Journal of Software Engineering Research and Development, 2020, 6:1, doi: 10.5753/jserd.2020.738

Is It Worth Using Gamification on Software Testing Education? An Extended Experience Report in the Context of Undergraduate Students

Gabriela Martins de Jesus [**Federal University of São Carlos** | ***gabriela.jesus@ufscar.br***]
Fabiano Cutigi Ferrari [**Federal University of São Carlos** | ***fcferrari@ufscar.br***]
Leo Natan Paschoal [**University of São Paulo** | ***paschoalln@usp.br***]
Simone do Rocio Senger de Souza [**University of São Paulo** | ***srocio@icmc.usp.br***]
Daniel de Paula Porto [**Federal University of São Carlos** | ***daniel.porto@ufscar.br***]
Vinicius Humberto Serapilha Durelli [**Federal University of São João del-Rei** | ***durelli@ufsj.edu.br***]

Abstract

Context: Testing is fundamental in the software development process. Nevertheless, testing education faces the key challenge of ensuring that undergraduate students acquire knowledge and skills they need for their future careers by matching what is taught in the classroom to industry standards. In this context, gamification can be used as an alternative educational approach. It uses game elements in real-world contexts in order to increase people's motivation and engagement in tasks that require external stimuli, especially in educational contexts. *Objective:* Reporting on results of an experimental study designed to assess the impact of gamification on software testing education, as well as reporting on the experience of building a supporting gamified platform. *Method:* We performed a systematic literature mapping aiming at characterizing how gamification has been explored in the software testing context. In addition, some of the problems faced by testing education were identified through an *ad-hoc* search. Then, we developed a gamified approach and a platform that have been used to run five 4-hour functional testing classes with undergraduate students from four Brazilian institutions of higher education. *Results:* Overall, students who learned with the traditional approach felt more motivated than those who learned with the gamified approach. The performance of both groups was similar, however, feedback questionnaires indicated that the gamified class was more attractive (in terms of attention) and funnier. Moreover, we observed that building a gamified platform is complex and challenging, particularly for the definition of the game mechanics and dynamics. *Conclusion:* Although the results were neutral performance-wise for both groups, and the motivation of students of the control group was slightly higher when compared with the experimental group, the experience of having used gamification is considered positive, as it provided a more enjoyable and funny environment, both from the researcher's and students' point of view. Furthermore, with this experience we foresee what we can do better in terms of gamified teaching in future work.

Keywords: *Education, gamification, game elements, software testing, quality assurance, experience report.*

1 Introduction

Since computers have gone mainstream there has been a tremendous growth in the software industry. Software systems that initially were restricted to the scientific computing and data processing domains now pervade our daily lives. Given the omnipresence of software systems, it is arguable that the quality of these software systems has the potential to have far-reaching consequences. In this context, software testing plays a pivotal rule in improving the quality of software systems. Essentially, software testing is carried out in hopes of ensuring that software systems do what they were designed to do, while also checking that they do not perform anything unintended (Myers et al., 2011). However, despite the profound impact of software quality, software testing education is lagging: there has not been enough emphasis on properly teaching the testing skills needed in the industry. The lion's share of a typical undergraduate Computer Science curriculum is dedicated to development activities (The Joint Task Force on Computing Curricula, 2015); software testing is often neglected in favor of design and implementation activities. To make matters worse, a common misconception among students and practitioners is that software testing is somewhat uninteresting compared to design or coding (Deak et al., 2016). As a result, there is a pronounced shortage of skilled software testers (Smith et al., 2012).

A common challenge in education is engaging students during learning activities that relate to their future professional practices. It is often hard to impart testing concepts and skills to students while keeping them engaged due to the inherent complexity of software testing. Recently, to cope with the challenges of teaching Software Engineering (SE) related concepts and skills, the SE education community has turned to novel pedagogical strategies such as flipped classroom (Paschoal et al., 2017, 2019) and, more substantially, serious games and gamification (Clegg et al., 2017; Rojas and Fraser, 2016a; Anderson et al., 2015; Bell et al., 2011; Sheth et al., 2012; Yujian Fu and Clarke, 2016). Basically, gamification has to do with employing game design elements in non-game settings (Deterding et al., 2011). Put simply, gamification entails creating learning experiences that engage students as if they were playing games. In a sense, learning is gamified in the hopes of tapping into the students' reward and motivation systems.

Based on the premise that gamification is well suited to engage students, we set out to investigate how gamification can be used to improve software testing education. More specifically, we developed a gamified platform whose purpose is to support and engage students in the acquisition of fundamental knowledge on software testing and functional testing concepts. It is worth mentioning that our gamified platform was built based on the insights we gained from a systematic mapping study (Jesus et al., 2018) that we carried out to better understand how gamification has been explored in the context of software testing.

With the support of the developed platform, we ran five experimental sessions comprised of 4-hour classes, in which we taught the basic testing concepts and, primarily, functional testing and its main criteria (Equivalence Partitioning, and Boundary Value Analysis). The subjects were undergraduate students of computing-related programs from four Brazilian institutions. Additionally, we administered an attitudinal survey to the experiment participants in order to get an overview of their attitudes towards our gamified platform. The main benefit for the participants was the opportunity to learn more deeply the most used testing technique in industry.

The results collected with the experimental sessions allow us to: (i) compare the level of motivation and performance of students when both approaches (gamified and traditional[1]) are applied; (ii) report on the experience we gained in developing a gamified platform to support software testing education; (iii) discuss the feedback we received from the experiment participants; and (iv) summarize observations made during the experimental sessions. This paper extends our prior paper (Jesus et al., 2019) by adding data and corresponding analyses for (i) and (iii); in particular, we ran an extra experimental session and performed two additional analyses of results. Furthermore, this paper brings original material concerning (ii) and (iv). Our findings add to a growing body of literature on understanding gamification applied to teaching SE related concepts. Moreover, we highlight that it is possible to adapt our approach in other educational contexts or even in industry.

As for our results, despite the positive outcomes with the use of gamification reported in previous research, we achieved some different results. Overall, we observed that our gamified approach elicited less motivation from the students than the traditional approach. On the other hand, the performance of the students who learned through our gamified approach was on par with the performance of the students that learned via the traditional approach. Beyond this, feedback provided by the students reveals that the gamified approach provided a more enjoyable and funny environment,

The remainder of this paper is organized as follows: Section 2 gives background to our work, describing key concepts of software testing and gamification. Section 2 also describes related work, summarizing studies related to gamification applied to education, particularly for software testing. Section 3 presents the study setup and summarizes the experience we gained while developing the gamified platform. Section 4 reports on the study results regarding students' motivation and performance (Sections 4.1 and 4.2), feedback received from the students (Section 4.3), and researchers' observations made during the experimental sessions (Section 4.4). Section 5 lists some threats to validity and limitations we identified in our research. Finally, conclusions and future work are presented in Section 6.

2 Concepts and Related Work

This section brings the main concepts of software testing and gamification, and briefly discusses how gamification has been investigated for education purposes, particularly for software testing.

2.1 Software Testing

Testing plays a central role in software development projects (Myers et al., 2011): it stands for a key activity performed by industry to evaluate their software products (Ammann and Offutt, 2016), In short, testing consists in executing the program with the objective of revealing faults (Myers et al., 2011). To test a system, the quality assurance analysts create test cases and execute them against the program under test. The oracle (be it a person or an automated procedure) compares the observed system behavior with the expected system behavior and decides if the test passed or failed. In that sense, a widely used testing technique is functional testing, also known as black-box testing (Myers et al., 2011). This technique focuses on functional requirements to derive test requirements. It is traditionally used to demonstrate that the software behaves accordingly by observing that the inputs are accepted by the software and the output is produced as specified.

Given its importance, testing may consume from 30% to 50% (or even more) of IT budget yearly (Capgemini et al., 2018; Myers et al., 2011; Harrold, 2000). On the downside, software testing is considered a destructive and a low-motivating task (Myers et al., 2011). However, while there are many challenges facing the testing community, much effort has also been made to solve (or at least minimize) them. Gamification is a candidate solution, as next described.

2.2 Gamification in Education

Gamification means the use of game elements in non-game contexts (Deterding et al., 2011), in which the fun aspects from games are extracted and inserted into to reach higher purposes. As an example, imagine you are a professor who needs your students to pay attention to the concepts you are teaching. However, nowadays it has a huge adversary: Social Networks. But you are clever and have another strategy as strong as your opponents: the Games! As you do not want to develop a full-game, but rather make use of its benefits, you design an approach in which you reward your students when they have the desired behavior or fulfill the expected assignments. For example, if they ask questions or answer yours, they earn points; if they attend the classes, they earn more points; if a student achieves a goal, you give to him/her a badge, and so on. When a certain score is accumulated, it can be exchanged for benefits such as removing a wrong answer,

[1] We refer to *traditional* as being the approach in which the lecturer explains the concepts and the students just listen to him/her passively.

or consulting studying notes for a period of time during a test. In the end, you get your objective that is attracting your students' attention (so that they can learn the taught concepts), and the students can participate in classes funnier and more enjoyable, being immersed in what is actually happening at that moment.

In the above example, rewards such as points and badges are game elements that might serve as extrinsic motivators for the students to pay attention, participate, and collaborate with their classmates during the classes. This is gamification.

Gamification has been used in educational contexts to stimulate and increase students' motivation and performance (Anderson et al., 2015; Bell et al., 2011; Sheth et al., 2012; Yujian Fu and Clarke, 2016). For example, Anderson et al. presented a cloud-based learning environment (Learn2Mine[2]) developed to support the teaching of programming in data science courses. The environment contains problems in which, solving one by one, the students complete a major lesson. The gamification goals were to increase students' motivation, engagement, and enjoyment with the use of game elements such as points, badge, leaderboard, and duels. Anderson et al. concluded that the approaches they used led to positive outcomes regarding students' performance. Besides that, the Learn2Mine environment is, according to the authors, appropriated to support education in data science.

Pedro et al. (2015), for comparison purposes, developed two educational systems: one gamified, and the other without game elements. The author carried out an experiment during the mathematics classes of students from the sixth year of the elementary education The author evaluated if the game elements reduced negative students' behavior (*e.g.* "gaming the system"), and the results showed that undesirable behavior decreased in the experimental group. Another finding reported was that the male gender gamed the system lesser in the gamified environment, while the female gender avoided that behavior in the control group.

2.3 Related Work

From the studies selected by Jesus et al. (2018) in their systematic mapping, three of them regard CODE DEFENDERS[3], a gamified system for mutation testing education (Rojas and Fraser, 2016a,b; Clegg et al., 2017). In short, mutation testing (DeMillo et al., 1978) consists of creating several slightly modified versions of a program under test (*i.e.* the *mutants*, which are presumably faulty), and of identifying test cases that reveal the introduced faults (in such a case, the mutant is said to be killed.) Among the pursued goals, the CODE DEFENDERS authors aimed to improve students' performance and increase their enjoyment, engagement, and motivation. Beyond the PBL (points, badges, and leaderboards) triad, they also used the duels and team game elements. In one study, CODE DEFENDERS was presented as an early prototype (Rojas and Fraser, 2016a), while in the other two (Rojas and Fraser, 2016b; Clegg et al., 2017), the authors described an approach to provide practical experience for teaching mutation testing. In the years that followed, the authors published two more in-depth studies of the tool. In 2017, the authors presented a study in which the crowdsourcing aspect of the tool was explored (Rojas et al., 2017). In that study, CODE DEFENDERS was compared with other tools for the automatic generation of test cases. The results showed that CODE DEFENDERS generated greater coverage and its mutants are more difficult to kill. More recently, a study explored the use of the CODE DEFENDERS in the classroom (Fraser et al., 2019). The results pointed to students more engaged and satisfied with the use of the tool. It was observed that the performance of students statistically improved throughout the semester, and the existence of a relationship between in-game performance and exam performance.

Yujian Fu and Clarke (2016) proposed a cyber-enabled learning environment called WReSTT-CyLE[4]. It is a gamified system with the PBL triad to engage and motivate students to learn software testing. The system contains tutorials of tools and other content about testing techniques, but no other details such as testing criteria, levels, or process phases, in contrast with our work that explores functional testing and its associated testing criteria. One of the points evaluated by Yujian Fu and Clarke was if there was a relationship between the students' grade and their points earned in the gamified environment. As a result, Yujian Fu and Clarke concluded that the relationship exists (even with biases), and that the WReSTT-CyLE is an efficient system to engage and motivate software testing learning.

Marabesi and Silveira (2019) devised a gamified tool aimed at improving how unit testing is taught in undergraduate education. In their study, the authors described a prototype whose focus was to include game elements that engage students. However, as of the time their paper was published, the tool was still under development, and hence no results were presented.

Given that agile methods and practices have gone mainstream and, along with them, there has been an increasing interest in testing, similar research efforts were conducted by Elgrably and Oliveira (2018) and Costa and Oliveira (2019). Elgrably and Oliveira set out to explore how gamification can be used to better teach agile testing practices to graduate and undergraduate students. According to their results, gamification had a positive impact on the participants, fostering motivation and engagement. On the one hand, our results align with that of Elgrably and Oliveira in terms of students' engagement (Section 4.3 reports on students feedback regarding more attractive classes when gamification was adopted); however, our results pointed to lower motivation level when compared with the traditional teaching approach.

Costa and Oliveira (2019) investigated how gamification can be used to create an environment that is supposedly more conducive to teaching and learning exploratory testing, Costa and Oliveira surmised that the application of game mechanics has the potential to make the process of teaching and learning exploratory testing more interesting, interactive, and engaging. Nevertheless, it is worth mentioning that Costa and Oliveira did not carry out a formal study to evaluate the benefits provided by their proposed gamified environment as we

[2] `http://learn2mine.appspot.com/` - accessed on February, 2020

[3] `http://code-defenders.org/` - last accessed on February, 2020.

[4] `http://wrestt.cis.fiu.edu/` – Accessed on February, 2020.

did in our work.

Also in the context of mutation testing, given the labor-intensiveness of identifying equivalent mutants, Houshmand and Paydar (2017) set out to provide computational and motivational support for the experts involved in this analysis. Specifically, they devised a gamification-based approach for engaging experts in the process of identifying equivalent mutants. Their approach is suitable for testing education purposes as well. Through experiments, Houshmand and Paydar investigated *(i)* whether gamification is able to better engage experts into identifying equivalent mutants, *(ii)* whether gamification improves their overall performance, and *(iii)* if experts pay attention to the game elements. The results would seem to suggest that gamification had a positive impact and was able to improve the effectiveness of the experts in analyzing and identifying equivalent mutants. Houshmand and Paydar also reported that the game elements seemed to create a sense of competition, which in turn made the participants try harder to achieve better positions in the leader board. In addition, according to Houshmand and Paydar, gamification resulted in a noticeable increase in participant involvement.

3 Study Goals and Setup

This section brings details of our research. More specifically, it presents the study goals (Section 3.1), describes the developed gamification approach/platform (Section 3.2), and the experimental material (Section 3.3), characterizes the subjects and experimental sessions (sections 3.4 and 3.5), and describes the methodological approach (Section 3.6).

3.1 Experiment Goals

The primary purpose of this study was to analyze the impact of the use of gamification on software testing education for students' motivation and performance. As complementary goals, we aimed to analyze students' feedback and researchers' observations regarding the usage of the platform during the experimental sessions.

To achieve this goal, we collected data from five experimental sessions performed with the groups of subjects described in Section 3.4. The two first experimental sessions were considered pilot studies that helped us evolve the Bug Hunter platform (described in Section 3.2) to provide better and easier support in subsequent sessions. Details regarding improvements made after the pilot study are presented also in Section 3.2, Step 3 (*Building the Platform*). Data collected in the last three sessions allowed us to address the research questions shown in Table 1.

3.2 Gamification Approach and Platform

To develop the gamification approach and the supporting platform, we followed some steps that are following described.

Step 1 - Literature review: We initially identified several issues concerning software testing education through an *ad-hoc* search at *Google Scholar*[5] engine. We used terms such as *"problems in software testing education"*. The results are shown in Table 2, in which we also present possible solutions, expected behavior, and gamified activities aimed at solving each issue. For example, for the *unattractive classes* (Valle et al., 2017; Pinto and Silva, 2017) issue, we believe that the use of gamification may turn "formal" classes into a funnier and more enjoyable experience. By inserting activities such as *competitive quizzes*, *challenges*, and the use of a gamified platform, we believe students will feel more motivated and engaged.

Some solutions that we present in Table 2 do not directly involve gamification. However, as shown in the *Gamified Activity* column, we can use this alternative approach by inserting game elements into the activities to reach the expected students' behavior, while we try to solve the identified problems. Thus, although gamification usually focuses on addressing specific issues such as turning the classes more enjoyable or boosting adoption, it can also be used to support other possible solutions.

After identifying the challenges in software testing education, and aiming to minimize them, we used a Systematic Mapping (SM) study carried out in our prior research (Jesus et al., 2018) to understand how gamification has been explored to support software testing. In that study, we presented a bubble chart (shown in Figure 1) that combines results based on two data classifications: *used game elements* and *gamification goals*. The numbers in a bubble represent the number of studies that addressed the combination of elements (*e.g.* 12 studies used *points* to *increase engagement* of testers while performing their tasks).

The analysis of the literature, including our SM, supported the definition of the gamification approach as next described.

Step 2 - Gamification approach definition: In this step, we followed six steps to gamification (known as *6D's*) proposed by Werbach and Hunter (2012):

- DEFINE business objectives: Increase students' motivation and performance in software testing educational context;
- DELINEATE target behaviors: Lure students' attention and participation;
- DESCRIBE your players: As it is not possible to identify the player's profile before the experimental sessions, we are going to consider all of them in our design (e.g., socializers, explorers, killers, and achievers (Bartle, 1996));
- DEVISE activity cycles: Present software testing concepts → apply quiz 1 → explain functional testing and its criteria → apply quiz 2 → apply a practical exercise;
- DON'T forget the fun!: Use the gamified tool called ![6] to introduce the fun and competitive aspect during the quizzes.
- DEPLOY the appropriate tools: Implement a gamified platform inserting 10 game elements to support our goals.

[5] `https://scholar.google.com/` – accessed in February, 2020.
[6] `https://kahoot.com/` – accessed in February, 2020.

Table 1. Research Questions and Hypotheses.

Question	Null Hypothesis	Alternative Hypothesis
RQ1. Will the students' motivation who learn in a gamified approach be higher than the students' motivation who learn in a traditional approach?	**H1.0.** There is no difference in the level of motivation of students who learn in a gamified approach in comparison with students who learn in a traditional approach.	**H1.1.** The level of students' motivation who learn in a gamified approach is different to the level of students' motivation who learn in a traditional approach.
RQ2. Will the students' performance who learn in a gamified approach be higher than the students' performance who learn in a traditional approach?	**H2.0.** There is no difference in the performance of students who learn in a gamified approach in comparison with students who learn in a traditional approach.	**H2.1.** The students' performance who learn in a gamified approach is different to the students' performance who learn in a traditional approach.

Table 2. Problems in software testing education.

Problems	Solutions	Expected Behavior	Gamified Activity
Neglected education addressing only an introduction to software testing subject (Valle et al., 2017)	Teach more deeply a testing technique with practical exercises to support the learned concepts	Students focused and participative during classes	Competitive quizzes after each taught concept, rewarded attention and active participation
Difference between what is taught and what needed in industry (Valle et al., 2017; Benitti, 2017)	Teaching the software testing technique and its criteria most used in industry (e.g. Functional Testing)	Use of the Functional Testing criteria to create test cases	Practical exercise in teams
Difference between the levels of what is taught and what is required in tests (Valle et al., 2017)	Require in tests and exercises only what was taught and in an appropriate level	Performance of at least 70% in quizzes	Quizzes and practical exercise
Inefficiency of theoretical classes in the traditional approach education (Smith et al., 2012)	Use an alternative approach education	Students focused and participative during classes	Use the gamified approach described in this paper
Unattractive classes (Valle et al., 2017; Pinto and Silva, 2017)	Use gamification to turn the classes more fun and enjoyable	Students more motivated, engaged, and relaxed	Competitive quizzes, practical exercise in team, use of a gamified platform
Lack of practical exercises (Cheiran et al., 2017)	Apply practical exercises right after teaching a concept	Performance higher than 70% in quizzes	Quizzes and practical exercise

By considering the pursued business objectives are increasing the students' motivation and performance, we defined the following set of 10 game elements to be inserted in our gamified platform: *achievement*, *avatar*, *badge*, *duel*, *leaderboard*, *level*,*points*, *quest*, *team*, and *virtual goods*. We highlight the game elements were selected based on the bubble chart presented in Figure 1.

Based on the information gathered so far, we dealt with a set of challenging questions, such as "*what kind of rewards should we give for each achievement?*", "*what system points should we use in the achievements?*", and "*what score range should we use between the levels?* ". Although there are some studies that proposed frameworks to gamify an environment (García et al., 2017; Dubois and Tamburrelli, 2013; Dal Sasso et al., 2017), and other studies that proposed gamified tools for education purposes (Anderson et al., 2015; Yujian Fu and Clarke, 2016), none of those completely fit our needs (for example, we did not find a solution that supports the visualization of information in a centralized way). Therefore, we decided to adopt *Google Sheets*[7] to create the game mechanics and the interactions among them, as depicted in Figure 2. The focus of the spreadsheet was on the definition of the scoring strategy (*Points* System) for each level (columns H to J), and on how to split activities and rewards within each *level* (columns B to F). In doing so, each *level* contains a set of activities to be performed (column C), the goals to be pursued (column D), the rewards (column E) and their values (column F).

The rules to reward students for their achievements rely on avatar upgrades, points, virtual goods ($ coins), badges, and score ranges for each level. We first defined a score range from 0 to 100 points, being 0 in level 1, and 100 in the last level. However, we thought that each reward would apply a short score for expected behavior, which could be "a big pain for a little gain." We kept refining the approach until the final list of goals and their rewards was done, as partially shown in Table 3.

Step 3 - Building the platform: After creating and refining the approach, we started to define the gamified environment called *Bug Hunter*[8] with the 10 game elements defined in our

[7] `https://docs.google.com/spreadsheets/` – accessed in February, 2020.

[8] `https://bit.ly/2Ei8JbW` – accessed in February, 2020.

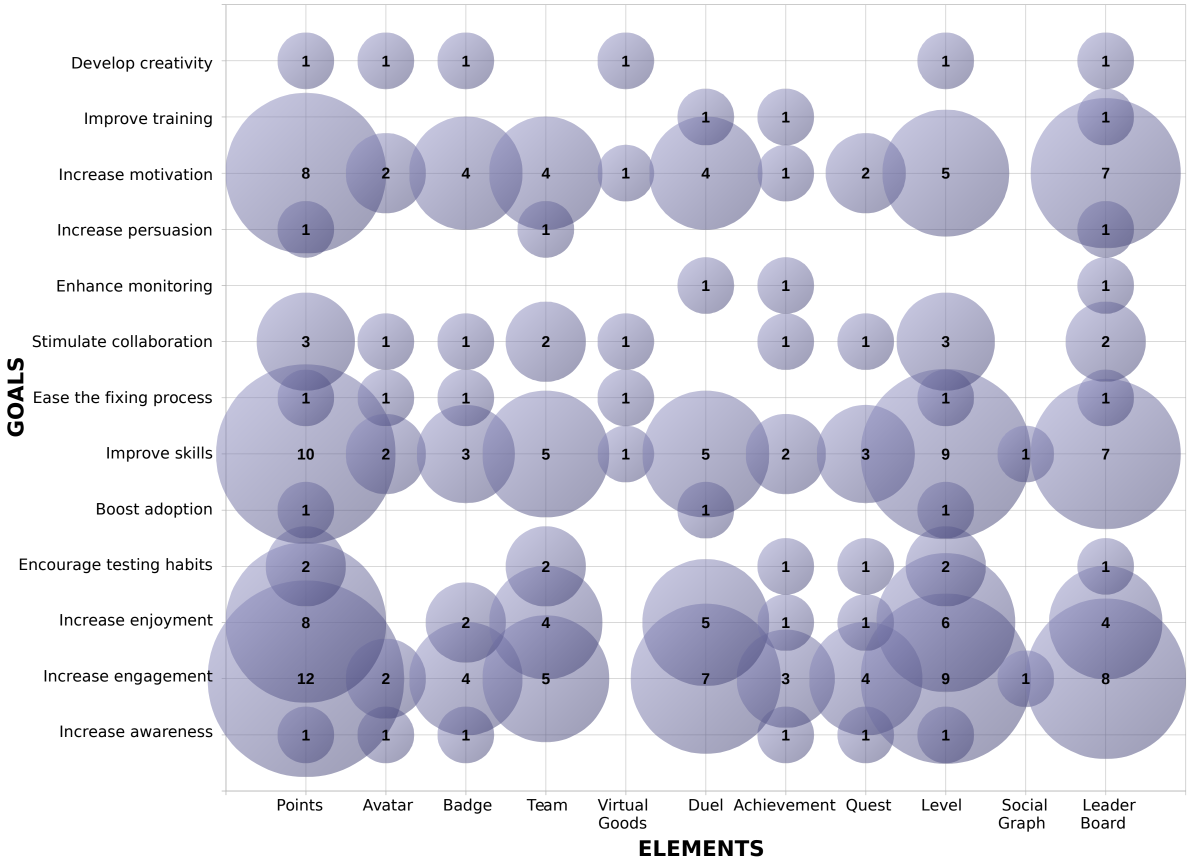
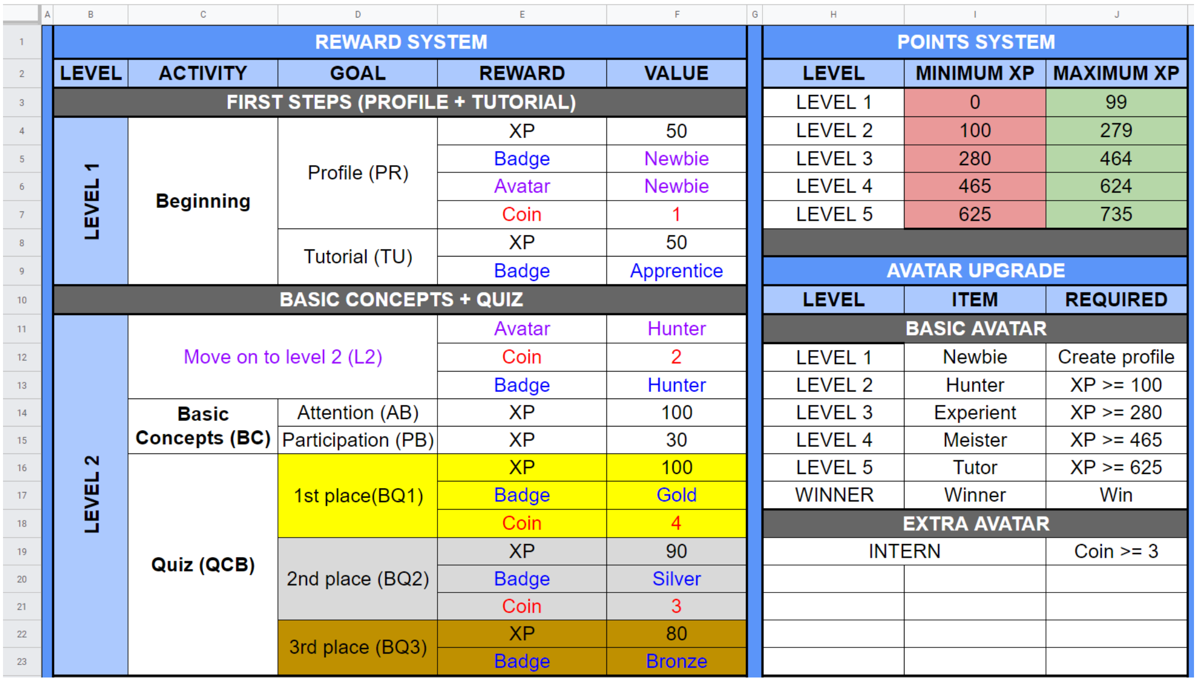


Figure 1. Number of studies that relate gamification elements to gamification goals (Jesus et al., 2018).



REWARD SYSTEM				
LEVEL	ACTIVITY	GOAL	REWARD	VALUE
FIRST STEPS (PROFILE + TUTORIAL)				
LEVEL 1	Beginning	Profile (PR)	XP	50
			Badge	Newbie
			Avatar	Newbie
			Coin	1
		Tutorial (TU)	XP	50
			Badge	Apprentice
BASIC CONCEPTS + QUIZ				
LEVEL 2	Move on to level 2 (L2)		Avatar	Hunter
			Coin	2
			Badge	Hunter
	Basic Concepts (BC)	Attention (AB)	XP	100
		Participation (PB)	XP	30
	Quiz (QCB)	1st place(BQ1)	XP	100
			Badge	Gold
			Coin	4
		2nd place (BQ2)	XP	90
			Badge	Silver
			Coin	3
		3rd place (BQ3)	XP	80
			Badge	Bronze

POINTS SYSTEM		
LEVEL	MINIMUM XP	MAXIMUM XP
LEVEL 1	0	99
LEVEL 2	100	279
LEVEL 3	280	464
LEVEL 4	465	624
LEVEL 5	625	735

AVATAR UPGRADE		
LEVEL	ITEM	REQUIRED
BASIC AVATAR		
LEVEL 1	Newbie	Create profile
LEVEL 2	Hunter	XP >= 100
LEVEL 3	Experient	XP >= 280
LEVEL 4	Meister	XP >= 465
LEVEL 5	Tutor	XP >= 625
WINNER	Winner	Win
EXTRA AVATAR		
INTERN		Coin >= 3

Figure 2. Partial view of the approach definition spreadsheet.

gamified approach. Beyond promoting competitiveness and fun, as well as making classes more enjoyable, by using the platform, the main behavior we expected from the students was:

- Attention: the students should pay attention to the explanation during the class. The applied competitive quizzes, for example, were a strategy to attract their attention because, as they knew that they would have to answer the correct option for each question, they could not be dis-

Table 3. Reward rules (partially shown).

	REWARDS			
Achievements	**XP**	**$**	**Badge**	**Avatar**
Access the platform	50	1	Newbie	Newbie
Follow the tutorial	50	-	Apprentice	-
Reach level 2	-	2	Hunter	Hunter
Pay attention - Basic concepts	100	-	-	-
Be participative - Basic concepts	30	-	-	-
Participate on Quiz 1 - 1º place	100	4	Gold	-
Participate on Quiz 1 - 2º place	90	3	Silver	-
Participate on Quiz 1 - 3º place	80	-	Bronze	-
Participate on Quiz 1 - other scores	70	-	-	-
Reach level 3	-	-	Expert	Expert
...				

tracted. Thus, we challenge them and rewarded the best 3 students in each quiz with points, badges, virtual goods, and the possibility of upgrading their avatar if their score sums up the needed value to level up.

- Active participation: the students should participate by asking and answering questions To encourage them to participate, the instructor rewarded the students with points and a *Participative* badge.

- Collaboration: the students should collaborate with each other in the final challenge; besides that, the platform has a tab called *Forum*, in which the students could post their questions and answers. Badge, points, and virtual goods were used to motivate collaboration among students. Although collaboration and competitiveness might be contradictory, the former was motivated by expectations from an industry viewpoint; working in teams is one of the soft skills required in recruitment processes. Competition, differently, is intended to increase fun during the activity (by applying a quiz, this indeed happened). The competition dynamics were introduced during the quiz, but in the challenge, the expectation was an intra-group collaboration to win the challenge; without it, fewer rewards would be achieved, thus affecting both students and their teams.

- Application of learned concepts: the students should use the taught functional testing criteria (*e.g.* Equivalence Partitioning, and Boundary Value Analysis) to create their test suite during the practical exercise. Working as a team, the students were divided into groups and had a mission of revealing faults in the system under test.

As for the approach creation, no existing gamified tool completely met the needs of this study. In some cases, the tools did not employ the set of game elements defined for this research. In other cases, they were not open source projects, whereas in others the tools focused on supporting students outside the classroom. Some tools aimed to support teaching software testing within other disciplines such as data science, yet others focused on long-term analysis experiments (experiments conducted throughout the semester).

As a solution, we again chose to develop our platform in the *Google Sheets* environment because it is a simple, well-known, free tool, and can be shared with others. Therefore, the information and awarded rewards are automatically synchronized and viewed by every participant.

Figure 3 depicts a template of Bug Hunter's *Administrative* tab, in which the instructor rewards the students and posts information throughout the class. We highlight that the platform was improved after gathering feedback from the pilot study. While running the pilot study, the researcher felt that it was hard to manually apply all rewards (namely, *XPs*[9], *badges*, *avatar* upgrade, and *coins*) to all participants. Thus, we refined the platform and the only information manually entered by the researcher during the experiments were the posts in the notification area and the assignment of *XPs* for students who achieved a goal. The other rewards are automatically applied based on the earned *XPs*.

In the *Administrative* tab, the *XPs* are manually entered in every column labeled with "*XP*" (see line 4 in Figure 3), except column *C* that automatically sums up the *XPs* earned by each student. Besides that, notifications are posted in the notification section and the *leaderboard* provides feedback about the current ranking. The other rewards (namely, *badges*, *virtual goods* ($ coins), *levels*, and *avatar* upgrades) are automatically calculated and shown by the platform as the students fulfill their assignments.

Beyond the *Administrative* tab, Bug Hunter has the following tabs: *Students*, *Rules*, *Badges*, *Avatar*, *Virtual Goods*, and *Forum*. Figure 4 depicts part of the *Avatar* tab, which contains the avatar of each level, including a special (namely, *Winner*) that only the winner receives. Besides those, there is an extra upgrade that could be bought by the participants by spending $3 virtual *coins*.

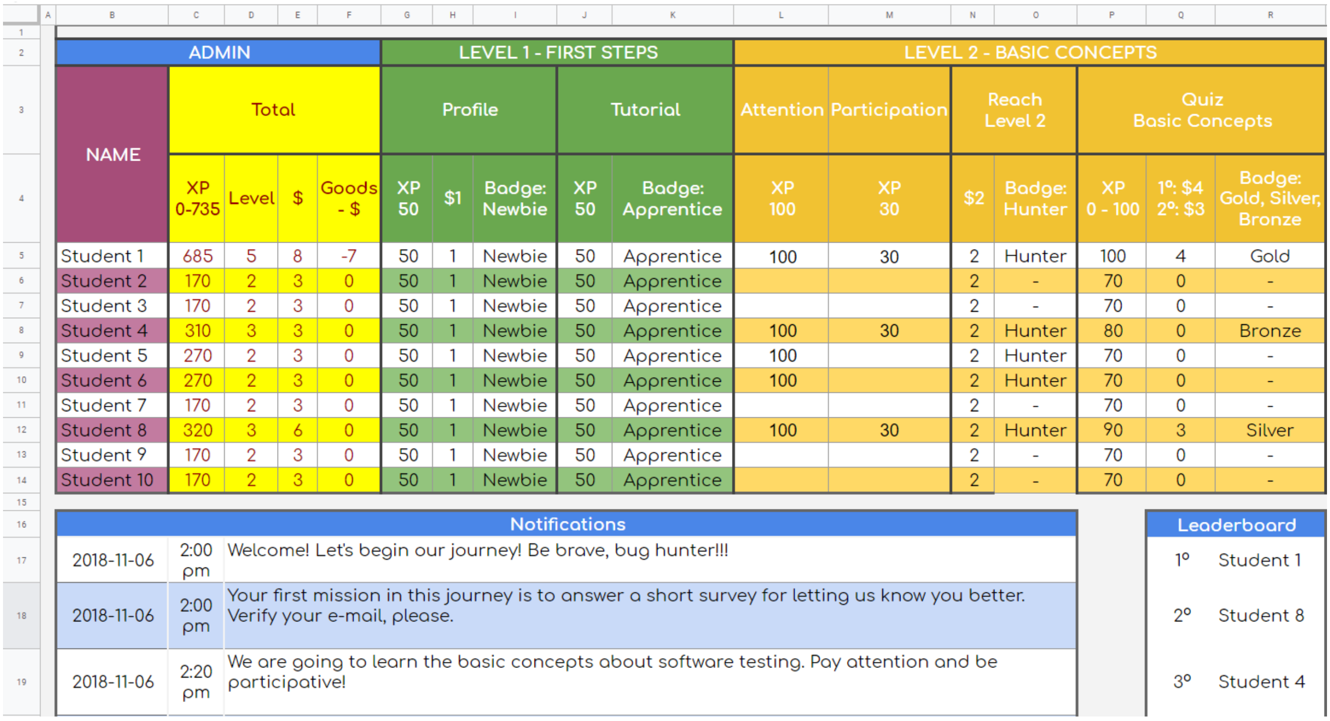
In the *Students* tab, each participant had a dashboard such as the one shown in Figure 5. In this tab, the student receives feedback about his/her progress, rewards, available virtual goods, ranking, and the notifications sent by the researcher. Note that all information in this tab is linked to the administrative tab and automatically filled in as the conditions are satisfied.

The remaining tabs are *Rules*, *Badges*, *Virtual Goods*, and *Forum*. The *Rules* tab contains information regarding the goals, what was necessary to achieve them, and the rewards earned in each one. The *Badges* tab contains all of the badges the participants could earn, while the *Virtual Goods* tab contains three virtual goods that could be bought by the students, their description, and price. Finally, the *Forum* tab is a virtual place where the participants could post their questions, comments, or answers to other students' questions. Due to space limitations, details regarding these tabs can be checked in the online platform[8].

How the "game" is played: First of all, it is worth explaining that gamification is not a *game*. These are two different concepts, once gamification means the "simple" usage of the game elements in a non-game context (Deterding et al., 2011). Thus, the game elements inserted into the Bug Hunter platform has a visual appeal in order to stimulate students' motivation through the dynamics of the given rewards.

That said, for using Bug Hunter in a new class, initially the instructor has to create a clean copy of the spreadsheet. Subsequently, the names of the students are placed in the Admin-

[9] *XP* means "Experience Points".



ADMIN					LEVEL 1 - FIRST STEPS					LEVEL 2 - BASIC CONCEPTS						
NAME	Total				Profile			Tutorial		Attention	Participation	Reach Level 2		Quiz Basic Concepts		
	XP 0-735	Level	$	Goods - $	XP 50	$1	Badge: Newbie	XP 50	Badge: Apprentice	XP 100	XP 30	$2	Badge: Hunter	XP 0 - 100	1º: $4 2º: $3	Badge: Gold, Silver, Bronze
Student 1	685	5	8	-7	50	1	Newbie	50	Apprentice	100	30	2	Hunter	100	4	Gold
Student 2	170	2	3	0	50	1	Newbie	50	Apprentice			2	-	70	0	-
Student 3	170	2	3	0	50	1	Newbie	50	Apprentice			2	-	70	0	-
Student 4	310	3	3	0	50	1	Newbie	50	Apprentice	100	30	2	Hunter	80	0	Bronze
Student 5	270	2	3	0	50	1	Newbie	50	Apprentice	100		2	Hunter	70	0	-
Student 6	270	2	3	0	50	1	Newbie	50	Apprentice	100		2	Hunter	70	0	-
Student 7	170	2	3	0	50	1	Newbie	50	Apprentice			2	-	70	0	-
Student 8	320	3	6	0	50	1	Newbie	50	Apprentice	100	30	2	Hunter	90	3	Silver
Student 9	170	2	3	0	50	1	Newbie	50	Apprentice			2	-	70	0	-
Student 10	170	2	3	0	50	1	Newbie	50	Apprentice			2	-	70	0	-

Notifications		
2018-11-06	2:00 pm	Welcome! Let's begin our journey! Be brave, bug hunter!!!
2018-11-06	2:00 pm	Your first mission in this journey is to answer a short survey for letting us know you better. Verify your e-mail, please.
2018-11-06	2:20 pm	We are going to learn the basic concepts about software testing. Pay attention and be participative!

Leaderboard	
1º	Student 1
2º	Student 8
3º	Student 4

Figure 3. Bug Hunter - Administrative tab.

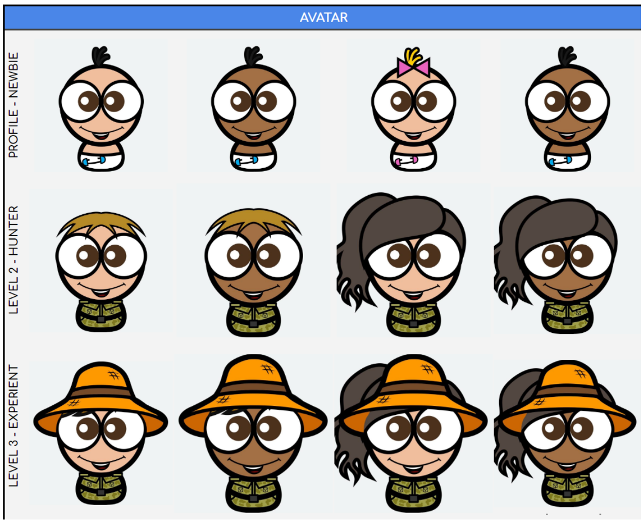


Figure 4. Bug Hunter - Avatar tab.

istrative tab (column B) and in the student's tab (See Figure 5). Notice that in the latter case, one tab is created for each student.

Every interaction between the students and the platform occurs between the teaching sessions (that is, interactions do not happen simultaneously with the presentation of the taught topics). Rewards (XPs and coins) are given in the class interval and/or right after the class assignments, and students are promptly notified to check their progress. At the end of the class, the first student in the leader board is declared the winner, and the effort of the other students are positively acknowledged.

3.3 Experimental Material

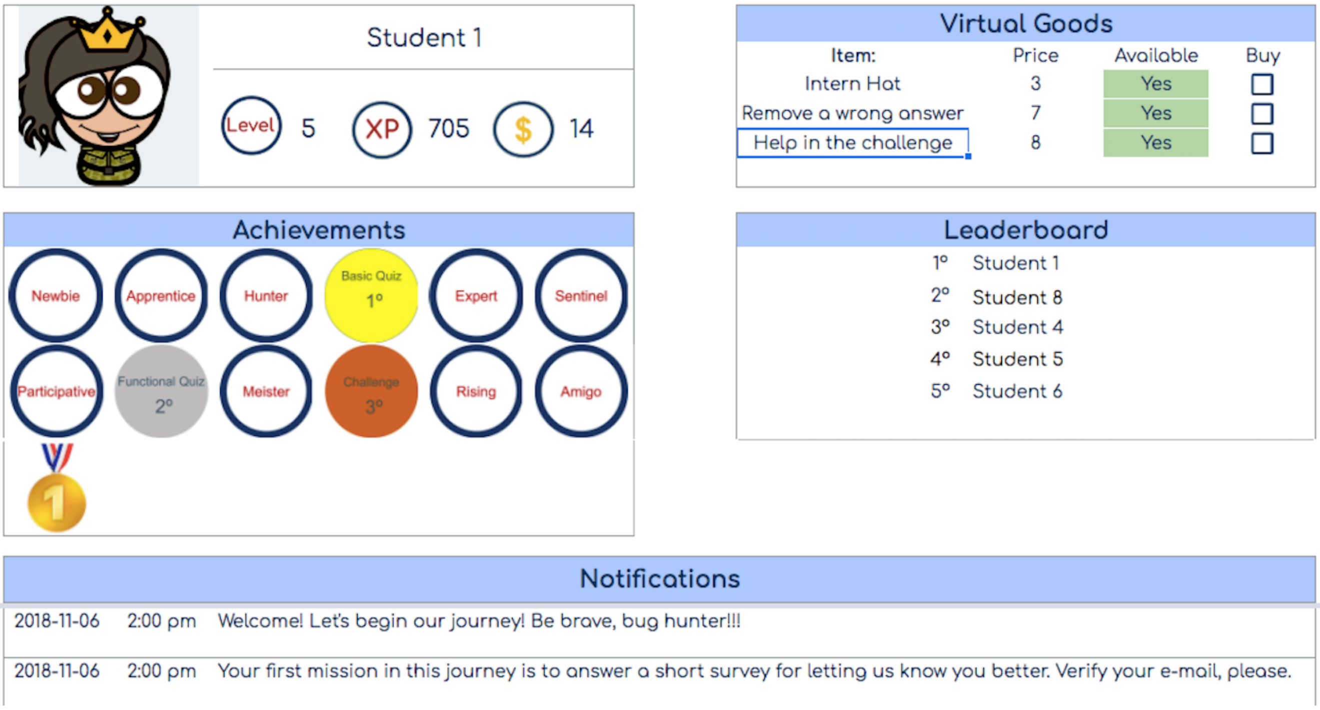
To answer the research questions presented in Table 1, we analyzed if the gamified approach to teaching software testing contributes to higher students' motivation and performance compared to the traditional approach. This analysis was carried out through two metrics. The first metric is related to the students' motivation and the second is related to their performance[10].

The students' motivation was measured through the well-known *Intrinsic Motivation Inventory* (IMI), a multidimensional measurement instrument intended to assess the subjects' subjective experience related to activities in laboratory experiments. The IMI was proposed (and applied) by Deci and Ryan (2011), and comprehends 22 short questions with a 7-value Likert scale. These questions are meant to measure the motivation in four categories: interest/enjoyment, perceived competence, perceived choice, and felt pressure and tension. In our research, we applied a Portuguese version of IMI whose translation was done and validated by Pedro (2016).

The students' performance was measured and tracked through two quizzes. The first one encompasses questions related to basic concepts of software testing and its terminologies, such as quality assurance, software testing process, levels, and techniques. The second quiz encompasses questions related to both basic concepts of software testing, and the functional testing technique and its Boundary Value Analysis and Equivalence Partitioning criteria.

Other analyses were performed in this study. In the first analysis, we applied a pre- and post-test with three questions to assess the students' previous knowledge about software testing, and to check (after the experiment) if they learned what the software testing objective is, its importance, and what the functional testing technique is. In the second analysis, we applied a practical exercise in which the students were

[10] In performance, we considered the students' learning (measured with knowledge tests) regarding the taught testing concepts.



Student 1

Level 5 XP 705 $ 14

Virtual Goods

Item:	Price	Available	Buy
Intern Hat	3	Yes	☐
Remove a wrong answer	7	Yes	☐
Help in the challenge	8	Yes	☐

Achievements

Leaderboard

1º	Student 1
2º	Student 8
3º	Student 4
4º	Student 5
5º	Student 6

Notifications

2018-11-06	2:00 pm	Welcome! Let's begin our journey! Be brave, bug hunter!!!
2018-11-06	2:00 pm	Your first mission in this journey is to answer a short survey for letting us know you better. Verify your e-mail, please.

Figure 5. Gamified Environment - Student 1's tab.

supposed to create test cases based on three requirements for a specific e-commerce website presented to them. After having created the test cases, they were supposed to execute them against the website. At the end of this exercise, we analyzed if the students applied the functional testing criteria—and which ones—taught during the experiment sessions. Finally, we also applied a questionnaire aiming to identify if the used approach helped to minimize the challenges faced in software testing education.

The analysis of results regarding students' motivation and performance are presented in Sections 4.1 and 4.2, respectively. The analysis of this last aforementioned questionnaire is presented in Section 4.3.

3.4 Subjects

To perform this study, we invited undergraduate students from four Brazilian Institutions of Higher Education (IHE): A, B, C and D[11]. As shown in Table 4, the sample comprised 70 students, in which 13 were from IHE A, 18 were from IHE B, 22 were from IHE C, and 17 students were from IHE D.

Table 4. Demographic data.

		#	# considered	# considered	
Approach	**IHE**	**subjects**	**activities**	**IMI**	**Course**
Traditional	A	8	8	7	IS
Gamified	A	5	4	4	IS
Gamified	B	18	11	11	IS
Traditional	C	22	22	22	SAD
Gamified	D	17	16	16	CS/CE
TOTAL		70	61	60	

As the pilot-study was performed at IHE A, we are not going further on that. Nevertheless, we will describe the subjects who participated in all the sessions.

Subjects from IHE A and B were Information Systems (IS) undergraduate students. Subjects from IHE C were System Analysis and Development (SAD) undergraduate students. Finally, subjects from IHE D were either Computer Science (CS) or Computer Engineering (CE) undergraduate students. All participants in the experimental sessions were either middle or final year students.

We highlight that in the IHEs B, C and D the experimental sessions were conducted either as ordinary classes or as complementary credits, all during the regular academic semester. As such, this context characterizes a real course environment, even though with a limited time length (*i.e.* a 4-hour short course). Furthermore, we highlight that the choice for an IHE was random. In other words, we did not specify that students from a particular IHE would take a gamified/traditional short course. Instead, as long as the course coordinators/professors signaled positively to the introduction of the short course into the agenda of classes, we arbitrarily chose a teaching approach (traditional or gamified). Only for IHE D the choice was predefined, since we needed to enlarge the size of the experimental group. We decided not to further divide participants from the same IHE into subgroups because this would lead to some issues: (1) the size of these groups would end up being too small, and (2) we would run into (logistics and organizational) problems to organize the experiments so that they would take place in the time frame available for the participants (most participants had only a small time frame available).

To analyze the impact of gamification on students' motivation, we considered the participants who answered the *Intrinsic Motivation Inventory* (IMI). In total, we had 11 subjects from the experimental group at IHE B, 22 students from the

[11] To keep anonymity, we omit the names of the host institutions.

control group at IHE *C*, and 16 students from the experimental group at IHE *D*.

To analyze the impact of gamification on students' performance, we considered only the students who participated in the two quizzes and in the practical activity. In the experimental group at IHE *B*, 4 participants did not attend the two parts of the experimental session (*i.e.* they did not answer both quizzes), and other 3 did not participate in the practical activity. Therefore, we considered 11 out of 18 participants. At IHE *C*, all 22 students participated in all activities. Lastly, in the experimental group at IHE *D*, 1 participant did not attend the two parts of the session, and hence we considered 16 out of 17 participants.

None of the participants had any previous professional experience, thus we assume that there was not a significant difference in terms of practical knowledge between control and experimental group. Additionally, the participants were asked if they had already studied software testing: none of the participants had contact with software testing beyond basic concepts that they were exposed to during previous courses (a topic in the software engineering discipline at IHE *B*, and in the software quality discipline at IHE *C*). Students from IHE *D* had not been taught any software testing concept.

Notice that all participants were aware they were volunteers and it would not imply in any punishment if they quit the experiment. Moreover, the researcher who applied the sessions, and the professors who invited their students, highlighted the gains students could have while learning the most used software testing technique in the industry.

3.5 Experimental Sessions

The same researcher conducted the five experimental sessions. Sessions in IHE *A*, *B* and *C* were conducted in the second semester of 2018. The last session was conducted in IHE *D* in the second semester of 2019. The pilot sessions had a duration of five hours and the remaining ones had a duration of four hours each. All sessions were split into two parts with the same time length, and the interval happened between the parts.

At IHE *A*, the researcher ran the two pilot sessions. The first session was with the control group, in which a traditional approach was applied, and the second one was with the experimental group using gamification. These pilot sessions had refinement purposes with respect to: the content taught, the gamified platform developed to support the experimental group (more details in Section 3.2, Step 3), the questionnaires, the quizzes, the practical activity, and the time for running the experiment. Regarding the content taught, we noticed that the students got confused with the names of each criteria of each software testing technique. Thus, we focused only on listing the three testing techniques (*i.e.* functional, structural, and fault-based testing), and on explaining more deeply the two functional testing criteria: Equivalence Partitioning, and Boundary Value Analysis. Besides that, we also decreased the time of execution of the experiment in one hour (from five to four hours each session). The quizzes were refined to be aligned with the content taught. In the practical activity, instead of allowing the students to choose any e-commerce website on their own, we defined a specific one[12] to be tested by them. Aiming to minimize the researcher's efforts during the class, we automated the gamified platform to calculate and reward the students with points, badges, levels, ranking, avatar upgrades, etc. Finally, the questionnaires were also refined to include questions more related to the issues on software testing education we were investigating, and whether gamification helped to minimize them.

In the third session, the researcher taught software testing using the gamified approach at IHE *B*. A professor of the IS course from this IHE invited his/her students to participate in the experiment and provided his/her two classes in two different days in a week (four hours in total).

The fourth session was performed using the traditional approach with the control group at IHE *C*. The coordinator of the SAD course invited his/her students to participate in the experiment in an afternoon, out of ordinary class time.

Finally, the fifth session was performed using the gamified approach with the experimental group at IHE *D*. The professor of the Software Engineering 2 course invited his/her students to participate in the experiment in a morning, ordinary class time.

3.6 Methodological Approach

We followed the experimental process described by Wohlin et al. (2012). The design included one factor (teaching approach) and two treatments (traditional and gamified). In this perspective, the teaching approaches were independent variables that caused some effect on the results (*i.e.* students learning).

To run the experimental sessions, we followed some steps, as shown in Figure 6. On stage 1, weeks before beginning the experimental session in a given IHE, we invited the students to participate in the study. After that, in the second stage, the research was presented to the subjects, who were given the consent form to be filled out. In the sequence, the pre-test was performed by the subjects (recall that the pre-test aimed to characterize previous knowledge on functional testing, including its importance and goals). After that, a tutorial of the gamified platform was presented for the experimental group; for the control group this step was not followed, once the Bug Hunter was not used in the traditional approach.

Stage 3 started with the presentation of basic concepts of software testing, including main terminology (*e.g.* input domain, test data, test case, test oracle, test suite etc.), test levels, testing techniques, and phases of a testing process. Afterwards, we applied a quiz that encompassed questions regarding the contents presented so far.

After the quiz, the researcher taught the functional testing technique and its associated criteria Equivalence Partitioning and Boundary Value Analysis. Along with the explanations, the researcher encouraged the students to solve an example. After that, a second quiz was applied. It included questions of both sets of topics (namely, basic concepts and functional testing). For the experimental groups, we used the gamified tool called Kahoot! to apply the two quizzes and the non-

[12] `https://www.casasbahia.com.br/` – accessed in February, 2020.

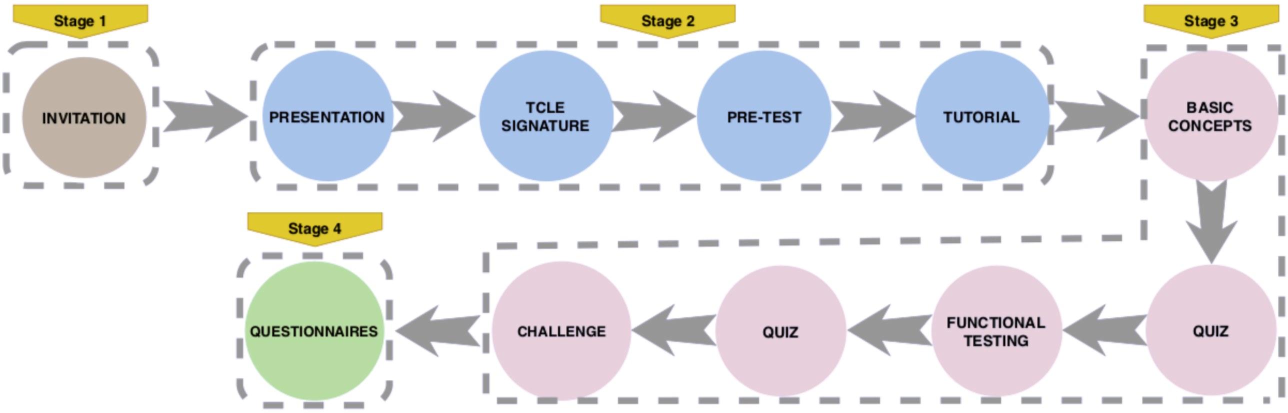


Figure 6. Methodological Approach.

gamified tool Google Forms[13] for the control group. We highlight that the same questions were applied to both groups.

Once finished the second quiz, the students were organized in groups of 3 or 4 members to perform the final activity. They were asked to create and execute a test suite based on three functional requirements presented to them at the beginning of the activity. After that, we analyzed if the groups used one, two, or none of the functional testing criteria taught during the experiment to create their test cases.

In stage 4 (the last one), participants were asked to answer four questionnaires: post-test, Intrinsic Motivation Inventory (IMI), Short-course Evaluation, and Gamified Platform Evaluation. There four questionnaires, together with the pre-test, were assessed with the conduction of the pilot study (described at the beginning of Section 3.5).

4 Results and Analysis

This section presents the experiment results. We compared results regarding the control group (CG) (*i.e.*, 22 students from IHE C) and the experimental group 1 (EG1) (*i.e.*, 11 students from IHE B), the control group (CG) and the experimental group 2 (EG2) (*i.e.*, 16 students from IHE D), and the control group (CG) and the combined experimental groups 1 and 2 (EG1+EG2), hence summing up 27 students. Note that EG2 helped us get the samples balanced in terms of the number of subjects. This motivated us to combine EG1 and EG2 to perform additional analysis.

Initially, we gathered descriptive statistics (*e.g.* mean, median, mode, variance). Additionally, we applied the Shapiro-Wilk test to check if the data had a normal distribution, with a confidence interval $\alpha = 0.05$. Given that the data had a non-normal distribution, we applied the Mann-Whitney, which is a non-parametric test that we used to compare differences between two independent groups (the control group and each experimental group).

We graded all pre-tests and post-tests for students' performance analysis. We then calculated a Δ value, which represents the increment of knowledge observed after a learning session. Notice that a similar approach was adopted in prior research (Lyra et al., 2016; Paschoal et al., 2019) as a way to measure how much a student learned after a class. Δ is calculated with the following formula, where X and Y are the numbers of correct answers in the post-test and pre-test, respectively, and i is a given student.

$$\Delta = X_{(\text{i})} - Y_{(\text{i})}$$

To complement the performance analysis, we used the quizzes to keep track of the students' performance along with the teaching sessions. We graded the quizzes and calculated the students' efficacy concerning the maximum number of correct answers through the following formula, where n represents the student's number of correct answers, i is a given student, and $TOTAL$ is the total number of quiz questions.

$$Efficacy_{(\text{i})} = \frac{n_{(\text{i})}}{TOTAL}$$

We applied the Intrinsic Motivation Inventory (IMI) to compare the students' motivation of all groups (control group, and each experimental group), considering four aspects, which are: (a) *Interest/enjoyment*, (b) *Perceived choice*, (c) *Perceived competence*, and (d) *Pressure/tension*. The questionnaire contains 22 short questions (each with a 7-value Likert scale), and a set of them are related to each of the four aspects. To measure each aspect, first, we must calculate the mean of the answers for each question. After that, we sum up the means of the set of questions related to a specific aspect and repeat this process to the other three aspects. For example, questions 1, 5, 8, 10, 14, 17, and 20 are related to *Interest/enjoyment*, and (supposedly) 10 students responded the questionnaire. We calculate the mean of the 10 answers on question 1, then for question 5, and so on. After that, we sum up the 7 means and obtain the value for this motivational aspect.

Finally, we analyzed the questionnaires that aimed to collect students' feedback about the two teaching approaches. We also used a 7-value Likert scale for each of the 10 questions. The lowest value (=1) meant *totally disagree*, and the highest value (=7) meant *totally agree*. We then used the following formula to evaluate the answers from both groups, where x represents an answer (*i.e.* a value from 1 to 7) for a given question, N is the number of times an answer for a given question was provided, and i is a given student.

$$Result = \sum_{i=1}^{N} x_{(\text{i})}$$

[13] `https://www.google.com/forms/about/` – accessed in February, 2020.

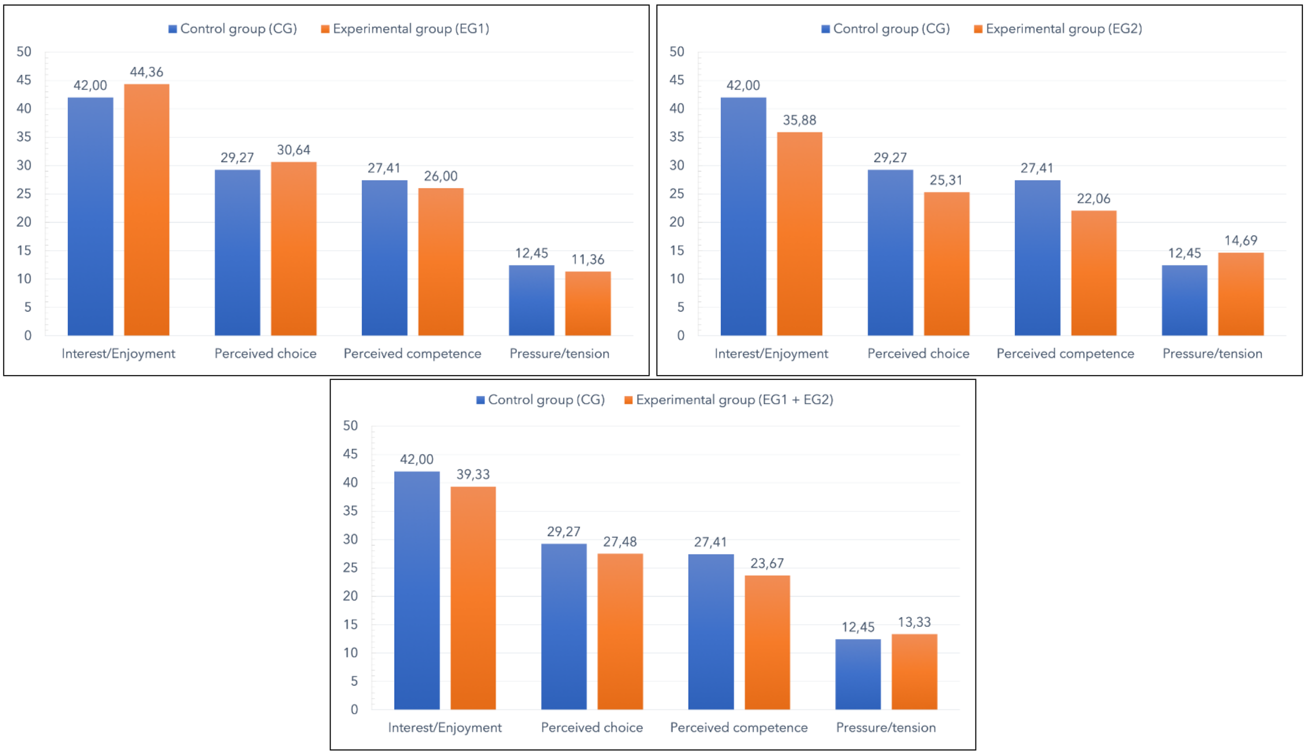


Figure 7. Motivation results.

4.1 Results Regarding Students' Motivation

Figure 7 depicts the results of the Intrinsic Motivation Inventory (IMI) applied to the students from all groups. In the first comparison (CG vs. EG1), for the *Interest/enjoyment* and *Perceived choice* aspects, the experimental group had a higher score than the control group. The opposite occurred in the other two aspects (namely, *Perceived competence* and *Pressure/tension*). Therefore, we believe the experimental group enjoyed and had more interest in learning software testing with the gamified approach. Besides that, although the students from the control group had felt more pressure/tension during the class, they also perceived themselves more competent than the experimental group to perform the activities, which is confirmed in the results of the analysis for students' performance (Section 4.2).

In the second and third comparisons (that is, CG vs. EG2, and CG vs. EG1+EG2, respectively), we observed lower motivation for the experimental groups, for all aspects (note that higher marks for *pressure/tension* also represent negative results). In this sense, we believe students from the experimental group have felt less free and competent, with lower interest and enjoyment, and tenser and under pressure than that those of the control group.

Regarding **RQ1** (*Will the students' motivation who learn in a gamified approach be higher than the students' motivation who learn in a traditional approach?*), in two out of three analyses (namely, CG vs. EG2, and CG vs. EG1+EG2), we found out that the experimental group was less motivated than the control group, thus the H1.0 (null hypothesis) is rejected in favor of the traditional approach (H1.1).

4.2 Results Regarding Students' Performance

Based on the grading of pre-tests and post-tests, we noticed that the control group achieved lower grades in the pre-test than the experimental group in all comparisons, as shown in the *box-plots* charts of Figure 8. While the maximum grade for the control group was 2, the corresponding values for the experimental group were 3, 2 and 3 for EG1, EG2 and EG1+EG2, respectively. The median values reinforce these results. In the post-test, grades were mostly similar in two comparisons (CG vs. EG1; and CG vs. EG1+EG2), whereas EG2 solely performed better than CG. In summary, results suggest that the traditional approach has been more effective in improving students' knowledge when compared with the gamified approach.

To check which teaching approach had better results concerning students' performance, we calculated the Δ values. Figure 9 summarizes the results for all comparisons. It shows larger variations for the control group, and hence no trend for a particular value. The median values show that the control group had higher Δ values than the experimental groups. Nevertheless, the Mann-Whitney tests indicate no significant difference for CG vs. EG1 (*p-value* = 0.1010), for CG vs. EG2, (*p-value* = 0.4180), and for CG vs. EG1+EG2 (*p-value* = 0.3090). Therefore, we conclude that there was no difference regarding the learning level when either traditional or gamified approaches are adopted.

We also analyzed students' performance along with the experiment observing the students' efficacy to answer correctly the quiz questions. Thus, we could compare the performance of the experimental and control groups in each taught content (*i.e.* basic concepts, and functional testing criteria).

The *box-plots* of Figure 10 depict the participants' efficacy in answering correctly the questions about the basic concepts

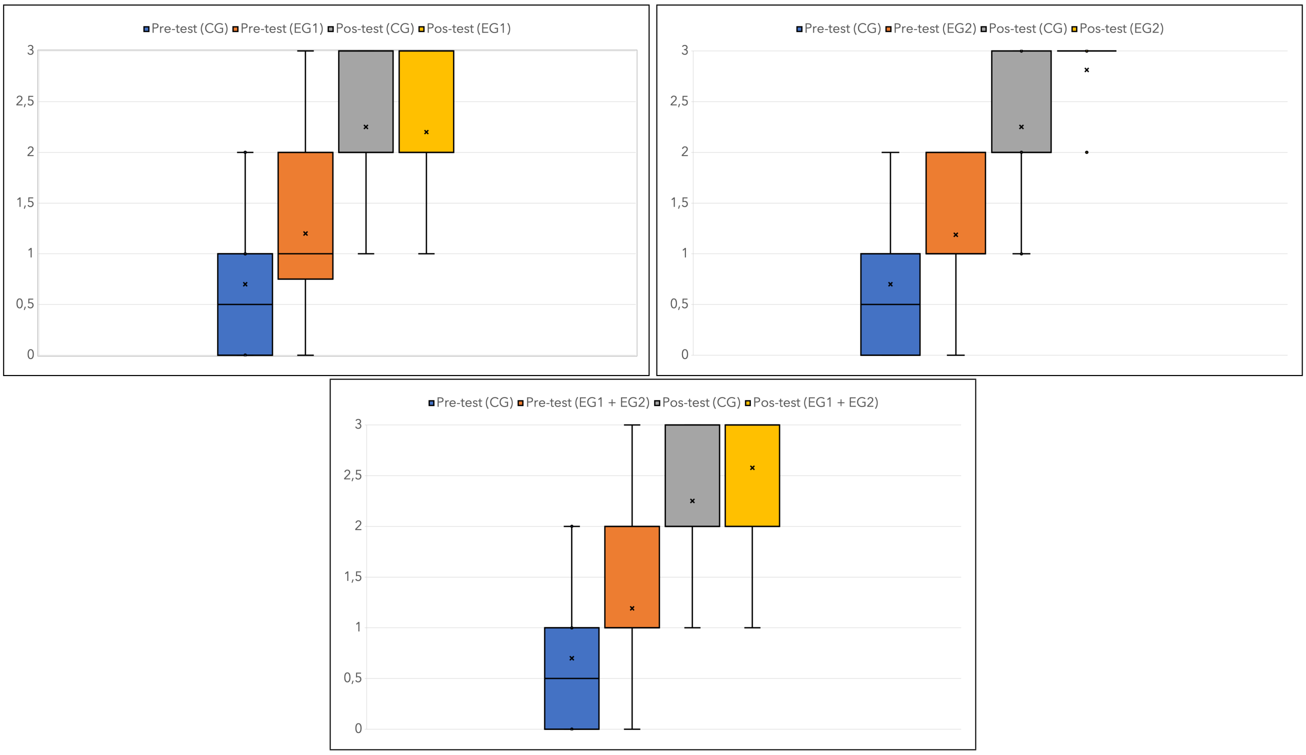


Figure 8. Knowledge tests results.

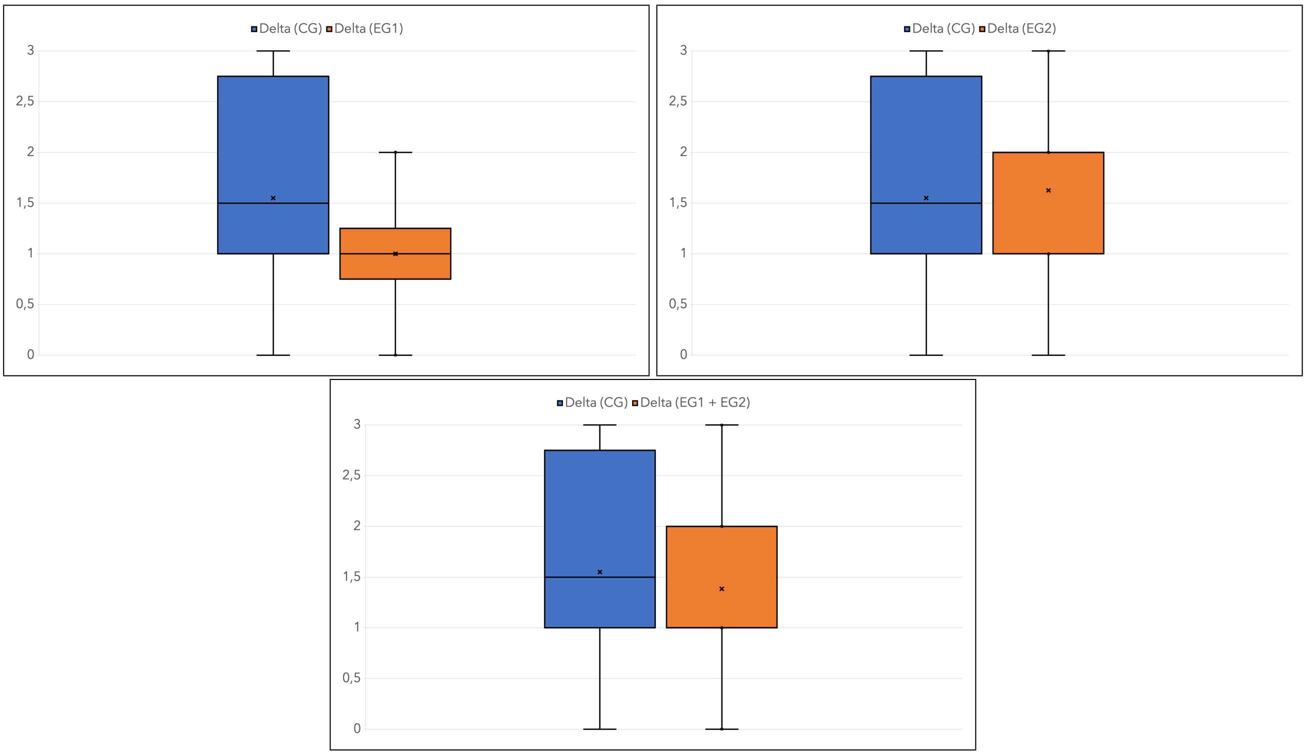


Figure 9. Summary of the Δs.

of software testing (that is, the charts regard the first quiz). The left-hand chart includes results for comparisons CG vs. EG1 and CG vs. EG2, whereas the right-hand chart brings results for CG vs. EG1+EG2. As seen, the median of the control group was 90%, whereas the experimental groups obtained the median of 70%, 80% and 70% of efficacy for EG1, EG2, and EG1+EG2, respectively. Regarding the minimum values, the control group obtained 50% of efficacy, while for the experimental groups this value was 50%, 40%m and 40%, respectively.

Regarding the maximum values, the control group obtained 100% of efficacy, while for the experimental groups this value was 90%, 100%, and 100%, respectively. Completing our analysis, for CG vs. EG1, the Mann-Whitney test revealed a significant difference between the efficacy value of each group (*p-value*=0.0155), whereas for CG vs. EG2, no significant difference was found (*p-value*=0.0536) and for CG vs. EG1+EG2, the test pointed to a significant difference (*p-value*=0.0155). From this perspective, we conclude that it is possible to infer that the students who learned the basic concepts of software testing in the traditional approach were more effective to answer correctly the questions than the students who learned with the gamified approach.

Another quiz was applied after teaching the functional testing criteria. The *box-plots* of Figure 10 depict the participants' efficacy in answering correctly the questions of Quiz 2. Similarly to Figure 10, in Figure 11 the left-hand chart includes results for comparisons CG vs. EG1 and CG vs. EG2, whereas the right-hand chart brings results for CG vs. EG1+EG2. We can see that the median values for CG, EG1, EG2, and EG1+EG2 were 80%, 70%, 85%, and 80%, respectively. In this analysis, we observed an outlier in the control group, which might indicate that a student did not learn the testing criteria satisfactorily. For CG vs. EG1, the Mann-Whitney test revealed a significant difference between the efficacy value of each group (*p-value*=0.0140), whereas for CG vs. EG2, no significant difference was found (*p-value*=0.3614), likewise for CG vs. EG1+EG2 (*p-value*=0.1802). From this perspective, different from the results regarding Quiz 1, we conclude that none of the learning approaches stood out in terms of efficacy.

Regarding **RQ2** (*Will the students' performance who learn in a gamified approach be higher than the students' performance who learn in a traditional approach?*), we performed three evaluations: (i) knowledge increment from pre-test to post-test, (ii) students' performance in Quiz 1, and (iii) students' performance in Quiz 2. On two out of three analyses—namely, (i) and (iii)—we did not find a significant difference in favor of a particular group. Only in the analysis (ii) we found out that the control group performed better than the experimental groups in two situations (CG vs. EG1, and CG vs. EG1+EG2). All in all, we conclude that our results do not support the rejection of the null hypothesis (H2.0).)

4.3 Students Feedback

For this section, we present in Table 5 the questions (and scores) of the *Evaluating the Short Course* questionnaire. It was intended to gather students' feedback regarding the issues on software testing education (shown in Table 2), in which we also pointed out possible solutions, expected students' behavior, and the gamified activity that might contribute to achieve such behavior.

From the problems presented in Table 2, we thought about the possible consequences that each issue could cause. For example, if the traditional approach has been inefficient, a possible consequence would be student distraction. Thus, we proposed to use gamification to make the classes more enjoyable, thus attracting students' attention to the concepts to which they were being exposed. In this way, question #1 of the questionnaire aimed to verify if this objective was reached. The result was that the gamified approach applied in the experimental groups (EG1, and EG2) helped to attract more students' attention than the traditional approach (CG). The combined comparison (EG1+EG2) groups also demonstrate a better assessment in this regard.

Another result was that the use of gamification made the class funnier, which suggests that the use of gamification was a positive solution to the problem of *unattractive classes* (Table 2). This is reflected by results for question #2 and by the researcher's point of view (see Section 4.4 for more details). Besides that, question #5 refers to the expected behavior of *collaboration*. While it was hoped that gamification could motivate student collaboration, especially when working in teams during the final exercise, our results suggest the opposite: the control group (CG) was more motivated to work as a team than students from the experimental groups (EG1, EG2, and EG1+EG2). Furthermore, question #9 indicates that the control group was also more motivated to use functional criteria in future projects (even if not required) than the experimental groups.

As seen in Table 5, the other questions had similar scores between the experimental and control groups. These results raise two possibilities: (i) the created gamified approach was not sufficient to reduce the challenges faced when teaching software testing, or (ii) simply applying quizzes and a hands-on exercise (both without game elements) after each taught concept was as good as using an alternative to the traditional teaching approach (in this case, the gamified approach). Thus, it is suggested that further investigation must be conducted to obtain answers to these questions.

Another questionnaire (*Evaluating the Platform*) was also applied to the two experimental groups (EG1, and EG2) to get students' opinions regarding the *Bug Hunter* platform. In one question, was asked *"What did you like most about the Bug Hunter platform?"*. Among the answers, some mentioned: the competitive aspect thus generating a learning stimulus; immediate feedback; easiness to understand the environment; real-time use (during class); the evolution of the avatar; and the possibility to follow the development of each colleague. When asked what was least pleasant, the answers were difficulty in buying items for the avatar, few reward options for buying with virtual currencies, and a lot of duplicate information (*e.g.* redundant tabs). Students were also asked about which game elements were the most motivating on the platform, and the three most voted were *duel*, *leaderboard*, and, tied, *achievement*, and *quest*.

4.4 Researchers' Observations

During the experimental sessions, the researcher observed students' behavior, comments, and facial expressions, which we present in the following:

Concentration and focus: We observed that participants from the two experimental groups (i.e., EG1 and EG2) often got sidetracked by Bug Hunter. We believe that some of the elements in our gamified environment might have negatively impacted the participants' ability to remain focused on learning activities, which in turn might have been the reason for their (non-significant) inferior performance in comparison to the control group. Additionally, we noticed that some exchanges of ideas among the experimental participants in EG1 constantly went off-topic: *e.g.* some of the students were con-

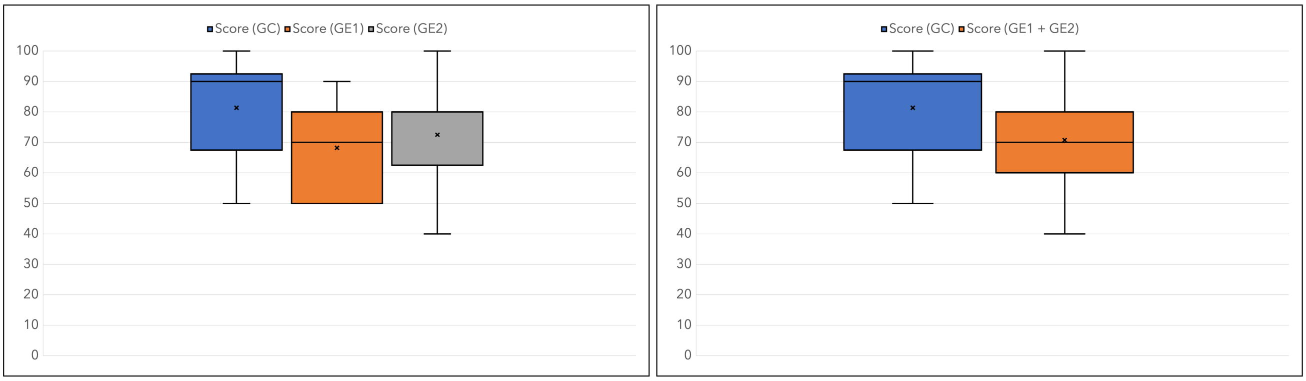


Figure 10. Students' efficacy in Quiz 1.

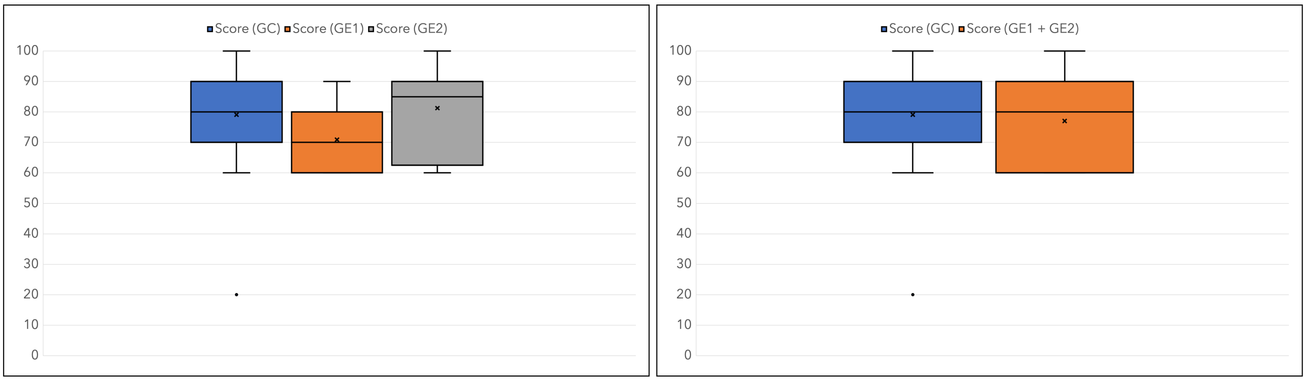


Figure 11. Students' efficacy in Quiz 2.

Table 5. Results of the students' feedback form.

Questions	CG	EG1	EG2	EG1+EG2
1. The manner as software testing concepts were taught attracted my attention	5,30	6,55	5,50	5,93
2. The manner as software testing was taught made the learning more enjoyable	4,65	6,64	5,38	5,89
3. The quizzes reinforced the learned content	6,35	6,73	6,00	6,30
4. The practical exercise reinforced the learned content	6,45	6,36	5,63	5,93
5. The exercise performed in groups motivated me to work in team collaborating with my classmates	6,50	5,55	5,50	5,52
6. The difficulty level in the quizzes and in the practical exercise was coherent to the taught content	6,05	6,82	6,63	6,70
7. I still remember the content that was taught in the short course	5,85	5,55	5,81	5,70
8. After this short course, I am confident that I can perform software testing activities using the functional testing criteria	5,25	5,36	4,56	4,89
9. I intend to perform testing activities in my future projects (even if not required)	6,55	5,91	5,06	5,41
10. After this short course, I understood the importance of the software testing activities	7,00	6,82	6,25	6,48

cerned about non-experiment-related assignments that were due later on the same week.

Participation: Although subjects were encouraged to actively participate during the learning process, only four subjects in EG1 participated (*e.g.* asked questions, volunteered answers, and contributed to the discussion) during the early systematic exposition of basic software testing concepts. Afterward, however, during the first competitive quiz, all participants seemed to be actively engaged in the learning process. As for the participants in EG2, we observed less willingness to participate during the early introduction of testing concepts as well as when we ran the competitive quizzes. Conversely, the participants in the control group seemed more actively engaged throughout the learning process. We conjecture that the participants in the control group welcomed the whole experiment process and overall were more likely to stay engaged because they seldom have had the chance to take part in studies of this type.

Collaboration: Bug Hunter includes a forum to allow for collaborative learning and provide a space for students to engage with each other while learning. Surprisingly, no participant in either group took advantage of such a feature. We also expected some sort of collaboration among participants during the conduction of the final learning activity in which the participants were divided into groups. Then, in a similar fashion to what happens in practice (*i.e.* industrial settings), participants were asked to come up with test cases, implement, and execute them. However, it turns out that this activity was regarded as one of the least enjoyable by the participants in EG1 and EG2. Conversely, participants in the control group rated this activity as the most enjoyable throughout the experimental session.

Putting the recently learned concepts in practice: During the final learning activity, we expected the participants to employ Boundary-value Analysis and Equivalence Partitioning to create test cases. It turns out only six out of seven teams from the control group and three out of four teams from EG2 used both criteria. EG1 performed quite poorly in this sense: only one out of four teams used both criteria, one team used only one criterion, a team used no criterion while devising test cases, and one team failed to complete the learning activity (participants in this group claimed to have faced many problems while carrying out the activity). It is worth mentioning that participants in the control group frequently asked help from the lecturer (who is one of the authors of this paper), whereas participants in EG1 and EG2 claimed that they seldom asked for help because they felt driven to finish the activity by themselves.

Participants' comments: All experimental groups reported they felt overwhelmed because in their opinion too much content was covered in a short period. Another negative aspect observed by the subjects in the control group is that many exchanges of ideas among them ended up leading them off-topic during learning activities. The use of gamification is bolstered by the positive remarks made by some participants in EG1, according to them "all lectures should be as dynamic as the ones that took place during the experimental sessions". Furthermore, participants in EG2 stated that the quizzes were very helpful in reinforcing the information presented. Participants in EG2 also confirmed that given the hands-on nature of the final learning activity, they were able to get a better grasp of the concepts and how they might be employed in professional practice.

5 Threats and Limitations

The threats to validity we identified are classified in four categories: internal, external, construct, and conclusion (Wohlin et al., 2012). When possible, actions taken to mitigate the threats are also outlined.

Internal validity: The identified threats were: (1) researcher influence; (2) questions and requirements used in the quizzes and final challenge; (3) fatigue. Regarding (1), in order to avoid the researcher influence in the results, the same researcher ran the experimental sessions, and the analysis was carried out by, and discussed with the other researchers. Regarding (2), the quizzes only included questions about the taught concepts, and the requirements for creating the test cases were the ones frequently used in software engineering classes. Regarding (3), although we were aware that four hours of an experimental session might cause fatigue, we had limitations (next discussed) from one of the IHEs. Nevertheless, students in such a context were used to have everyday four-hour classes along the academic semester.

External validity: The identified threats were: (1) previous knowledge; (2) group formation; (3) used questions in the quizzes and requirements in the final challenge. Regarding (1), participants from IHEs B and C (cf. described in Section 3.4) have already had software engineering and software quality classes with testing topics. Regarding (2), students chose their groups aiming at increasing their engagement and collaboration. Regarding (3), as expected due to time constraints, the used questions and requirements do not represent all existing scenarios.

Construct validity: The threat concerns the metrics used to analyze students' motivation and performance. Regarding motivation, we used the Intrinsic Motivation Inventory (IMI), proposed and applied by Deci and Ryan (2011) to specifically measure intrinsic motivation. Regarding performance, we gathered descriptive statistics and, additionally, applied the largely adopted Shapiro-Wilk and Mann-Whitney tests to compare the differences between our two independent groups. It is worth mentioning that the last experimental group was added with the purpose of balancing the experimental groups, given that in our previous analysis (reported in our previous study) the control group consisted of data produced by 22 subjects while the experimental group was comprised of only 11 subjects. We cannot rule out the potential threats associated with adding more participants to the experimental group in this fashion.

Conclusion: Our sample is small and homogeneous (comprised only of undergraduate students), as described in Section 3.4. Moreover, and unfortunately, 7 (out of 18) students from the experimental group of IHE B (*i.e.* EG1) did not complete the teaching sessions, and hence the samples were not balanced by only considering the control group (with 22 subjects) and EG1. To ameliorate this situation, we performed an extra gamified session in IHE D (*i.e.* EG2) (with 16 subjects that completed all assignments), and this allowed us to run two additional analyses. We highlight the difficulty to get middle or final year students (when software testing courses are usually taught) to take part in this sort of experiment. The main hurdle we faced was the slow process for getting approval from ethics committees (it may take up to a couple of months), and the number of students enrolled in advanced software engineering courses is low when compared to freshmen, sophomore, and junior Computer Science students. It is worth mentioning that our samples were purposive, *i.e.* we selected the subjects according to their capacity to provide information that is relevant to the phenomenon under investigation. Thus, although these samples tend to be small, by including information-rich cases (*i.e.* subjects) they allow for in-depth analysis. As a result, we argue that our relatively small samples add important information to the discussion: we believe our gamified approach and results thereof can be seen as a first step towards more robust experiments with larger number of subjects.[14] That said, we cannot rule out the threat that the results could have been different if a different sample (or more samples) had been selected.

We also identified some limitations of our research. We consider that the ideal scenario would be performing the experiments during the entire academic semester to approximate to the real scenario. However, only 4 hours were provided in each institution where the sessions were executed.

[14] All experimental material is available at `https://bit.ly/2MCvVY1`.

Consequently, the experiment was shaped like a short course to present the most important concepts, and focusing on teaching more deeply functional testing with practical examples.

Regarding the experimental sessions, the control group (CG) and the second experimental group (EG2) had two 2-hour sessions on the same day, with an interval of 20 minutes. The first experimental group, differently, had two 2-hour sessions split in two consecutive days. Although there is a recommendation of sessions no longer than two hours to avoid fatigue (Siegmund and Schumann, 2015), the main reason for not splitting the CG sessions in two days was that students from that group were attending classes of the last week of the academic semester, and there was no other available day for running the experiment. As a last note, an interesting point is that even with the possibility of fatigue, the control group had performance similar to that of the experimental groups.

6 Conclusion and Future Work

Given the importance of testing activities and the challenges faced in the educational context, we followed a set of steps to design a gamification approach aiming at mitigating them. More specifically, to assess the impact of gamification in software testing education, we developed a gamified approach with a supporting platform that enabled us to conduct five experimental sessions. Two of the sessions were pilot studies for approach and platform refinement purposes, and the other three yielded results to address two research questions that regarded students' motivation and performance during the classes when compared with a traditional teaching approach.

The results of our experiments would seem to indicate that students that learned the content through the traditional approach felt slightly more motivated than the students who learned with the gamified approach. We surmise that gamification has the potential to help increase students' motivation, but students that take traditional (non-gamified) classes may have intrinsic motives for learning (they engage in learning activities not for external reward but because they find these activities interesting and gratifying). As for performance, the results of our quantitative analysis indicate that there were no significant differences between the control and experimental group. We also observed that building a gamified environment is a complex and incremental process, especially in the definition phase of a reward system and the ranges of scores and levels, which are related to the game mechanics and dynamics.

We highlight that the experience of having used this alternative approach is considered positive, as it provided a more enjoyable and funny environment, both from the researcher's and students' point of view. In fact, feedback questionnaires showed that gamification helped us to attract students' attention and make the class more engaging. Additionally, students in the experimental group seemed to be able to stay on task longer and found the learning experience gratifying. Therefore, we believe that smoothly turning to gamification may lead to motivation and performance improvements in the medium- and long-term.

Finally, we emphasize this research is our first foray into investigating the impact of gamification on undergraduate students. Despite the great amount of (academical and behavioral) engagement and attention the experimental group showed during the experiment, we need more evidence to draw a proper conclusion regarding the best way to teach software testing-related knowledge. Summing up, the evidence is still lacking, but we believe that after incorporating all the feedback we got from the experiments into our gamified platform we can achieve a more enjoyable teaching and learning platform. As for the current version of our gamified platform, it is worth mentioning that it is not centered around Kahoot!. Rather, Kahoot! has been used to foster competition among the participants. As mentioned, this first set of experiments were conducted in hopes of identifying possible improvements we could make to our gamified platform and only after identifying such issues we plan on coming up with strategies to remedy them. In other words, at first, we developed a sort of "proof of concept" implementation of our platform, as a Google spreadsheet and a set of Kahoot! quizzes, which we intend to improve based on the feedback we got from participants and our own experience interacting with the platform.

Some other lessons learned are that applying quizzes and hands-on exercises after teaching the fundamental concepts might foster students' engagement. Moreover, the implementation of a gamified environment is challenging because several elements must be taken into account, *e.g.* the students' profile and ease of use (for instructor and students).

As future work, to obtain more evidence of the effectiveness of our approach, we intend to carry out more experiments involving more participants. As mentioned, from the outset, our approach was tailored to undergraduate students. Owing to this fact, first, we set out to evaluate the impact of our gamified approach on undergraduate students. However, we surmise that investigating the impact our approach might have on different samples can add further value to our research and help us further improve our approach. Therefore, in the future, we will also carry out follow-up experiments to probe into the performance of different experimental subjects (*e.g.* graduate students and practitioners) when exposed to software testing concepts through our gamified approach. Moreover, we also plan on comparing how different experimental groups fare in comparison to one another.

Acknowledgements

This work was financed by CNPq (grants #167513/2017-6, #306310/2016-3, and #312922/2018-3), State of São Paulo Research Foundation - FAPESP (grants #2017/10941-8 and #2013/07375-0), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, and Federal University of São Carlos.

References

Ammann, P. and Offutt, A. J. (2016). *Introduction to Software Testing*. Cambridge University Press, Cambridge, UK, 2nd edition.

Anderson, P. E., Nash, T., and McCauley, R. (2015). Facilitating Programming Success in Data Science Courses Through Gamified Scaffolding and Learn2Mine. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE)*, pages 99–104, Vilnius, Lithuania. ACM.

Bartle, R. (1996). Hearts, Clubs, Diamonds, Spades: Players Who Suit MUDs. *Journal of MUD Research*.

Bell, J., Sheth, S., and Kaiser, G. (2011). Secret Ninja Testing with HALO Software Engineering. In *Proceedings of the 4th International Workshop on Social Software Engineering (SSE)*, pages 43–47, Szeged, Hungary. ACM.

Benitti, F. B. V. (2017). As a teacher, i want to know what to teach in requirements engineering so that professionals can be better prepared. In *Proceedings of the 31st Brazilian Symposium on Software Engineering*, pages 318–327, New York, NY, USA. ACM.

Capgemini, Micro Focus, and Sogeti (2018). World Quality Report 2018-19. Tech. Report 10th edition, Capgemini, Micro Focus and Sogeti. `https://www.capgemini.com/service/world-quality-report-2018-19/` – accessed on February, 2020.

Cheiran, J. F. P., Rodrigues, E. M., Carvalho, E. L. S., and Silva, J. P. S. (2017). Problem-based learning to align theory and practice in software testing teaching. In *Proceedings of the 31st Brazilian Symposium on Software Engineering*, pages 328–337, New York, NY, USA. ACM.

Clegg, B. S., Rojas, J. M., and Fraser, G. (2017). Teaching Software Testing Concepts Using a Mutation Testing Game. In *Proceedings of the 39th International Conference on Software Engineering (ICSE) – Software Engineering and Education Track*, pages 33–36, Buenos Aires, Argentina. IEEE.

Costa, I. and Oliveira, S. (2019). A systematic strategy to teaching of exploratory testing using gamification. In *Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering*, ENASE 2019, pages 307–314. SCITEPRESS.

Dal Sasso, T., Mocci, A., Lanza, M., and Mastrodicasa, E. (2017). How to Gamify Software Engineering. In *Proceedings of the IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 261–271, Klagenfurt, Austria. IEEE.

Deak, A., Stålhane, T., and Sindre, G. (2016). Challenges and Strategies for Motivating Software Testing Personnel. *Information and Software Technology*, 73:1–15.

Deci, E. L. and Ryan, R. M. (2011). Levels of analysis, regnant causes of behavior and well-being: The role of psychological needs. *Psychological Inquiry*, 22(1):17–22.

DeMillo, R. A., Lipton, R. J., and Sayward, F. G. (1978). Hints on test data selection: Help for the practicing programmer. *Computer*, 11(4):34–41.

Deterding, S., Dixon, D., Khaled, R., and Nacke, L. (2011). From game design elements to gamefulness: Defining ”gamification”. In *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*, pages 9–15, New York, NY, USA. ACM.

Dubois, D. J. and Tamburrelli, G. (2013). Understanding Gamification Mechanisms for Software Development. In *Proceedings of the 9th Joint Meeting on Foundations of Software Engineering (ESEC/FSE)*, pages 659–662, Saint Petersburg, Russia. ACM.

Elgrably, I. S. and Oliveira, S. R. B. (2018). Gamification and evaluation of the use the agile tests in software quality subjects: The application of case studies. In *Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering*, ENASE 2018, pages 416–423. SCITEPRESS.

Fraser, G., Gambi, A., Kreis, M., and Rojas, J. M. (2019). Gamifying a software testing course with code defenders. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE)*, pages 571–577, Minneapolis, MN, USA. ACM.

García, F., Pedreira, O., Piattini, M., Cerdeira-Pena, A., and Penabad, M. (2017). A Framework for Gamification in Software Engineering. *Journal of Systems and Software*, 132:21–40.

Harrold, M. J. (2000). Testing: A Roadmap. In *Proceedings of the Conference on Future of Software Engineering (FoSE) - held in conjunction with ICSE*, pages 61–72, Limerick, Ireland. ACM Press.

Houshmand, M. and Paydar, S. (2017). Emville: A gamification-based approach to address the equivalent mutant problem. In *2017 7th International Conference on Computer and Knowledge Engineering (ICCKE)*, pages 326–332.

Jesus, G. M., Ferrari, F. C., Porto, D. P., and Fabbri, S. C. P. F. (2018). Gamification in software testing: A characterization study. In *Proceedings of the III Brazilian Symposium on Systematic and Automated Software Testing (SAST)*, pages 39–48, New York, NY, USA. ACM.

Jesus, G. M., Paschoal, L. N., Ferrari, F. C., and Souza, S. R. S. (2019). Is It Worth Using Gamification on Software Testing Education?: An Experience Report. In *Proceedings of the 18th Brazilian Symposium on Software Quality (SBQS)*, pages 178–187, Fortaleza, CE, Brazil. ACM.

Lyra, K. T., Isotani, S., Reis, R. C. D., Marques, L. B., Pedro, L. Z., Jaques, P. A., and Bitencourt, I. I. (2016). Infographics or graphics+text: Which material is best for robust learning? In *Proceedings of the 16th International Conference on Advanced Learning Technologies (ICALT)*, pages 366–370, Austin, TX, USA.

Marabesi, M. and Silveira, I. F. (2019). Towards a gamified tool to improve unit test teaching. In *2019 XIV Latin American Conference on Learning Technologies (LACLO)*, pages 12–19.

Myers, G. J., Sandler, C., and Badgett, T. (2011). *The Art of Software Testing*. ITPro collection. Wiley, John Wiley & Sons, 2011.

Paschoal, L. N., Oliveira, B. R. N., Nakagawa, E. Y., and Souza, S. R. S. (2019). Can we use the flipped classroom model to teach black-box testing to computer students? In *Proceedings of the XVIII Brazilian Symposium on Software Quality*, SBQS'19, page 158–167, New York, NY, USA. Association for Computing Machinery.

Paschoal, L. N., Silva, L. R., and Souza, S. R. S. (2017). Flipped classroom approach compared with the traditional teaching model: An empirical investigation in the context of software testing (in Portuguese). In *Proceedings of the 28th Brazilian Symposium on Informatics in Education*. Brazilian Computer Society.

Pedro, L. Z. (2016). The Use of Gamification in Virtual Learning Environment to Reduce the Problem of Externalization of Undesirable Behaviors (in Portuguese). Master's thesis, University of São Paulo, São Carlos, SP, Brazil.

Pedro, L. Z., Zem-Lopes, A. M., Prates, B. G., Vassileva, J., and Isotani, S. (2015). Does Gamification Work for Boys and Girls? An Exploratory Study with a Virtual Learning Environment. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing (SAC)*, pages 214–219, Salamanca, Spain. ACM.

Pinto, F. S. and Silva, P. C. (2017). Gamification applied for software engineering teaching-learning process. In *Proceedings of the 31st Brazilian Symposium on Software Engineering (SBES)*, pages 299–307, New York, NY, USA. ACM.

Rojas, J. M. and Fraser, G. (2016a). Code Defenders: A Mutation Testing Game. In *Proceedings of the 11th International Workshop on Mutation Analysis (Mutation)*, pages 162–167, Chicago, IL, USA. IEEE.

Rojas, J. M. and Fraser, G. (2016b). Teaching Mutation Testing using Gamification. In *Proceedings of the European Conference of Software Engineering Education (ECSEE)*, pages 1–5, Seeon Monastery, Germany. Shaker Publishing.

Rojas, J. M., White, T., Clegg, B., and Fraser, G. (2017). Code Defenders: Crowdsourcing effective tests and subtle mutants with a mutation testing game. In *Proceedings of the of the 39th International Conference on Software Engineering (ICSE)*, pages 677–688, Buenos Aires, Argentina. IEEE.

Sheth, S., Bell, J., and Kaiser, G. (2012). Increasing Student Engagement in Software Engineering with Gamification. Tech. Report CUCS-018-12, Columbia University, New York, NY, USA.

Siegmund, J. and Schumann, J. (2015). Confounding parameters on program comprehension: a literature survey. *Empirical Software Engineering*, 20(4):1159–1192.

Smith, J., Tessler, J., Kramer, E., and Lin, C. (2012). Using peer review to teach software testing. In *Proceedings of the 9th Annual International Conference on International Computing Education Research (ICER)*, pages 93–98, New York, NY, USA. ACM.

The Joint Task Force on Computing Curricula (2015). Software Enginering 2014 – Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. Technical report, IEEE Computer Society and Association for Computing Machinery.

Valle, P. H. D., Toda, A. M., Barbosa, E. F., and Maldonado, J. C. (2017). Educational Games: A Contribution to Software Testing Education. In *2017 IEEE Frontiers in Education Conference (FIE)*, pages 1–8, Indianapolis, IN, USA. IEEE.

Werbach, K. and Hunter, D. (2012). *For the Win: How Game Thinking Can Revolutionize Your Business*. Wharton Digital.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B., and Wesslén, A. (2012). *Experimentation in Software Engineering*. Computer Science. Springer Berlin Heidelberg.

Yujian Fu, P. E. and Clarke, P. J. (2016). Gamification-Based Cyber-Enabled Learning Environment of Software Testing. In *Proceedings of ASEE's 123rd Annual Conference & Exposition (ASEE)*, pages 1–15, New Orleans, LA, USA. American Society for Engineering Education.