Does Gamification Improve the Training of Software Testers? A Preliminary Study from the Industry Perspective*

Pedro Henrique Dias Valle*
pedrohenriquevalle@usp.br
University of São Paulo
São Carlos, São Paulo, Brazil

Ricardo Ferreira Vilela*
ricardovilela@usp.br
University of São Paulo
São Carlos, São Paulo, Brazil

Elis C. Montoro Hernandes†
elis.hernandes@monitoratec.com.br
Monitora Soluções Tecnológicas
São Carlos, São Paulo, Brazil

ABSTRACT

Background: Software testing is an essential activity in software companies to ensure the quality of their systems. Nevertheless, the most recent studies demonstrate that there is a lack of qualified professionals in this area. **Aim**: To check whether the gamification contributes to the training of software testers in the industry, analyzing their motivation and test cases designed by them. **Method**: For this, we conducted an experimental study with junior testers in a software company to compare gamified training against non-gamified training. **Results**: We observed significant contributions from gamification in the training of software testers, especially in their motivation and quality of the test cases. We noted that non-gamified training still offers contributions to software testing training. This result intensified when we embedded gamified elements in the training since we noticed a difference between groups at scenario coverage analysis. **Conclusions**: Our results should encourage software companies to emphasize training in software testing and adopt gamified elements because software testers are more motivated in training and design better test cases from the theories of software testing.

CCS CONCEPTS

• Software and its engineering → Software testing and debugging; **• Social and professional topics →** *Software engineering education.*

KEYWORDS

Software Testing Training, Gamification, Industry

ACM Reference Format:
Pedro Henrique Dias Valle, Ricardo Ferreira Vilela, and Elis C. Montoro Hernandes. 2020. Does Gamification Improve the Training of Software Testers? A Preliminary Study from the Industry Perspective. In *19th Brazilian Symposium on Software Quality (SBQS'20), December 1–4, 2020, São Luís, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3439961.3440004

*This author is also with Centro de Ciências Técnológicas (CCT/UENP)

SBQS'20, December 1–4, 2020, São Luís, Brazil

ACM ISBN 978-1-4503-8923-5/20/12…$15.00
https://doi.org/10.1145/3439961.3440004

1 INTRODUCTION

The purpose of software testing is to execute programs or models with specific inputs to verify that they behave as expected. In light of this, it is possible to perform a detailed analysis to lead them to failures and through by debug eliminate the defects that caused to the failures [8]. Despite software testing be recognized as an important activity for software product quality assurance, there is a lack of qualified professionals in this area since that they have difficulties to apply testing techniques, criteria, and tools [25, 31]. The lack of qualified professionals may be related to, mainly, the deficiency in the education and training of software testers and the lack of motivation caused by the workplace. Also, there is an absence of strategies of allocation and accountability of such professionals in the development and test teams [29].

In order to understand how the software testers have been trained, Valle et al. [29] carried out an analysis of curricula proposed for undergraduate in Computer Science. They observed that undergraduate courses in computing area address software testing contents as a topic of Software Engineering course in which few class hours are given for education of this content [29]. The authors also analyzed the curricula of the best universities. As a result, they observed that, in general, software testing is addressed only in the Software Engineering course, applying an isolated view of other undergraduate courses, such as Programming Fundamentals.

Aiming to mitigate these issues, many initiatives have been proposed to train and motivate the software testers, as educational modules, peer testing, software testing education with programming, PBL (Problem Based Learning), and social networks [30]. The gamification is an example of these initiatives in which uses game elements outside their original scope [9], aiming to improve the individual's engagement, motivation, and experience, as well as improve education and training processes.

In this perspective, Bell et al. [4] proposed a social approach, in which a system with gamified questions about software testing was used to support the education and training of this content, providing an increased interest in the content addressed and encouraging social interaction among those involved. Fraser [12] also investigated how to apply gamification in the education and training of software testing. For this, the author analyzed three environments that used gamification, called: CodinGame, Code Defenders [24], and Code Hunt [5].

Despite there are different approaches to support the education and training of software testing [31], there is a lack of research that shows the effectiveness of gamification in the education and training of software testing, mainly in the industry. This may be related to the fact that gamification is a relatively new area and also the difficulty of finding gamified resources of good quality to support software testing training.

This paper analyzes whether gamified training contributes to the knowledge of software testers. For this, we carried out an experimental study in the industry to compare the motivational stimulus and the accuracy and precision of designed test cases by software testers who received a gamified and non-gamified training. The contributions of our paper are summarized as follows:

(1) We conducted an experimental study that provides preliminary evidence of how software testing training can contribute to the generation of test cases by software testers;
(2) Also, we show that gamification improves software test training and the quality of test cases;
(3) To the best of our knowledge, this study is the first empirical evaluation of gamification in the motivation and skills of software testers in the industry.

This paper is structured as follows: Section 2 provides the main concepts needs to understand the contribution of this study and related work. Section 3 presents the planning of the experimental study.Section 4 outlines a gamified material snippet to exemplify the material used in our experimental study. Section 5 presents the main results for each research question. Section 6 discusses the main results obtained. Finally, Section 7 summarizes the conclusions and makes suggestions for future work.

2 BACKGROUND AND RELATED WORK

This section defines the main concepts needed to understand this experimental study. First, we describe the main definition, techniques, and criteria for software testing, as well as the definition of gamification, their benefits, and main elements. We also describe related work.

2.1 Software testing

Software testing is by a great deal the most generally used method for quality assurance and quality control in a software company and a very important step in the development process. Testing makes it possible to analyzing whether such programs behave according to expected, performing a detailed analysis to find faults and use debugging to eliminate the defects that cause such faults [8].

The software testing activity is composed of different criteria that define which constituent parts of the product need to be tested to reveal the presence of faults in the system under test (SUT). Each criterion divides the input domain into different subdomains. Different and infinite test cases can be derived to satisfy a given criterion. Software testing is used in practice to ensure the quality of software developed since it is impossible to ensure that it is free from defects [8]. Software testing criteria can be divided into three techniques: **functional**, **structural**, and **fault-based** testing. In our study, we considered only functional testing.

The functional testing (also known as black-box testing) is based on only the software specification. The criteria of functional testing can be applied at any stage of software testing (unit testing, integration testing, system testing, and acceptance testing), regardless of the paradigm used since the details of implementation are not analyzed. In the functional testing perspective, the software is analyzed according to the user's point of view [6, 20].

Functional testing identifies only defects related to software malfunction. These errors are revealed from the outputs of the system under testing. This technique is used to identify incorrect functions that lead to unwanted output when executed in practice [20]. It is important to highlight that the effectiveness of functional testing depends on the product specification quality [6]. The most well-known functional testing criteria are equivalence class partitioning, boundary value analysis, systematic functional testing, and cause-effect graph [8].

2.2 Gamification

In the last years, many studies about educational games have been conducted to proposed new approaches to improve the teaching of educational contents of an attractive and dynamic way [22]. In a similar perspective, gamification has emerged as a new research area in recent years, due to several benefits that it provides. According to Kapp [17] gamification can be defined as the use of digital games elements to create a playable experience in tasks and contexts that are not digital games to provide for people a better engagement and motivation in teaching and training of contents [32].

The number of research about gamification applied in education has increased considerably in the last years [27, 28]. Such an increase can be related to the capability of gamification in improving the people's engagement and motivation [17]. According to Dignan [10], there are at least 19 game elements that could be used in applications with gamification, such as goals, competition, opportunities, news, levels, badges, progress, points, recognition, classification, among others. In the educational context, such elements can provide a better degree of involvement and motivation compared to traditional teaching methods. Despite this, it is not necessary to use all elements of gamification to obtain an engaging effect [10].

Gamification can contribute positively or negatively to education/training of specific contents. Regarding the positive aspects, an increase in the motivation and engagement of participants that received a gamified approach has been reported, as shown by the results of this study. There are also studies that show that gamification can provide some disadvantages, like indifference, loss of performance, and undesired behavior [28]. Human motivation and personal satisfaction are reasoned by three main pillars: autonomy, mastery, and purpose. These pillars are present in digital games context since autonomy is related to the control of all the player's actions, mastery is related to the player's continuous evolution, and the purpose to the interactions among the player and other characters or objects within the game context [3].

In this perspective, studies that used gamification to support the teaching and training have been conducted. Kapp [17] and Hamari et al. [15] obtained positive results with the application of gamification in teaching and training processes in schools and companies. Andrade and Canese [2] developed an application to support the teaching of formal logic. According to the authors, the performance of students in the discipline has improved considerably. Domínguez et al. [11] proposed a study that explored gamification on a teaching platform in a university course. The results showed that the students who participated in the experiment were more motivated and obtained better performance in the tests compared to the other students.

Despite the benefits aforementioned, some studies have recognized negative effects on the application of gamification. These studies related the use of gamification elements in applications, however, such elements do not provide motivation and engagement of people involved, causing effects contrary to what is proposed by the definition of gamification [26]. Therefore, it is necessary to use systematic methods and approaches that provide guidelines to avoid undesirable results when applying gamification in the educational context.

2.3 Related Work

To the best knowledge, this is the first work that investigates the impact of gamification in software testing training from the industry perspective. Therefore, we do not have directly related works to our experimental study. In another perspective, we identified works that investigate how gamification impacts the software testing education at the university, as discussed below.

Jesus et al. [16] conducted an experimental study to evaluate the impact of gamification on software testing education. This experimental study was carried out with undergraduate students from three Brazilian institutions. As a result, the authors reported that the treatment group was more motivated than the control group, and regarding the performance.

Fu and Clarke [14] proposed software testing cyber-enabled learning environment, called WReSTT-CyLE. This tool is a gamified system with the PBL triad to engage and motivate students to learn the contents of software testing. WReSTT-CyLE uses concepts as social interaction, collaborative learning, and learning objects to support student education of software testing concepts effectively and efficiently. As a result, the authors concluded that this tool is an efficient system to engage and motivate software testing learning. Rojas and Fraser [24] investigated the use of gamification to support the mutation testing education and to strengthen testing skills. The authors proposed the Code Defenders to support the teachers in delivering complex mutation testing concepts, besides that make the learning experience more enjoyable and fruitful for students.

The main innovation of our work concerning to existing works in the literature, it is that we investigate how gamification can impact software testing training in the industry, analyzing the motivation of software testers, in initial of career, to learn new contents and the accuracy and precision of the designed test cases by them.

3 STUDY PLANNING

We followed the guidelines proposed by Wohlin et al. [33] to plan and carry out our study. This section presents the planning to analyze the feasibility of gamified training in software testing context. The planning composed of definition and context, independent and dependent variables, hypotheses formulation, data analysis, instrumentation, and threats to validity.

3.1 Definition and context

We use the GQM (Goal Question Metrics) template to define the goal of our study, as presented follows:

"**Analyze** the designed test cases by software testers and their motivational stimulus **for the purpose** of evaluate them **with respect to** the accuracy and precision of test cases and the motivational aspects of software testers **from the point of view** of software testers **in the context of** the industry".

From the goal mentioned before, we formulated and investigated two Research Questions (RQs):

RQ_1: What are the impacts on the motivation of software testers in their training? In order to answer RQ_1, we compare two types of training (i.e., non-gamified and gamified) with regard to the motivation of software testers. We measure the motivation metric from four different components with 36 questions provided by IMMS (Instructional Materials Motivation Survey).

RQ_2: How gamified training contributes to the quality of designed test cases by software testers? For answering this RQ, we intend to identify whether the type of training contributes to the quality of testing concerning the scenarios covered and the number of test cases.

We invited junior software testers, in the company, who would receive software testing training to participate in our study. Six software testers accepted participating in our study. All participated voluntarily and they should: i) declare interest in participating in this study by signing a consent form; ii) fill the form of profile characterization to check each participant's level of knowledge; and iii) answer a feedback questionnaire.

3.2 Independent and dependent variables

According to Wohlin et al. [33], we need to choose the dependent and independent variables of study before defining the experiment design. In light of this, independent variables are variables that we control and change in the experiment. Such variables should be some effect on dependent variables and they require domain knowledge. Usually, independent variables represent the cause that affects the result of the experimentation process. For our experimental study, we manipulate one independent variable:

- **Software Testing Training:** there are numerous types of training, concern different levels and techniques of software testing. Furthermore, most techniques use the same teaching resources, which usually is exhaustive and, hence may not meet training expectations. Gamified methods have been used successfully in other areas of knowledge for skills training. In order to investigate these techniques in the context of software testing, we manipulate two different types of software testing training (*i.e.,* gamified and non-gamified).

Dependents variables refer to the output of the experimentation process, which is affected during this process. Usually, the effect of treatment is measured in the independent variables. In this context, we measure three dependent variables computed over motivation and test cases of participants:

- **Motivation:** the motivational stimulus of software testers was analyzed by application of IMMS [19] that is composed of four components, as described following:
  - **Attention:** it can be obtained through encourage perception and inquisitive stimulation. For this, many elements must be considered, as active participation, variability, humor, contradiction, and conflict.
  - **Relevance:** it aims to expand the motivation of participants. For this, a concrete language that participants are familiar with must be used.

  - **Confidence:** it can support participants to understand their possibilities of success, as well as the objectives and prerequisites for performing tasks. For this, appropriate feedback can be provided to participants considering the incremental learning levels.
  - **Satisfaction:** it suggests that learning be rewarding or satisfactory (feeling of achievement, praise, among others)
- **Accuracy:** (Measured by Equation 1) we analyzed the coverage of scenarios achieved by test cases.
- **Precision:** given by the Jaccard index (Measured by Equation 2). Let **A** and **B** be two test case suites, where **A** represents test suite reference and **B** test suite designed by testers.

$$Accuracy = \frac{Achieved\_Scenarios}{Total\_Scenarios} \tag{1}$$

$$Jaccard(A, B) = \frac{|A \cup B|}{|A \cap B|} \tag{2}$$

3.3 Hypotheses formulation

For RQ_1, we formulated the following hypotheses:

- **Null hypothesis** (H_0): There is no difference, between the motivation level of gamified and $non-gamified$, with regard to the motivation subscales (H_0: gamified = non-gamified).
- **Alternative hypothesis** (H_1): There is a difference between the motivation level of gamified and $non-gamified$, with regard to the motivation subscales (H_0: gamified ≠ non-gamified).

We evaluated the H hypothesis for the motivation of software tester. With the null hypothesis, the motivation of gamification training achieves a motivation level equal to non-gamified. Following hypotheses were defined on the basis of subscales of motivation. Following, we present the hypothesis formulated to RQ_2:

- **Null hypothesis** (H_0): There is no difference, between the test quality of gamified and $non-gamified$, with regard to coverage scenarios and number of test cases ($H_0 : gamified = non-gamified$).
- **Alternative hypothesis** (H_1): There is a difference, between the test quality of gamified and $non-gamified$, with regard to coverage scenarios and number of test cases ($H_0 : gamified \neq non-gamified$).

3.4 Data analysis

For each RQ, we calculate the statistical hypothesis tests according to the number and sampling distribution. The sample distribution is calculated from the Shapiro-Wilk test and nonparametric data is calculated using the Mann-Whitey Wilcoxon test, matched-pairs signed-rank for RQ_1 and unpaired for RQ_2. For each statistical hypothesis tests, we decided to accept a confidence interval of 95% (0.05 probability of a mistake). In addition, Cohen's Effect Size paired measure was used to evaluate the magnitude of the difference estimated by the statistical test.

3.5 Instrumentation and Execution

In order to conduct this study, we defined a consent form for that software testers declared their interest in participating in our study in a voluntary way before starting the application of study. Figure 1 illustrates an overview of our study, the artificial generated facial pictures were extracted by StyleGAN Karras et al. [18]. Following, we present how each step was performed.

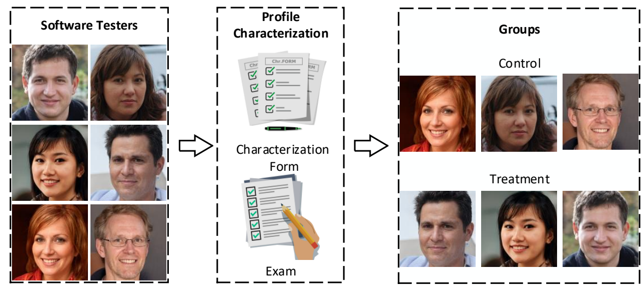

Figure 1: Overview of experimental study. Artificially Generated Facial Pictures from StyleGAN

First, we applied a questionnaire to characterize the knowledge level of each software tester. We also used the exam score adopted by the company for the admission of employees. This exam helped us understand the participants' profiles of this study. These instruments were important to support the distribution of participants into two groups (control and treatment).

We provide two pages of the CRM Medicalbox system (Page A and Page B) for groups to design their test cases for features of the system. MedicalBox is a medical system that provides many features, since saving money (phone bill and secretary) to automating tasks (confirmation of clinical consultation). Examples of such features are: smart to address and patient electronic records. The pages of the system adhered in our study are shown in Figures 2 and 3.

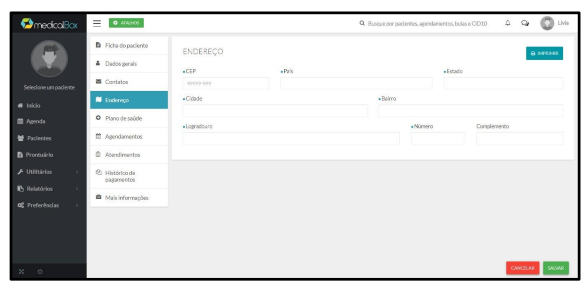

Figure 2: Page A of CRM MedicalBox

On page A, the required data are the patient's name, profession, indication, sex, marital status, birthday, age, Brazilian ID (RG), and Brazilian Social Security Number (CPF). *Print*, *Cancel* and *Save* features also must be working correctly. On page B the addresses of the patients are registered with the information: Country, State, City, Neighborhood, and Street. After software testers were divided into two groups they design test cases to identify faults features of the MedicalBox without receiving the training for improving their software testing skills, as illustrated in Figure 4.

In this way, each group received software testing training. The control group received the training composed of material about

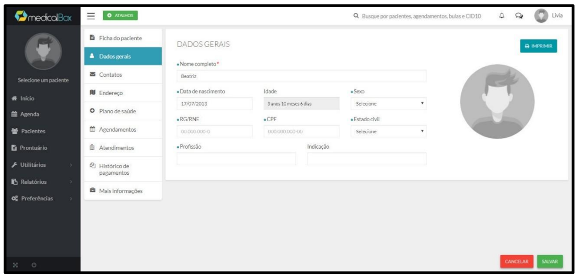

Figure 3: Page B of CRM MedicalBox

software testing techniques. This material are slides about introduction of software testing (*materials suppressed for blind review*). We considered the slides: "Introduction", "Equivalence Partition" and "Limit Value Analysis" from Topic 2 - ATS Black Box Testing Technique. Following, software testers answered a quiz composed of questions to update software testing contents aiming to fix the knowledge about functional testing. From this intervention, software testers could update designed test cases by them previously. The motivation to use the functional testing concepts is due to the application of such concepts in the industry by exclusive testing teams. It is worth highlighting that we use the training material provided by the company. Also, it is important to stress/emphasize that functional testing is the technique with the most comprehensive compatibility between academia and industry since many concepts from literature are not yet employed in medium-sized companies.

At the same time, the treatment group used a gamified material to improve their software testing skills, specifically regarding to equivalence partitioning and boundary value analysis. The gamified material is composed of the same slides used by the control group with the addition of gamification elements, as points, time, level, achievement, badges, among others. In this way, software testers answered the same quiz the control group answered. This quiz was answered through the Kahoot platform with add of gamification elements. An example of this gamified material is displayed in Section 4. From this intervention, software testers were able to review the designed test cases elaborated by them previously, enabling the change in the final set of test cases.

In order to standardize the designed test cases, we provide a template for software testers to design their test cases in the required format. This helped us analyze the test cases that were designed. After the review of test cases, software testers answered an instrument to evaluate their motivation in receiving the proposed training. This instrument called IMMS [19]. It is composed of thirty-six questions with five answer options in Likert scale format, in other words, raging from 1 to 5. The answer options could be: (1) strongly disagree; (2) partially disagree; (3) indifferent; (4) partially agree; and (5) I totally agree. Among 36 questions, 12 analyze attention, 9 analyze relevance, 9 measure confidence, and 6 evaluate user satisfaction. This instrument is a questionnaire that addresses the four subscales of the ARCS model.

Finally, each software tester answered a questionnaire about the study conducted. In this instrument, software testers should authorize the use of their data for our study. They also answered questions about the study structure and offered suggestions for improvements to future works.

3.6 Threats to Validity

Threats to validity should be considered in any experimental study planning. In order to minimize the biases of our study, we carried out some actions to mitigate the three types of threats identified, as presented below:

Internal Validity: in order to mitigate this problem, study participants should be carefully analyzed, mainly regarding population selection, class division, social aspects, among others. The selection of software testers may have influenced the result of the study. However, software testers did not receive rewards or favoritism from the company for behaving abnormally to obtain an advantage during the study.

External Validity: the interaction among participants selection and the execution environment interaction should be observed. The software testers' experience could be considered a type of threat to validity. For mitigating this issue, we applied a questionnaire of profile characterization to measure the participants' skills. Therefore, the control and treatment groups were divided equally regarding the professionals' skills. The number of participants can also be considered as a threat for this study. However, carry out a study with the participants from the industry is not a trivial task, mainly because our study is focused on newly hired professionals.

Construction Validity: the reliability of measurements should be carefully considered. The methods used to evaluate the motivation and the quality of test cases can be considered a threat to the validity of this study, once they may favor one of the addressed approaches. However, assessment methods that were selected are well-known, used, and evaluated in the scientific literature.

Conclusion Validity: refers is concerned with the relationship between the treatment and the outcome, in order words, it refers to the choice of the statistical test. For mitigating this issue, we select tests according to the characteristics of samples, considering the data distribution and the number of variables. In order to complete the hypothesis tests, we used effect size tests to demonstrate the degree of difference between the comparisons made.

4 GAMIFIED SOFTWARE TESTING TRAINING

On the basis of the experimental design, we adopted an existing material about software testing and we extended this it for a new gamified version to assess gamification in software testing training. These materials are composed of slides about the fundamentals of software testing. As mentioned in Section 3.5, the control group used a non-gamified material. Figure 5 illustrates a material non-gamified snippet about the Equivalence Class Testing [7], where is shown the explanation about valid and invalid classes.

In the gamified training, we addressed gamification elements, as outline in Figure 6. We included points, time, classification, and level into the gamified training. Therefore, participants interacted with the content presented, as well as with the other software testers who received the gamified training. Thus, we make available the ranking with the performance of software testers into the screen of gamified material, and also the time to finish each proposed

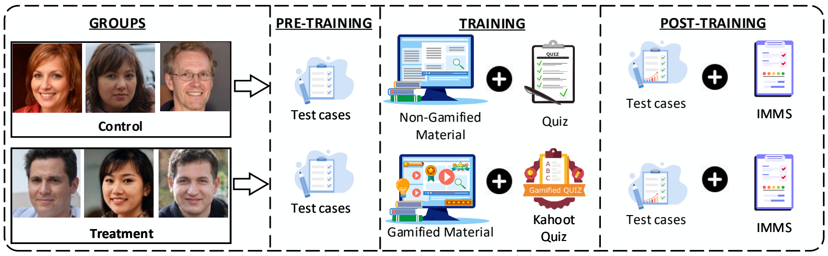

Figure 4: Steps of Experimental Design. Artificially generated facial pictures from StyleGAN

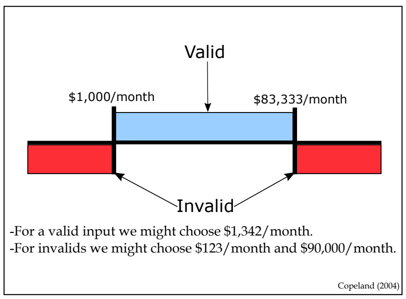

Figure 5: Material non-gamified snippet

activity. We updated the ranking of software testers according to their software testing achievements. Figure 6 shows that software testers were at level 2, in other words, they had learned the contents about equivalence class testing, and at this time they had learned about their application. The control and treatment groups answered the same quiz, for the treatment group, we make available this quiz in the Kahoot platform, which includes gamification elements. Figure 7 shows a quiz snippet in the Kahoot platform.

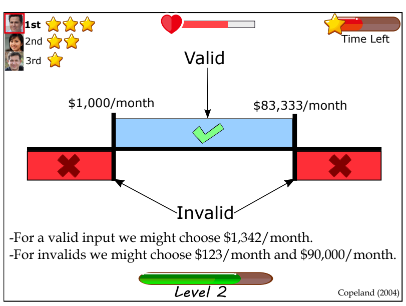

Figure 6: Gamified material snippet

In the Kahoot screen, we have shown the questions of the quiz for software testers and answer options for them to choose one. At the same time, the tester's screen displayed the answer options for each question. For each question, software testers had 20 seconds to choose an answer option. The Kahoot updates the ranking after each question answered, analyzing the correct answers in the shortest possible time.

5 RESULT ANALYSIS

The participants of our study are six software testers who were starting their careers in software testing in the company. The control and treatment groups are composed of three participants each. It worth highlighting that all participants had a basic knowledge of software testing acquired during the undergraduate course.

In the design of our experiment, treatment (gamified training) is applied to experimental units in the treatment group. Members of a control group receive standard treatment (non-gamified training). Following, we describe the answers to research questions planned in Section 3.1.

5.1 The motivation of software testers

In order to check whether there is a difference in the motivation level of software testers who received two training types, we analyzed the motivation level of each participant checking the motivation components of the IMMS (Attention, Relevance, Confidence, and Satisfaction). Figure 8 illustrates results among the different components of motivation for each training approach.

The gamified training presented statistically different for most of the IMMS components, except for the confidence. We note that software testers that received gamified training obtained a higher level of **attention** to perform the proposed training. We believe that gamification elements time, points, badges, and level contributed positively to that the software testers obtained and maintained their attention for conducting the tasks proposed since such elements promote competitiveness among participants and keep them attentive in the activities that they should perform.

Regarding **relevance**, software testers who received gamified training obtained a better result. We observe that the gamification elements: achievement, progress, and classification contributed to this result. This may have occurred because we clearly define goals and we also highlight the importance of contents addressed in their professional skills. The gamification elements mentioned before shown precisely the performance of software testers post received the gamified training.

The **satisfaction** of software testers who received gamified training was greater than testers who received non-gamified training. The gamification elements: points, level, badges, achievement, goals, competition, opportunities, progress, recognition, and classification may have contributed to this result since software testers created the feeling of successfully after they concluded the proposed tasks in the gamified training.

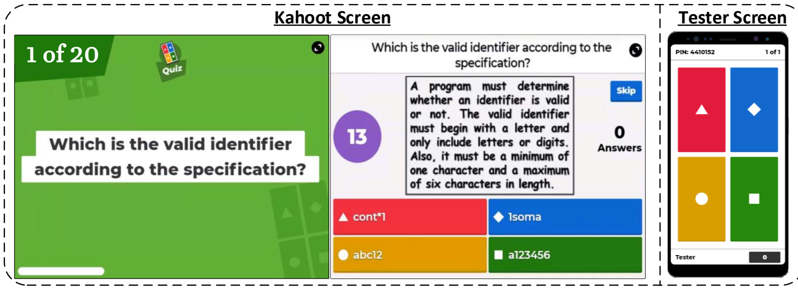

Figure 7: Quiz snippet of treatment group using Kahoot

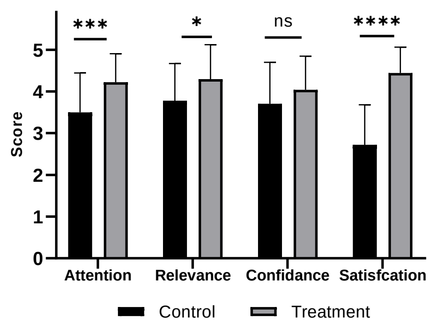

Figure 8: Range of motivation of each IMMS component. (ns p > 0.05, * p ≤ 0.05, *** p ≤ 0.001, **** p ≤ 0.0001)

Finally, **confidence** was the only IMMS component that did not have a statistical difference. This may have occurred because software testers had a positive expectation regarding their success in the company because all participants in our study were newly hired employees. From the results described above, we reject H_0 (null hypothesis) for following IMMS components: attention, relevance, and satisfaction. In other words, we observe that the use of gamified in training improves the motivation of software testers to learn the contents of this area.

In order to complement the statistical tests for evaluating the strength of a statistical claim, we compute the effect size between the training types for each motivation component. Table 1 presents the effect size and magnitude level by Cohen's D measure. In disagreement with the statistical test, the effect size measure indicates a small difference among training. This difference is concerning the confidence component. Although the magnitude level is still considered small.

Table 1: Effect Size Measure of Motivation Components

Motivation Components	Effect Size	Magnitude
Attention	0.879	Large
Relevance	0.604	Moderate
Confidence	0.368	Small
Satisfaction	2.14	Large

5.2 Quality of test cases elaborated by testers

The RQ_2 is related to the quality and efficiency of test cases elaborated by software testers. First, we analyzed the coverage of test scenarios achieved by software testers during the experiment. As mentioned aforementioned, we used two CRM MedicalBox pages with a total of 73 test scenarios (specifically, 43 scenarios on page A, and 30 scenarios on page B). The scenarios coverage is evaluated in two times, before and after software testing training. Figure 9 illustrates a summary of coverage results for page A achieved by software testers.

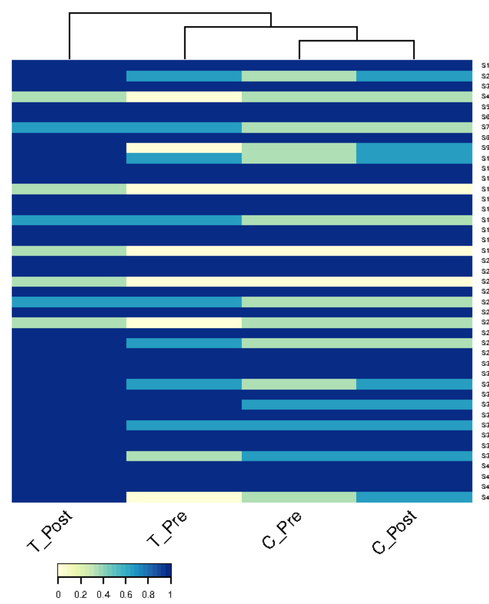

Figure 9: Scenarios Coverage Heatmap of MedicalBox Page A. (*C_Pre*: Control Group Pre-Training; *C_Post*: Control Group Post-Training; *T_Pre*: Treatment Group Pre-Training; *T_Post*: Treatment Group Post-Training)

The coverage illustrated in the heatmap shows an evolution of the treatment group post the application of gamified training. Based on cluster analysis, we noted that most scenarios are covered by the treatment group post gamified training. Besides that, there are scenarios, such as S13, S18, S22, which were covered only by the treatment group in post-training.

In another perspective, the coverage analysis on page B demonstrates that software testers found it easier to cover test scenarios, as shown in Figure 10. Although we noted that some scenarios, such as S3 and S22, were covered only by the treatment group in the post-training.

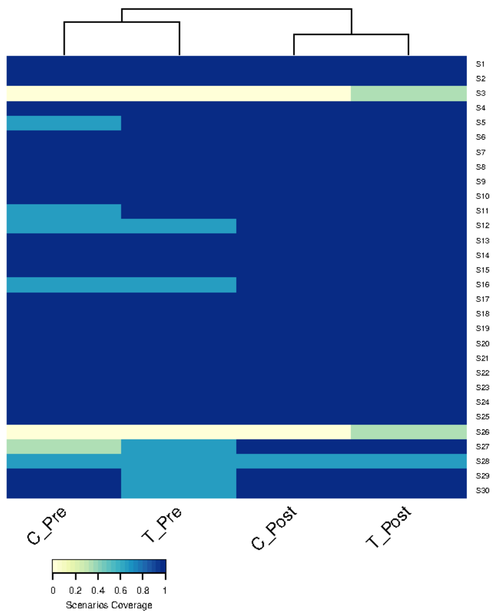

Figure 10: Scenarios Coverage Heatmap of MedicalBox Page B. (*C_Pre*: Control Group Pre-Training; *C_Post*: Control Group Post-Training; *T_Pre*: Treatment Group Pre-Training; *T_Post*: Treatment Group Post-Training)

In order to carry out the coverage analysis between treatment and control groups, we used the Wilcoxon matched-paired test considering all scenarios on the two pages of the CRM MedicalBox. First, we wanted to know if there was a difference between the groups before applying software testing training. The results of this analysis are shown in Figure 11.

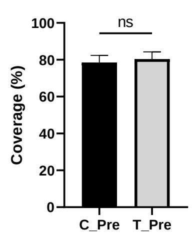

Figure 11: Scenarios coverage into pre-training by treatment and control groups. *C_Pre*: Control Group Pre-Training; *T_Pre*: Treatment Group Pre-Training (ns p> 0.05)

We compared the groups with aims to check the possible difference among software testers. Such a difference could compromise the final results of our study. Despite this threat to validity, results demonstrate that there is no significant difference between treatment and control groups concerning coverage of test scenarios.

We also investigated the effectiveness of standard training (non-gamified) for covering test scenarios. For this, we measure coverage before and after training to check if improvements have been identified concerning covered test scenarios. The figure 12 shows the results of this analysis.

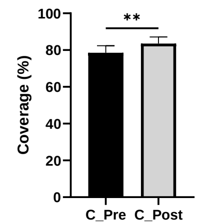

Figure 12: Scenarios coverage into pre-training and post-training by control group. *C_Pre*: Control Group Pre-Training; *C_Post*: Control Group Post-Training (** p< 0.01)

As expected, we noted that the application of non-gamified training provides the software tester additional knowledge to improve the ability to elaborate test cases. These results demonstrate the need for training to better understand the application domain and test objectives. We also investigated the previous analysis for gamified training in the pre-training and post-training. The results are summarized in Figure 13.

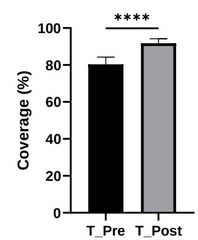

Figure 13: Scenarios coverage into pre-training and post-training by treatment group. *T_Pre*: Treatment Group Pre-Training; *T_Post*: Treatment Group Post-Training (**** p< 0.0001)

Similar to standard training, gamified training has a difference between the pre-training and post-training. The results indicate that the gamified training provided knowledge to software testers promoting improvements to the test case sets allowing them to achieve new test scenarios. Finally, to answer the RQ_2, we measure the difference between tests elaborated by groups that received the gamified and non-gamified training. Figure 14. shows the result of this analysis.

From the results is shown in Figure 14, we observed that the treatment group (gamified training) obtained a better result against the control group. Therefore, we reject H_0 (null hypothesis) for RQ_2. The effect size of the comparisons detailed above is summarized and displayed in Table 2.

Per RQ2, we also would like to quantify the ability of test suites to achieve high levels of scenario coverage. For doing this, for each page, we elaborate on a suite of reference tests. Our expectation

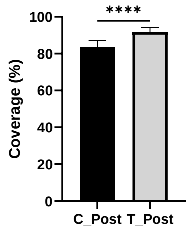

Figure 14: Scenarios coverage into Post-training by control and treatment groups. *C_Post*: **Control Group Post-Training;** *T_Post*: **Treatment Group Post-Training (**** p< 0.0001)**

Table 2: Cohen's Effect Size Paired Measure of comparisons

Comparison	Effect Size	Magnitude
Control_Pre ~ Treatment_Pre	0.110	Negligible
Control_Pre ~ Control_Post	0.380	Small
Treatment_Post ~ Treatment_Pre	0.527	Moderate
Treatment_Post ~ Control_Post	0.530	Moderate

is that if a test suite from a software tester is highly similar to the reference suite, then the suite should cover most test scenarios since the test suite reference reaches all scenarios.

We began by comparing the test suites by similarity index of Jaccard, this index allows identifying if the test sets developed by the software testers comprise the test scenarios, if they are redundant or if they do not cover the scenarios. The results of this comparison are summarized and displayed in Table 3.

Table 3: Jaccard similarity index of test cases

		Control	Treatment
Page A	Pre-Training	0.459	0.437
	Post-Training	0.621	0.722
Page B	Pre-Training	0.655	0.622
	Post-Training	0.666	0.644

We can see a similar trend between groups before the test training. For instance, on page A, the control and treatment groups present an index of 0.459 and 0.437, respectively, while on page B the indices are 0.655 and 0.622.

After groups are submitted to training, we note that the similarity with the reference suite increases, the treatment group has an increase of 65.2% post-training for page A and 3.5% for page B. The control group shows an improvement of 35.2% for page A and 1.6% for page B. The treatment group similarity index, on Page A, evolved significantly compared to the period before software training. Although there is no major change for groups on page B.

6 DISCUSSION

In a similar way to evidence found in the literature [4, 5, 13, 24], results obtained suggest that participants who received the gamified training have a greater motivation during the software testing training. We believe that gamification elements used in our context contributed positively to improve the motivation of software testers. We observe this, mainly post the treatment group answered the questionnaire inserted in the Kahoot platform.

From the use of Kahoot, software testers were encouraged to answer questions correctly, as quickly as possible. This can be encouraged competitiveness among participants involved in the treatment group. The score and classification of participants were made available, in real-time, for all software testers in the treatment group after they answer questions shown in the Kahoot platform.

Participants who received the gamified training were more engaged in the proposed activity since the created competitiveness encourages them to win in the overall classification of participants. This desire contributed to participants pay attention to software testing contents that were presented, as well as questions to evaluate their performance in the proposed activities. Thus, participants in our study understood the relevance of provided contents, as well as had satisfaction in participating into proposed training.

Gamification also contributed to the quality of designed test cases. In general, testing training is not an activity that has been much investigated [29] and, hence, most software testers enrich their knowledge through trial and error. The importance of software testing training is confirmed in our results, as shown in Figure 14, since even in traditional training, results are significant post the training of testers. Regarding gamified training, we observe that participants were more attentive to the contents of software testing during the design test cases since the scenarios were purposely established to comply with the fundamentals of software testing. The test cases of the treatment group have greater adherence to test techniques. For example, techniques of equivalence partition and limit value analysis were noticeable in designed test case sets.

The results of participants who received gamified training suggest improvements ranging from the comparison between the pre-training and post-training periods (see Figure 13) until the comparison between traditional training (non-gamified) and gamified training (see Figure 14). As an example, the heatmap illustrated in Figure 9 shows that there are more complex scenarios, such as S13, S19, and S22. These scenarios are generated from regular expressions contained in an HTML form that requires more effort to design test cases. We observe that, despite the low coverage, these scenarios are only covered by the treatment group because the participants better learned the fundamentals of testing.

It is important to highlight that, despite using gamified elements for the treatment group, the fundamentals of software testing were also included in the training materials for both groups. This evidence can directly impact the quality of services provided by software testers. We know that the formal application of some testing techniques is not feasible in an agile and dynamic environment of companies. Although the fundamentals of these techniques provide skills that can be employed an intrinsic way.

We understand that despite the high coverage of the test scenarios, the suites still do not show a high similarity index, as shown in Table 3. We note that both groups tend to expand the number of test cases to cover all functionalities of the system under test. However, software testers should pay attention to redundant test cases because they can harm the team in the time to market. The scenario coverage on page B shown in Figure 10 is an example that did not show much difference between the pre-training and post-training periods.

Although the index of similarity between reference tests and designed test case by the participants is not high, this is due to the occurrence of the redundant or inefficient designed test cases by software testers. This evidence demonstrates a demand for improvements in training, since the software development scenario is dynamic and agile, designing redundant or inefficient test cases can compromise the test activity.

7 CONCLUSION AND FUTURE WORK

Gamification has been investigated in different application domains due to the several benefits it provides. It has been contributed to the training of software engineering contents [1, 21, 23]. This paper has investigated whether gamification can contribute to software testing training since there is a lack of qualified professionals in this area and a lack of motivation to work in this position at companies. For this, we carried a preliminary study to check whether gamification increasing the motivation of software testing to learn testing contents and also to check whether it contributes to the accuracy and precision of the designed test cases by them.

As a result, we concluded that gamification contributes positively to increasing the motivation of software testers while they received a gamified training due to, mainly the competitiveness created from gamification elements inserted in proposed training. We also noted that gamification contributes to improving the quality of the designed test cases by them, specifically regarding the accuracy and precision of such test cases. We observed such facts post software testers received the gamified training as shown in Section 5.2. Although we observe such effects, we can not generalize these results. Therefore, conclusions obtained from this work are insights to investigate the benefits of gamification in software testing training from the industry perspective.

We did not note the adverse effects of gamification in the study. We believe that the stimulus of the beginning of a career may have mitigated the conflicting influences of gamification. We observed that gamification provided interaction among participants, and this may have contributed to the learning process. We also noted contributions to stress reduction, making the professional environment more pleasant. In future work, we intend to replicate this study with more software testers to check whether the results can be generalized. Also, we intend to address structural and defect-based testing techniques in future work.

REFERENCES

[1] M. M. Alhammad and A. M. Moreno. 2018. Gamification in software engineering education: A systematic mapping. *Journal of Systems and Software* 141 (2018), 131 – 150.

[2] J. O. Andrade and M. Canese. 2013. A Gamified Web System for Learning Formal Logic (portuguese). In *Capa, prefácio e comitês do SBIE 2013*. XXIV Brazilian Symposium on Computers in Education, Campinas, Brazil, 426.

[3] Andrés Francisco Aparicio, Francisco Luis Gutiérrez Vela, José Luis González Sánchez, and José Luis Isla Montes. 2012. Analysis and Application of Gamification. In *International Conference on Interacción Persona-Ordenador* (Elche, Spain). Association for Computing Machinery, New York, NY, USA, Article 17, 2 pages.

[4] J. Bell, S. Sheth, and G. Kaiser. 2011. Secret Ninja Testing with HALO Software Engineering. In *4th International Workshop on Social Software Engineering* (Szeged, Hungary). ACM, New York, NY, USA, 43–47.

[5] J. Bishop, R. N. Horspool, T. Xie, N. Tillmann, and J. Halleux. 2015. Code Hunt: Experience with Coding Contests at Scale. In *37th International Conference on Software Engineering* (Florence, Italy). IEEE Press, Piscataway, NJ, USA, 398–407.

[6] E. M. Bizerra, D. S. Silveira, M. L. P. M. Cruz, and F. J. A. Wanderley. 2012. A Method for Generation of Tests Instances of Models from Business Rules Expressed in OCL. *Latin America Transactions* 10, 5 (2012), 2105–2111.

[7] L. Copeland. 2004. *A Practitioner's Guide to Software Test Design*. Artech House, Inc., USA.

[8] M.E. Delamaro, J.C. Maldonado, and M. Jino. 2016. *Introdução ao teste de software* (2 edição ed.). Elsevier, Rio de Janeiro.

[9] D. Dicheva, C. Dichev, G. Agre, and G. Angelova. 2015. Gamification in education: a systematic mapping study. *Educational Technology & Society* 18, 3 (2015), 1–14.

[10] A. Dignan. 2011. *Game Frame: Using Games as a Strategy for Success*. Free Press, Avon, MA.

[11] A. Domínguez, J. Saenz-De-Navarrete, L. De-Marcos, L. FernáNdez-Sanz, C. PagéS, and J. J. MartíNez-HerráIz. 2013. Gamifying learning experiences: Practical implications and outcomes. *Computers & Education* 63 (2013), 380–392.

[12] G. Fraser. 2017. Gamification of Software Testing. In *2017 IEEE/ACM 12th International Workshop on Automation of Software Testing (AST)*. IEEE, Buenos Aires, Argentina, 2–7.

[13] G. Fraser and A. Arcuri. 2013. Whole test suite generation. *IEEE Transactions on Software Engineering* 39, 2 (2013), 276–291.

[14] Y. Fu and P. J. Clarke. 2016. Gamification-based cyber-enabled learning environment of software testing. In *Annual Conference & Exposition*. ASEE, New Orleans, Louisiana, 1–16.

[15] J. Hamari, J. Koivisto, and H. Sarsa. 2014. Does Gamification Work? – A Literature Review of Empirical Studies on Gamification. In *47th Hawaii International Conference on System Sciences*. IEEE, Waikoloa, HI, USA, 3025–3034.

[16] G. M. de Jesus, L. N. Paschoal, F. C. Ferrari, and S. R. S. Souza. 2019. Is It Worth Using Gamification on Software Testing Education? An Experience Report. In *XVIII Simpósio Brasileiro de Qualidade de Software*. ACM, Curitiba, 6:1–6:19.

[17] K. M. Kapp. 2012. *The gamification of learning and instruction: game-based methods and strategies for training and education*. John Wiley & Sons, Hoboken, Nova Jersey.

[18] T. Karras, S. Laine, M. Aittala, Ja. Hellsten, J. Lehtinen, and T. Aila. 2019. Analyzing and Improving the Image Quality of StyleGAN. arXiv:1912.04958 [cs.CV]

[19] J. M. Keller. 1987. Development and use of the ARCS model of instructional design. *Journal of instructional development* 10, 3 (1987), 2–10.

[20] M.A. Khan and M. Sadiq. 2011. Analysis of black box software testing techniques: A case study. In *International Conference and Workshop on Current Trends in Information Technology (CTIT)*. IEEE, Dubai, United Arab Emirates, 1–5.

[21] O. Pedreira, F. García, N. Brisaboa, and M. Piattini. 2015. Gamification in software engineering – A systematic mapping. *Information and Software Technology* 57 (2015), 157 – 168.

[22] M. R. Prensky. 2012. *From digital natives to digital wisdom: Hopeful essays for 21st century learning*. Corwin Press, California, USA.

[23] W. Qu, Y. Zhao, M. Wang, and B. Liu. 2014. Research on teaching gamification of software engineering. In *9th International Conference on Computer Science Education*. IEEE, Vancouver, 855–860.

[24] J. M. Rojas and G. Fraser. 2016. Code Defenders: A Mutation Testing Game. In *2016 IEEE Ninth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, Chicago, 1–6.

[25] L. P. Scatalon, J. M. Prates, D. M. de Souza, E. F. Barbosa, and R. E. Garcia. 2017. Towards the role of test design in programming assignments. In *30th Conference on Software Engineering Education and Training*. IEEE, Indianapolis, 170–179.

[26] S. Smith-Robbins. 2011. "This game sucks": How to improve the gamification of education. *EDUCAUSE review* 1 (2011), 1–10.

[27] S. B. Sousa, V. H. S. Durelli, H. M. O. Reis, and S. Isotani. 2014. A systematic mapping on gamification applied to education. In *29th Annual ACM Symposium on Applied Computing*. ACM, Gyeongju, 216–222.

[28] A. M. Toda, P. H. D. Valle, and S. Isotani. 2017. The dark side of gamification: An overview of negative effects of gamification in education. In *Researcher Links Workshop: Higher Education for All*. Springer, Cham, 143–156.

[29] P. H. D. Valle, E. F. Barbosa, and J. C. Maldonado. 2015. CS curricula of the most relevant universities in Brazil and abroad: Perspective of software testing education. In *XVII International Symposium on Computers in Education (SIIE)*. IEEE, Setúbal, Portugal, 62–68.

[30] P. H. D. Valle, F. B. Francine, and J. C. Maldonado. 2015. Um mapeamento sistemático sobre ensino de teste de software. In *XXVI Simpósio Brasileiro de Informática na Educação*. SBC, Maceió, Brasil, 71 – 80.

[31] P. H. D. Valle, A. M. Toda, E. F. Barbosa, and J. C. Maldonado. 2017. Educational Games: A Contribution to Software Testing Education. In *47th Annual Frontiers in Education (FIE)*. IEEE, Indianapolis, 1–10.

[32] P. H. D. Valle and M. C. A. Todorov. 2020. Gamificação Aplicada à Gestão do Conhecimento em Projetos: Um Mapeamento Sistemático. *RENOTE* 18, 10 (2020), 1–10.

[33] C. Wohlin, P. Runeson, M. Host, M. C. Ohlsson, B. Regnell, and A. Wesslen. 2012. *Experimentation in software engineering*. Springer Science & Business Media, Springer.