head of the "Research Chair for Software Process Improvement for Spain and the Latin American Region," where he was a member of the team that translated CMMI-DEV v1.2 and v1.3 into Spanish. He also holds the ITIL® v2 and v3 Foundation, and CMDB Certifications.

How to cite this article: Garcia I, Pacheco C, Méndez F, Calvo-Manzano JA. The effects of game-based learning in the acquisition of "soft skills" on undergraduate software engineering courses: A systematic literature review. *Comput Appl Eng Educ*. 2020;28:1327–1354. https://doi.org/10.1002/cae.22304