Is It Worth Using Gamification on Software Testing Education? An Experience Report

Gabriela Martins de Jesus
DC/UFSCar - Brazil
gabriela.jesus@ufscar.br

Fabiano Cutigi Ferrari
DC/UFSCar - Brazil
fcferrari@ufscar.br

Leo Natan Paschoal
ICMC/USP - Brazil
paschoalln@usp.br

Simone R. S. Souza
ICMC/USP - Brazil
srocio@icmc.usp.br

ABSTRACT

Context: Testing is essential to improve the quality of software products. Despite that, it is not a subject that students are motivated to learn. Gamification is a promising way to address issues in software testing education; it is used to insert game elements in educational contexts aiming to increase students' motivation and performance. *Objective:* Reporting on results of an experimental study designed to assess the impact of gamification on software testing education. *Method:* We carried out the experimental sessions with undergraduate students from three Brazilian institutions. They have been taught basic testing concepts and functional testing. Moreover, the experimental group used a gamified platform that included 10 game elements to increase students' motivation and performance, attract their attention, and instigate their participation, collaboration, and competitiveness. *Results:* The experimental group was more motivated than the control group. Regarding performance, the experimental group had an equivalent performance compared to the control group in the pre- and post- tests. However, we observed a trend for the control group to reach a higher performance, as the results from the quiz activities showed significant difference. *Conclusion:* Although several studies concluded that gamification has the potential to lead to positive outcomes, we reached both positive and negative results. So that, we present our findings and answer whether, in our point of view, it is worth using gamification on software testing education.

CCS CONCEPTS

• **Software and its engineering** → **Software testing and debugging**.

SBQS'19, October 28-November 1, 2019, Fortaleza, Brazil

ACM ISBN 978-1-4503-7282-4/19/10…$15.00
https://doi.org/10.1145/3364641.3364661

KEYWORDS

Education, gamification; game elements; software testing; quality assurance; experience report.

ACM Reference Format:
Gabriela Martins de Jesus, Fabiano Cutigi Ferrari, Leo Natan Paschoal, and Simone R. S. Souza. 2019. Is It Worth Using Gamification on Software Testing Education? An Experience Report. In *XVIII Brazilian Symposium on Software Quality (SBQS'19), October 28-November 1, 2019, Fortaleza, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3364641.3364661

1 INTRODUCTION

Testing aims to improve the quality of the products. Its main objective is revealing faults [14], and it is mandatory throughout the development process. However, there is a lack of qualified professionals the testing area [24]. Beyond the low-motivating and destructive nature of tests [14], there are several other challenges facing this area, specially in the educational context. Among them, there is a supposedly inefficiency of traditional[1] teaching approach [23]. To mitigate the challenges, some alternative approaches have been proposed to stimulate desired behavior and increase students' motivation and performance, such as serious games [6, 18], flipped classroom [15], and gamification [1, 3, 21, 29].

Gamification, defined as the use of game elements in non-game contexts [9], has become an increasingly investigated approach to reach benefits such as the increase of users' loyalty, motivation, engagement, collaboration, and productivity in different scenarios. To better understand how gamification has been explored in software testing context, we rely on a systematic mapping carried out by Jesus et al. [12], in which the authors categorized the selected studies in six perspectives. All analyzed studies reported positive outcomes. Despite that, open questions are: "*Is it worth using gamification in all contexts?*"; "*Given the observed benefits, is it worth it to be implemented?*". Instigated for the positive outcomes reported, we designed a gamification approach following a set of steps, and developed a gamified environment to investigate if gamification would help to increase the students' motivation and performance in the educational context of software testing. We highlight that it is possible to adapt our approach in other educational contexts or even in industry.

[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.

We ran four experimental sessions in which we taught the basic testing concepts, and Functional Testing and its main criteria (Equivalence Partitioning, and Boundary Value Analysis). The subjects were undergraduate students of computing related programs from three Brazilian institutions. The main benefit for the participants was the opportunity to learn more deeply the most used testing technique in industry.

Despite the positive outcomes with the use of gamification reported in previous research, we achieved some different results. Overall, we observed the students' motivation who learned in a gamified approach was higher than the students' motivation who learned in a traditional approach. On the other hand, the students' performance who learned in a gamified approach was lower than the students' performance who learned in a traditional approach.

The paper is organized as follows: Section 2 provides key concepts of software testing and gamification, and summarizes studies related to gamification applied to education, particularly for software testing. Section 3 presents the study setup, and Section 4 reports and discusses the study results. Section 5 lists some threats to validity that we identified in our research, and conclude this paper on Section 6.

2 CONCEPTS AND RELATED WORK

This section discusses how gamification has been investigated for education purposes, particularly for software testing.

2.1 Gamification in Education

Approaches such as serious games [6], flipped classroom [15], and gamification [1, 3, 21, 29] have been used in educational contexts to stimulate and increase students' motivation and performance. Focusing on the gamification, 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 the game elements 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 the education in data science.

Pedro [16], 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 the undesirable behavior decreased in the experimental group. Other 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] http://learn2mine.appspot.com/ - accessed on July, 2019

2.2 Gamification in Software Testing

From the studies selected by Jesus et al. [12] in their systematic mapping, 3 of them regard CODE DEFENDERS[3], a gamified system for mutation testing education [6, 18, 19]. Among the pursued goals, the CODE DEFENDERS authors aimed to improve students' performance, and increase their enjoyment, engagement, and motivation. Beyond the PBL triad, they also used the duels and team game elements. In one study [18], CODE DEFENDERS was presented as an early prototype, while in the other two [6, 19], the authors described an approach to provide practical experience for teaching mutation testing. Even though the authors presented their studies' hypotheses, no results were discussed, nor evaluation was performed.

Yujian Fu and Clarke [29] 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. One of the points evaluated by the authors was if there was a relationship between the students' grade and their points earned in the gamified environment. As 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 the software testing learning.

3 STUDY SETUP

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. To achieve this goal, we conducted four experimental sessions to address the following research questions shown on Table 1:

Table 1: Research Questions and Hypotheses

Question	Null Hypothesis	Alt. 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** The student's motivation who learn in a gamified approach is equivalent to the student's motivation who learn in a traditional approach.	**H1.1** The student's motivation who learn in a gamified approach is different to the student's 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** The student's performance who learn in a gamified approach is equivalent to the student's performance who learn in a traditional approach.	**H2.1.** The student's performance who learn in a gamified approach is different to the student's performance who learn in a traditional approach.

As 2 out of 4 experimental sessions were pilot-studies (*i.e.* one session for each group: experimental and control), we will not describe their corresponding results.

[3] http://code-defenders.org/ - last accessed on July, 2019

[4] http://wrestt.cis.fiu.edu/ – Accessed on July 2019

3.1 Experimental Material

To answer the research questions, we analyzed if the gamified approach to teach 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[5].

The students' motivation was measured through the *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 [8], and comprehend 22 short questions with a 7-value Likert scale. This questions are meant to measure the motivation in four categories: interest/enjoyment, perceived competence, perceived choice, and felt pressure and tension.

On the other hand, student's performance was measured and tracked through two quizzes. The first one encompass questions related to basic concepts of software testing and its terminologies, such as quality assurance, software testing process, levels, and techniques. The second quiz encompass 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 student's previous knowledge about software testing, and to check (after the experiment) if they learned what is the software testing objective, its importance, and what is the Functional Testing technique. In the second analysis, we applied a practical exercise in which the students were 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.

3.2 Subjects

To perform this study, we invited undergraduate students from three Brazilian Institutions of Higher Education (IHE): *A, B*, and *C*[6]. As shown on Table 2, the sample comprised 53 students, in which 13 were from IHE *A*, 18 from IHE *B*, e 22 students were from IHE *C*.

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

Subjects from IHE *A* and *B* were Information Systems (IS) undergraduate students, while subjects from IHE *C* were System Analysis and Development (SAD) undergraduate students. All subjects were either middle and final year students.

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

[6]To keep anonymity, we refer to the host institutions as *A*, *B* and *C*.

Table 2: Demographic data.

Approach	IHE	# subjects	# considered activities	# considered 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
TOTAL		53	45	44	

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*, and all of the 22 students from the control group at IHE *C*.

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 sections, 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.

The participants were asked if they had already studied software testing. According to them, theoretical concepts were taught as a topic in the software engineering discipline at IHE *B*, and in the software quality discipline at IHE *C*.

Notice that all participants were aware they were voluntaries 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.

3.3 Experimental Sessions

In the last semester of 2018, the same researcher conducted the four experimental sessions. At IHE *A*, the researcher run two pilot sessions with duration of four hours each. 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, the questionnaires, quizzes, practical activity, and the time of execution of 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 contend taught. In the practical activity, instead of allowing the students to chose any e-commerce website for their own, we defined a specific one[7] to be tested by them. Aiming to minimize the researcher efforts during the class, we automated the gamified platform to calculate

[7] https://www.casasbahia.com.br/ - accessed on July, 2019.

and reward the students with points, badges, levels, ranking, avatar upgrades, etc. Finally, the questionnaires were also refined to contain questions more related to the issues on software testing education we were investigating, and if 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 their students to participate in the experiment, and provided his two classes in two different days in a week (four hours total).

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

3.4 Gamification Approach

Aiming at raising some of the challenges the software testing education faces of, we performed an *ad hoc* search on Google Scholar[8] using terms such as *"problems in software testing education"*. As the results were vast and broad, we started with a master thesis of Valle [25] to extract the problems and other studies [20, 23, 26] cited by Valle [25] that also mention issues related to software testing education. After that, we read other cited papers [4, 5, 17], and then extracted the difficulties the authors mentioned. As result of our *ad hoc* search, we summarized on Table 3 the problems we identified, and the possible solutions for each one.

Some solutions that we present on Table 3 do not directly involve gamification. However, as shown on the *Gamified Activities* column, we can use this alternative approach by inserting game elements into the activities with the goal of reaching the expected students' behavior, while we try to solve the identified problems. Thus, although gamification usually focuses on addressing punctual issues such as turning the classes more enjoyable or boosting adoption, it also can 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 by Jesus et al. [12] to understand how gamification has been explored to support software testing. Jesus et al. illustrated on a bubble chart the number of studies that relate the used game elements to the goals pursued with gamification. Based on this, we defined the problems we expected to solve, and selected the set of game elements to insert into our gamified environment.

The last analysis was the following six steps to gamification (known as *6D's*) proposed by Werbach and Hunter [27]:

- 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 [2]);
- 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 Kahoot![9] 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.

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 [7, 10, 11], or other studies that proposed gamified tools for education purposes [1, 29], none of those completely fit to our needs. Thus, we followed and adapted those authors' idea to create our own rules to reward the students for their achievements with 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 the level 1, and 100 in the last level. However, we thought that each reward would apply a short score for an 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 shown in Table 4.

We finally developed a gamified prototype called *Bug Hunter*[10] with 10 game elements mapped by Jesus et al. [12] to support our goals of increasing students' engagement, motivation, and enjoyment. The game elements we inserted into Bug Hunter were: *Achievement, avatar, badge, duel, leaderboard, level, points, quest, team, virtual goods* Specifically, the main behavior we expected from the students was:

- Attention: the students should pay attention to the explanation during the class. The competitive quizzes applied, for example, were an 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 distracted. 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 sum up the needed value to level up.
- Active participation: the students should participate by asking and answering questions To encourage them to participate, the research 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

[8] https://scholar.google.com.br/ - accessed on July, 2019.

[9] https://kahoot.com/ - accessed on July, 2019.

[10] https://bit.ly/2Ei8JbW - accessed on July, 2019.

Table 3: Problems in Software Testing Education

Problems	Solutions	Expected Behavior	Gamified Activity
Neglected education addressing only an introduction to software testing subject [26]	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 [4, 25]	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 [25]	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 [23]	Use an alternative approach education	Students focused and participative during classes	Use the gamified approach described in this paper
Unattractive classes [17, 25]	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 [5]	Apply practical exercises right after teaching a concept	Performance higher than 70% in quizzes	Quizzes and practical exercise

Table 4: 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
...				

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 was introduced during the quiz, but in the challenge the expectation was intra-group collaboration to win the challenge; without it, less 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 in groups and had a mission of revealing faults in the system under test.

3.5 Gamified Environment

Bug Hunter was developed to be a gamified environment in which the students could be rewarded and recognized for their achievements; their goal was to reach the fifth (*i.e.* the last) level, and achieve the first place in the ranking. It works as a live feedback mechanism by showing current rewards, achieved goals, and leaderboard, and, as such, by increasing competitiveness among students. Motivated by competition, students may have pursued higher performance during the quiz and challenge. The tool also makes a forum available to increase socialization, as well as some personal affinity through an avatar assigned to each student.

Figure 1 depicts a template of Bug Hunter's Administrative tab. In this tab, the researcher rewarded the students with XP[11] points (in every column in which the line 4 is written "XP", except the column C that sums up the XPs each student earned. Besides that, information were posted in the notification section (from cell *B16*), and the leaderboard (cell *P16*) provided feedback about the current ranking. In this tab, the only information manually entered by the researcher during the experiments were the posts in the notification area, and the attribution of the XPs for students who achieved a goal. The other rewards (badges, virtual goods ($ coins), levels, and avatar upgrades) were automatically calculated and imputed by the platform as the students fulfilled their assignments. For example, when a student accessed the platform for the first time, he/she received a tab with his/her profile, and earns 50 XPs, 1 coin ($), and a *Newbie* badge. After that, if he/she participates of the tutorial, he/she earns other 50 XP points and the *Apprentice* badge. As he/she sums up 100 XPs, he/she moves on to the second level, earns 2 more coins ($), the Hunter badge, and his avatar is upgraded. Also, the avatar *Intern* can be bought as a virtual good for 3$.

In the students tab, each participant had a space such as the one shown in Fig.2. In this tab, the student received feedback about his/her progress, rewards, available virtual goods, ranking, and the notifications sent by the researcher. Also, all information in this tab were linked to the administrative tab and automatically filled in as the conditions were satisfied.

Besides the *Administrative* and the *Students* tab, the Bug Hunter platform has other tabs, such as *Rules*, *Badges*, *Avatar*, *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 *Avatar* tab contains the avatar of each level, including an special avatar that only the winner receives. Besides those, there is also an extra upgrade which could be bought by the participants for $3 virtual coins. 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.

[11] 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	330	3	7	0	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	prata
Student 9	190	2	6	0	50	1	Newbie	50	Apprentice			2	-	90	3	prata
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.

Leaderboard	
1º	Student 1
2º	Student 8

Figure 1: Gamified Environment - Administrative tab.

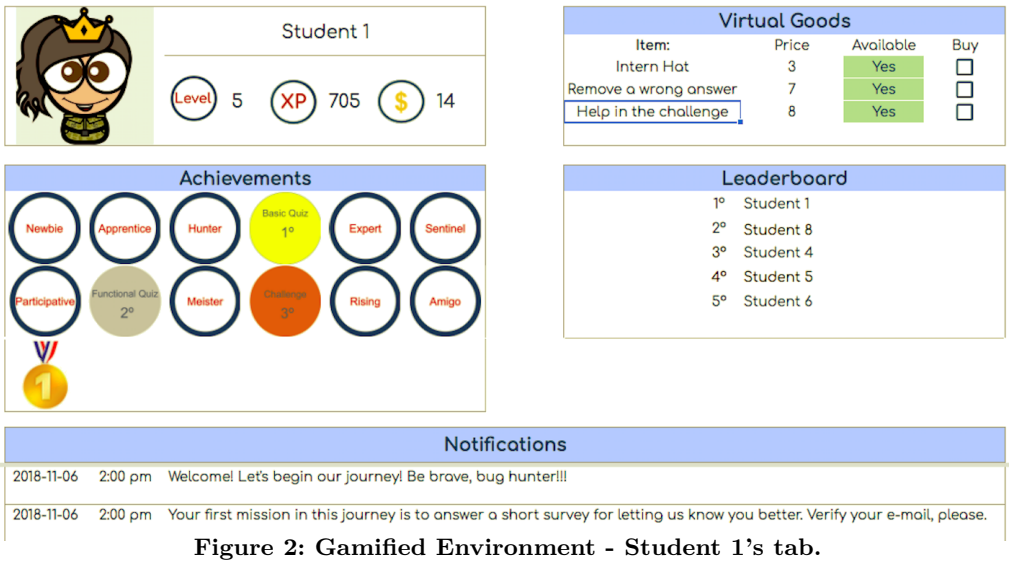

Figure 2: Gamified Environment - Student 1's tab.

3.6 Methodological Approach

We followed the experimental process described by Wohlin, et al. [28]. 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 3. 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 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, 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 the explanations, the researcher encouraged the students to solve an example. After that, a second quiz was applied. It included questions of both set of topics (namely, basic concepts and Functional Testing). For the experimental group, we used the gamified tool called Kahoot! to apply the two quizzes, and the non-gamified tool Google Forms[12] for the control group. We highlight that the exact 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

[12] https://www.google.com/forms/about/ - accessed on July, 2019.

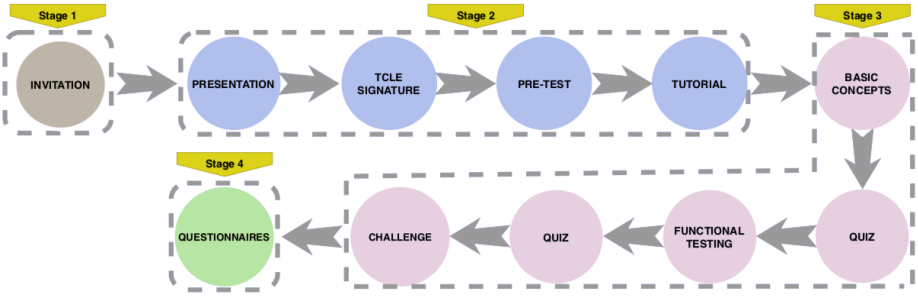

Figure 3: Methodological Approach.

were asked to create and execute a test suite based on three functional requirements presented to them in 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: pre-test and post-test, Intrinsic Motivation Inventory (IMI), and Mini-course Evaluation.

4 RESULTS AND ANALYSIS

This section presents the experiment results. 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, non-parametric test, which in our study compares differences between two independent groups (one control and one experimental groups).

We graded all pre-tests and post-tests for students' performance analysis. We then calculated a Δ value, which represents the increment on knowledge observed after a learning session. Notice that similar approach was adopted in prior research [13] 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.

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

To complement the performance analysis, we used the quizzes to keep track of the students' performance along the teaching sessions. We graded the quizzes and calculated the students' efficacy with respect to 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.

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

We applied the Intrinsic Motivation Inventory (IMI) to compare the students' motivation of both groups, considering 4 aspects, which are: (*a*) *Interest/enjoyment*, (*b*) *Perceived choice*, (*c*) *Perceived competence*, and (*d*) *Pressure/tension*. This questionnaire contains 22 short questions, and a set of them are related to each of the 4 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 3. For example, questions 1, 5, 8, 10, 14, 17, and 20 are related to *Interest/enjoyment*, and (consider that) 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 number of times an answer for a given question was provided, and i is a given student.

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

4.1 Motivation

Figure 4 depicts the results of the Intrinsic Motivation Inventory (IMI) applied to the students from both groups. In the *Interest/enjoyment*, and *Perceived choice* categories, the experimental group had a higher score than the control group. The opposite occurred in the other two categories (namely, *Perceived competence* and *Pressure/tension*). Therefore, we believed that the experimental group enjoyed and had more interest at 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, what is confirmed in the results of the analysis for students' performance.

4.2 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. As shown in Figure 5, while the maximum grade for the control group was 2, the corresponding value for the experimental group was 3 (*i.e.* all answers were correct). The median value reinforces this

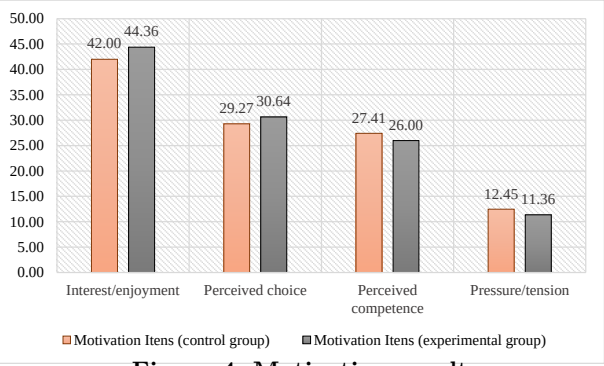

Figure 4: Motivation results

result. However, the difference is non-significant, given that the Mann-Whitney test resulted in p=0.0863.

After the teaching sessions, both groups achieved similar results. The minimum grade was 1, while the maximum grade was 3. The median for both groups was 2. This suggest that students from both groups achieve similar levels of knowledge, and that the traditional approach promoted a higher learning.

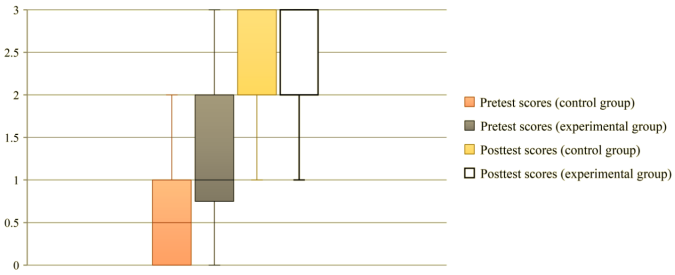

Figure 5: Knowledge tests results

To check which teaching approach had better results with respect to students' performance, we calculated the Δ value. Figure 6 summarizes the results. It shows a larger variation for the control group, and hence no trend for a particular value. Minimum and maximum values in pre-test and post-test for the control group were 0 and 2, and 1 and 3, respectively. The median shows that the control group had higher Δ values than the experimental group. However, the Mann-Whitney test indicates no significant difference (p=0.0645). Therefore, we conclude that there was no difference regarding learning level when either traditional or gamified approaches are adopted.

We also analyzed knowledge increment along 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.

The *box-plot* of Figure 7 demonstrates the participants' effectiveness in answering correctly the questions about the basic concepts of software testing. As seen, the experimental group obtained a median of 70% of effectiveness, while the median of the control group was 90%. Regarding the minimum and maximum values, both the control and experimental groups obtained 50% of effectiveness. On the other

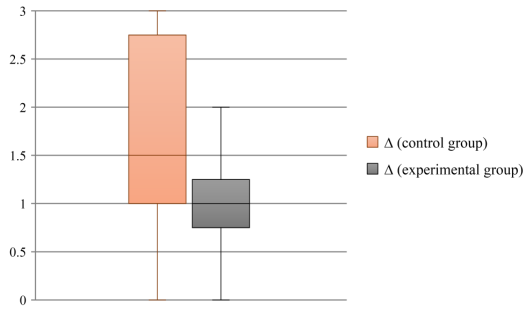

Figure 6: Boxplot with the Δs

hand, the maximum value obtained by the control group was 100% of effectiveness, against 90% for the experimental group. Completing our analysis, the Mann-Whitney test revealed a significant difference between the effectiveness value of each group (p=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.

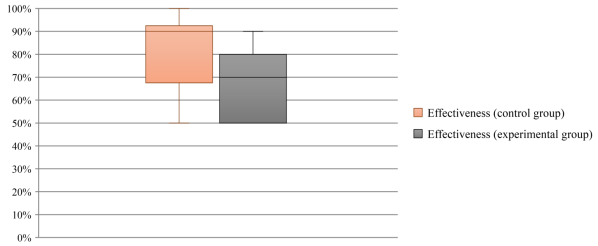

Figure 7: Students' effectiveness in the first quiz

Another quiz was applied after teaching the Functional Testing criteria. Similarly to the results in the first quiz, in this one the students from the control group reached better scores (and higher effectiveness) than the experimental group. The median for the control and the experimental groups were 80% and 70%, respectively. In this analysis it was observed an outlier in the control group, what might indicate that a student did not learn the testing criteria satisfactorily. The Mann-Whitney test revealed a significant difference between the effectiveness value of each group (p=0.0140). Therefore, we conclude that the students who learned the Functional Testing criteria in the traditional approach were more effective to answer correctly the questions than the students who learned with the gamified approach.

4.3 Students Perception

For this subsection, Table 5 presents the questions (and their scores) from a questionnaire that intended to gather students' feedback regarding the issues we tried to solve with our gamification approach.

Revisiting the problems presented in Table 3, we reasoned about the consequences each issue could cause. For example,

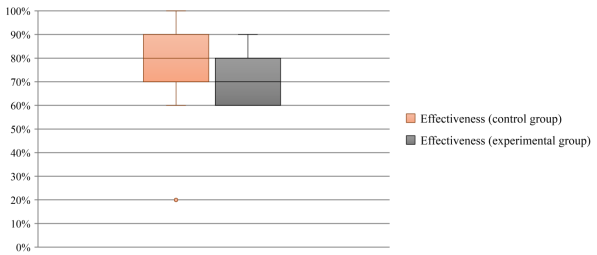

Figure 8: Students' effectiveness in the second quiz

Table 5: Results of the Students' feedback

Questions	Control Group	Experimental Group
1. The manner as software testing concepts were taught attracted my attention	5.3	6.55
2.The manner as software testing was taught made the learning more enjoyable	4.65	6.64
3. The quizzes reinforced the learned content	6.35	6.73
4. The practical exercise reinforced the learned content	6.45	6.36
5. The exercise performed in groups motivated me to work in team collaborating with my classmates	6.5	5.55
6. O difficulty level in the quizzes and in the practical exercise was coherent to the taught content	6.05	6.82
7. I still remember of the content that was taught in the mini course	5.85	5.55
8. After this mini course, I am confident that I can perform software testing activities using the Functional Test criteria	5.25	5.36
9. I intend to perform testing activities in my future projects (even if not required)	6.55	5.91
10. After this mini course, I understood the importance of the software testing activities	7	6.82

if the traditional approach was inefficient, a possible consequence would be distracted students. We then proposed to use gamification to turn the classes more enjoyable, thus luring their attention to the concepts in which they are being exposed. Question 1 aimed to check if this goal was achieved, and the result is that the gamified approach helped to attract students' attention more than the traditional approach.

Other positive result we obtained with the use of gamification was that it turned the learning funnier (question 2 in Table 5). Even though the students' performance from the experimental group was lower than expected, we believe that it is a first step towards a more successful approach.

The last question regards the desired behavior of collaboration (question 5 in Table 5). Although we expected gamification would motivate collaboration among students, specially when working in teams during the final challenge, the result suggests the opposite: the control group felt more motivated to work in teams than the experimental group.

As seen on Table 5, the other questions did not have significant differences between the experimental and control groups. This result raises two possibilities: (i) our gamified approach was not enough to reduce the challenges posed to the software testing education; or (ii) the simple fact of applying quizzes and a practical exercise (both without game elements) after each concept taught was as good as the use of an alternative approach (*i.e.* gamification).

5 THREATS AND LIMITATIONS

The threats to validity we identified are classified in four categories: internal, external, construct, and conclusion [28]. Actions taken to mitigate the threats are also presented.

Internal validity: The identified threats were: (1) researcher influence; (2) questions and requirements used in the quizzes and final challenge; (3) fatigue. Regarding (1), the same researcher ran the experimental sessions, and the analysis were carried out and discussed with the other researchers. Regarding (2), the quizzes only included questions about the taught concepts, and the requirements for the test cases creation 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 four-hour classes.

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), all participants had software engineering classes with testing topics already. Regarding (2), students chose their groups aiming at increasing their engagement and collaboration. Regarding (3), 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 [8] to specifically measure the intrinsic motivation. Regarding performance, we gathered descriptive statistics and, additionally, applied the Shapiro-Wilk and Mann-Whitney tests to compare the differences between our two independent groups.

Conclusion: Our sample is small (11 and 22 subjects in the experimental and control groups, respectively). Unfortunately, 7 (out of 18) students in the experimental group did not complete the teaching sessions, and hence the samples were not balanced. We highlight the difficult to attract middle or final year students (that is, when usually software testing courses are taught) to take part in this kind of experiment. A main difficulty 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 freshers in CS. Having this in mind, we believe our designed approach and results can be considered hints and a step towards more robust experiments with larger number of subjects[13]

We also identified some limitations in 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. So that, the experiment was shaped as a mini course to present the

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

most important concepts, and focusing on teaching more deeply Functional Testing with practical examples.

The experimental group had two 2-hour sessions split in two consecutive days. The control group, differently, had two 2-hour sessions on the same day, with an interval of 20 minutes. Although there is a recommendation of sessions no longer than two hours to avoid fatigue [22], students from the control group were having classes of the last week of the academic semester, and there was no other available day for running the experiment. For the control group, an interesting point is that even with the possibility of fatigue, such group had better performance than the experimental group.

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. To assess the impact of the use of gamification in software testing education for students' motivation and performance, we conducted four experimental sessions to address our research questions, which we revisit and answer next:

RQ1. As 2 out of the 3 positive motivational aspects (*Interest/enjoyment*, *Perceived choice*, and *Perceived competence*) in the experimental group were reported with higher scores, and the negative aspect (*Pressure/tension*) was lower than in the control group, we concluded that the students' motivation who learned in the gamified approach was higher than the students' motivation who learned in the traditional approach. Thus, we reject the null hypothesis **H1.0**

RQ2. Regarding learning level (assessed through the pre-test and post-test) when either traditional or gamified approaches are adopted, the Mann-Whitney test indicated no significant difference (p=0.0645). However, it was observed in both quizzes a significant difference between the effectiveness value of each group, which indicates that the students' performance who learned in the gamified was lower that the students' performance who learn in the traditional approach. Therefore, we reject the null hypothesis **H2.0**.

Even though the descriptive analyses showed that performance of the experimental group was lower than the performance of the control group, we believe that it is worth using gamification in software testing education. To support our belief, we remind the other positive benefits reached with this approach, such as the students' motivation, participation, attention, and the classes more enjoyable and funnier. Due to space limitation, we will discuss about the researcher's observations in other opportunities.

As future work, our approach can (and should) be improved. Besides that, other shorter experimental sessions through an academic semester and with more participants should be performed to obtain more representative data.

ACKNOWLEDGMENTS

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 #2013/07375-0), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, and Federal University of São Carlos.

REFERENCES

[1] P. E. Anderson et al,. 2015. Facilitating Programming Success in Data Science Courses Through Gamified Scaffolding and Learn2Mine. In *ITiCSE'15*.
[2] R. Bartle. 1996. Hearts, Clubs, Diamonds, Spades: Players Who Suit MUDs. *Journal of MUD Research* (1996).
[3] J. Bell et al. 2011. Secret Ninja Testing with HALO Software Engineering. In *SSE'11*.
[4] F. B. V. Benitti. 2017. As a Teacher, I Want to Know What to Teach in Requirements Engineering So That Professionals Can Be Better Prepared. In *SBES'17*.
[5] J. F. P. Cheiran et al. 2017. Problem-Based Learning to Align Theory and Practice in Software Testing Teaching. In *SBES'17*.
[6] B. S. Clegg et al. 2017. Teaching Software Testing Concepts Using a Mutation Testing Game. In *ICSE'17 Education Track*.
[7] T. Dal Sasso et al. 2017. How to Gamify Software Engineering. In *SANER'17*.
[8] E. L. Deci and R. M. Ryan. 2011. Levels of Analysis, Regnant Causes of Behavior and Well-Being: The Role of Psychological Needs. *Psychological Inquiry* 22, 1 (2011).
[9] S. Deterding et al. 2011. From Game Design Elements to Gamefulness: Defining "Gamification". In *MindTrek'11*.
[10] D. J. Dubois and G. Tamburrelli. 2013. Understanding Gamification Mechanisms for Software Development. In *ESEC/FSE'13*.
[11] F. García et al. 2017. A Framework for Gamification in Software Engineering. *Journal of Systems and Software* 132 (2017).
[12] G. M. Jesus et al. 2018. Gamification in Software Testing: A Characterization Study. In *SAST'18*.
[13] K. T. Lyra et al. 2016. Infographics or Graphics+Text: Which Material is Best for Robust Learning?. In *ICALT'16*.
[14] G. J. Myers et al. 2011. *The Art of Software Testing*. Wiley.
[15] L. N. Paschoal et al. 2017. Abordagem Flipped Classroom em Comparação com o Modelo Tradicional de Ensino: Uma Investigação Empírica no Âmbito de Teste de Software. In *SBIE 2017*.
[16] L. Z. Pedro. 2016. *Uso de Gamificação em Ambientes Virtuais de Aprendizagem para Reduzir o Problema da Externalização de Comportamentos Indesejáveis*. Master's thesis. ICMC/USP.
[17] F. S. Pinto and P. C. Silva. 2017. Gamification Applied for Software Engineering Teaching-learning Process. In *SBES'17*.
[18] J. M. Rojas and G. Fraser. 2016. Code Defenders: A Mutation Testing Game. In *Mutation'16*.
[19] J. M. Rojas and G. Fraser. 2016. Teaching Mutation Testing using Gamification. In *ECSEE'16*.
[20] T. Shepard et al. 2001. More Testing Should Be Taught. *Commun. ACM* 44, 6 (June 2001).
[21] S. Sheth et al. 2012. *Increasing Student Engagement in Software Engineering with Gamification*. Tech. Report CUCS-018-12. Columbia University.
[22] Janet Siegmund and Jana Schumann. 2015. Confounding parameters on program comprehension: a literature survey. *Empirical Software Engineering* 20, 4 (2015).
[23] J. Smith et al. 2012. Using Peer Review to Teach Software Testing. In *ICER'12*.
[24] D. M. Souza et al. 2012. Aspectos de desenvolvimento e evolução de um ambente de apoio ao ensino de programação e teste de software. In *SBIE'12*.
[25] P. H. D. Valle. 2016. *Jogos Educacionais: Uma Contribuição para o Ensino de Teste de Software*. Master's thesis. ICMC/USP.
[26] P. H. D. Valle et al. 2015. CS Curricula of the Most Relevant Universities in Brazil and Abroad: Perspective of Software Testing Education. In *SIIE'15*.
[27] K. Werbach and D. Hunter. 2012. *For the Win: How Game Thinking Can Revolutionize Your Business*.
[28] C. Wohlin, et al. 2012. *Experimentation in Software Engineering*. Springer Berlin Heidelberg.
[29] P. E. Yujian Fu and P. J. Clarke. 2016. Gamification-Based Cyber-Enabled Learning Environment of Software Testing. In *ASEE'16*.