Educational Games: A Contribution to Software Testing Education

Pedro Henrique Dias Valle, Armando Maciel Toda, Ellen Francine Barbosa, José Carlos Maldonado
Institute of Mathematics and Computer Science, University of Sao Paulo (ICMC-USP)
São Carlos/SP, Brazil, 13560-970
Email: pedrohenriquevalle@usp.br, armando.toda@gmail.com, francine@icmc.usp.br, jcmaldon@icmc.usp.br

***Abstract*—Software testing is a relevant activity to provide evidence of qualifty of software products. However, there is a lack of qualified professionals in this area. This can be caused due to difficulty in teaching software testing through approaches that use only theoretical classes and traditional tools. In addition, there is a lack of motivation due to the work environment and the strategies of allocation and responsibility of these professionals in development and testing teams. To mitigate these problems, approaches have been used to support software testing education, such as: educational games, integrated teaching of software testing with programming, educational modules, among others. The objective of this paper was to develop an educational game named Testing Game, addressing the following topics: functional testing, structural testing and mutation testing. To support the development of the Testing Game, we performed a systematic mapping aiming at selecting a game engine. To evaluate the efficiency of the game, we conducted a feasibility study to evaluate the quality regarding motivation, user experience and learning from the point of view of the students. We also evaluate the usability of the Testing Game. Approximately 85.64% of people who participated in the study assessed the quality of the game in a positive perspective regarding motivation, user experience and learning from the point of view of the students. Regarding the usability of the game, students identified minor problems. Through this work, we realize that the game Testing Game can constitute as a complementary resource to support software testing education.**

***Keywords*—Educational Games; Software Testing Education; Software Testing Techniques**

I. Introduction

In recent years, software industry improved the development of systems and high quality products. Therefore Verification, Validation and Testing activities have also been improved, especially software testing activities, using techniques, criteria, and tools to perform tests [1]. Software testing aims at executing programs or models, based on specific inputs that analyze if these products behave as expected, leading to a detailed analysis of the software, aiming at identifying failures and then, through debugging, eliminating the faults that originate the failures [1]. Software testing is recognized as an essential activity in any successful software development process [2].

Despite software testing being recognized as a relevant activity in quality assurance of software products, many students feel unmotivated to learn contents related to software testing [3]. This lack of motivation may be related to: i) the inconsistency between practical and theoretical contents that decrease the interest among students; ii) contents that are taught in the classroom are not the same that are required in the industry; iii) students have difficulties in understanding the process and the stages of tests; and iv) the absence of student experience in software development. As a result of this demotivation, test industry has found a lack of specialized professionals in this area [4, 5]. In this scenario, some researchers observed that it is essential to develop instruments that facilitate the teaching and training of software testing, in order to increase motivation to work with the software testing [6].

To know how the software test professionals are trained Valle, Barbosa and Maldonado [7] analyzed the reference curriculum proposed by Brazillian Society of Computing (SBC) and Association for Computing Machinery (ACM) for Computer Science courses. They observed, with few excpetions, that in the CS courses, either in Brazil or abroad, software testing is addressed only in the disciplines of Software Engineering and/or Programming Fundamentals, and few lectures are assigned to the software testing education, providing students only an overview of the contents of this area [7].

To mitigate these problems, some methodologies and tools have been used to facilitate software testing education [5]. To characterize the main approaches used in software testing education Valle, Barbosa and Maldonado (2015) [5] carried out a systematic mapping in wich they identified 11 different initiatives to support software testing education, as: Educational Modules, Educational Games, Test Driven Development (TDD) and Integrated Teaching of Software Testing and Programming [5]. In particular, educational games in the domain of software testing have been widely investigated because of the benefits these tools provide. The authors identified seven different educational games: U-TES [8], Jogo das 7 Falhas [9], iTest Learning [10], iLearnTest [11] and JoVeTest [12]. In general, these games addressed only the Functional Test technique and do not addressed contents regarding to Structural and Defect-Based Testing techniques. Thus, we perceived the importance in developing new educational games to address other techniques and criteria of software testing.

Based on the above, the aim of this paper is to develop an educational game to support software testing education regarding functional test, structural test and mutation test. This paper is organized as follows: Section II present the methodology used in the development of the game. Section III, explain an overview of the Testing Game, as phases and dynamics of the game. Section IV present the feasibility study

978-1-5090-5920-1/17/$31.00 ©2017 IEEE

performed to evaluate the quality and usability of the Testing Game. Finally, Section V present the conclusions and future work.

II. Methodology

To mitigate problems presented in Section I, we developed a game named **Testing Game**. In this context, we divided this process in three steps: **Literature Review** and **Systematic Mapping**, **Implementation of the Testing Game** and **Evaluation of Testing Game**. An overview of the methodology can be seen on Figure 1.

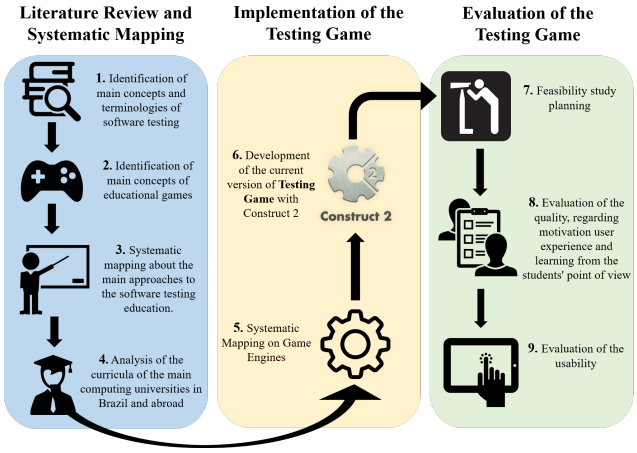

Figure 1. Methodology used for the development of the Testing Game

In the first step, we conducted a Literature Review to identify the main concepts and research related to software testing and educational games. To characterize the state of the art regarding software testing education, we carried a Systematic Mapping on the main approaches to support the software testing education [5]. As a result, we identified 11 initiatives to support software testing education. Among them, there are "educational games" and "integrated teaching of software testing and programming".

Therefore, we performed the second step related to the implementation of the Testing Game. To support this implementation, we carried a new Systematic Mapping to identify game engines that are available in the literature. So we developed the game using the game engine named Construct 2 [1]. Finally, we performed the third step of the methodology that was the evaluation of the Testing Game. This evaluation was performed using an experimental methodology defined by Shull, Carver and Travassos (2001) [13]. This methodology is composed of four stages that evaluate software products from its first version until its use in the industry. The feasibility study is the first step of this methodology. We perform this study to verify if the Testing Game is feasible and if there is a possibility in continuing the development of this game. This feasibility study was divided in two stages: the first to evaluate the quality of the game regarding motivation, user experience and learning from the point of view of the students and the second step to evaluate the usability of the Testing Game.

[1] Available in: https://www.scirra.com/construct2

III. An Educational Game for the Software Testing Education

From the scenarios presented in the section I, we developed a game, named Testing Game, to support the software testing education. This game was developed for the web platform. The target of the Testing Game is undergraduate students who study subjects containing content related to software testing. The game can be used as a support material for software testing classes because students can train their skills, that were acquired in the classroom. Therefore, this game can be considered a complementary material to the teaching of software testing contents.

The game addresses three software testing techniques, namely: Functional, Structural, and Based on Defects [1, 14]. To demonstrate the application of the criteria considered in the game, we used the program BubbleSort written in Java. In Table I we present the content addressed in the game for each test technique that was considered.

Table I. Contents addressed in Testing Game

Software Testing Techniques	Contents
Functional Test	Software Testing Criteria, Specification of BubbleSort Program, Equivalence Partitioning Criterion, Equivalence Table, Boundary Value Analysis Criterion and Set Minimum of Test Case.
Structural Test	Control Flow Graph, All-Nodes Criterion, All-Edges Criterion, Uses of Variable (definition, computation-use, predicate-use), Def-Use Graph, All-Definitions Criterion, All-Uses Criterion, Set Minimum of Test Case and Non-Executable Path.
Defect-Based Test	Mutation Operators in Java, Mutation, Equivalent Mutants, Mutation Score and Set Minimum of Test Case.

As presented in the Table I, the Testing Game uses an incremental test strategy. In this strategy, it is applied "weaker" criteria which may be less effective to the adequacy evaluation of the test cases set. Then, "stronger" criteria are applied incrementally, and possibly more effectively, but in general more expensive. However, the Testing Game present some limitations, such as: i) Use only a single program in the game; ii) It is not coupled with software testing tools; and iii) The Bubble Sort program, which was used in the game, does not contain known defects.

After defining the testing contents that were addressed in the game, we defined technical needs, which are the particularities required in educational games, as: programming and database. From the design point of view, the Testing Game is a game for the web platform with 2D graphic dimension. In this game, the student is represented by an avatar who can perform many actions during the game, such as: run, walk, jump, deflect and eliminate enemies, among other actions.

The game contains three levels which are represented by three different doors. Each level corresponds to a software test technique as shown in Figure 2. To open the doors, it necessary to use the keys that are made available during the game. Each level of the game is represented by a test technique that contains different phases. The functional test contains 8 phases, the structural test contains 10 phases and the defect-based test contains 5 phases. These phases are represented in the game by means of padlocks. To release a padlock, it is necessary to successfully execute the previous phase.

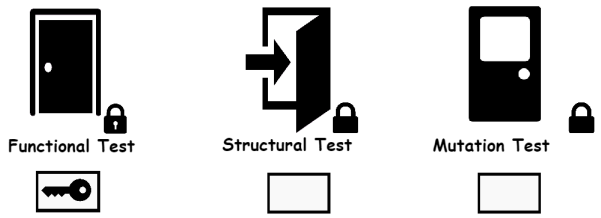

Figure 2. Doors that represent the levels of the Testing Game.

In each phase, there are different challenges regarding the addressed contents. At the beginning of each phase there is a description that inform the students what they must do to succeed in the phase. Another functionality is related to the theoretical content of software testing. For each new content considered in the game, there is a module in which the user can access and read about the new content. To access this feature, the student must click on an icon, represented by a book, that is located at the top of the screen on the right side. In the phases related to structural testing, students can access the code of the tested program by clicking on the icon represented by a magnifying glass.

In Figure 3 is shown the second phase of Level 1 of the Testing Game. In this phase, the specification of the Bubble Sort program is presented to students so that they apply the concepts of software testing during the execution of the game. According to the specification of Bubble Sort program, the program must only accept vectors with the size equal to 4. In this phase, students should find the vectors with the size equal to 4. For this purpose, they must eliminate the enemies and the invalid vectors. After students complete the challenge, a list of valid and invalid vectors is displayed on the screen and the phase is finished.

Figure 3. Second phase of functional test.

The second level of the Testing Game consider contents related to the structural test. In Figure 4 is shown the first phase of the second level of the Testing Game. In this phase the code of Bubble Sort program is presented to the student. After students observed the code, they must find the GFC that represents the control flow graph corresponding to the code of Bubble Sort program. In this context, students must eliminate the GFCs that do not correspond to the program and must eliminate the enemies during the challenges that are proposed. In this phase, a theoretical content module is available, related to structural test. This content can be accessed by clicking on the icon that represents a book, fixed at the top-right position of the screen.

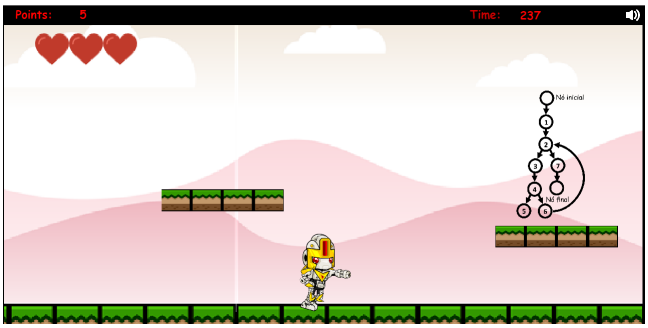

Figure 4. First phase of structural test.

The eighth phase of the second level of the game contains a different dynamics as shown in Figure 5. In this phase, students must observe the Def-Use graph of the Bubble Sort program and fill in a table that demonstrates in which nodes of the graph the variables were defined.

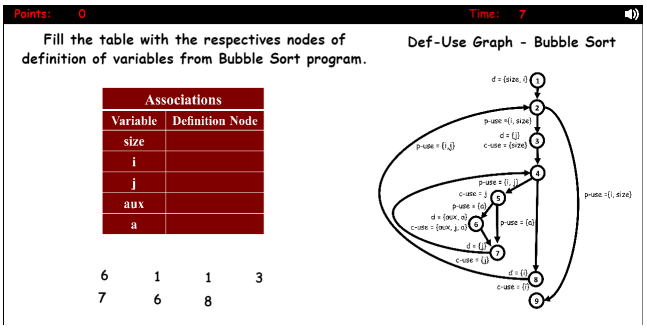

Figure 5. Eighth phase of structural test.

Finally, Figure 6 shows an example that consideres content related to mutation test. In this phase, students must eliminate mutant programs that are considered equivalent mutants. For this purpose, students must eliminate enemies, obstacles and mutants that are not equivalent mutants. To facilitate the understanding of the considered contents, we provide a theoretical content module about equivalent mutants. This content can be accessed by clicking on the icon that is represented by a book which, is fixed at the top-right position of the screen. It is relevant to note that the code location that containes a mutation is marked in red.

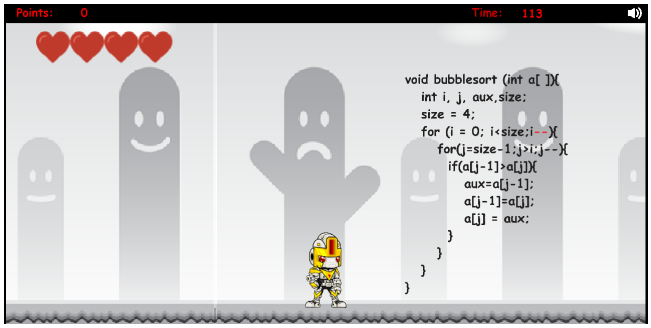

Figure 6. Fourth phase of Mutation Test.

Testing Game is a game that uses an incremental approach to software testing education, first it considers the functional

test, then the structural test, and finally the mutation test. In addition, it is important to note that the Testing Game combines two of the most used approaches to support the software testing education, according to the systematic mapping performed by Valle, Barbosa and Maldonado (2015) [5] since this game considers the Integrated Teaching of Software Testing and Programming.

IV. Quality and Usability Evaluation of the Testing Game

Educational games are tools to support the teaching of educational contents, attractive and dynamic manner. However, before an educational game is used, it is necessary to evaluate if it can support the teaching of the proposed contents. For this purpose, it is necessary to perform an evaluation of the game quality [15].

A. Feasibility Study Planning

To verify the quality and usability of the Testing Game, we used the feasibility study planning proposed below. The planning of the feasibility study contains research questions, objectives, selection of subjects and instrumentation. In this section, it contains the feasibility study conducted to evaluate the Testing Game. Following, planning and execution of the feasibility study and the analysis of the results are discussed.

1) Research Questions: : In this subsection we present our research questions to evaluathe the quality and usability of the Testing Game.

- RQ_1: Does the Testing Game have good quality regarding motivation, user experience and learning, from students' point of view?
- RQ_2: Does the Testing Game have good usability from the student's point of view?

2) Objective: This subsection presents the objective of the performed feasibility study, according to the GQM (Goal, Question, Metric) template [16]. This template is commonly used to express the goal of a software engineering study.

Analyze the Testing Game
for the purpose of evaluate it
with respect to quality and usability
from the point of view of the of students
in the context of graduate students in Software Engineering, specifically Software Testing.

3) Selection of Subjects: To perform our evaluation, we invited graduate students (master's and doctorate degree) from the Software Engineering Laboratory of the Institute of Mathematical and Computer Sciences (ICMC) of the University of São Paulo (USP).

The subjects who participated in the feasibility study have deep knowledge in the area of software testing. The subjects that participated in the feasibility study did it voluntarily. To participate in this study, subjects had to:

- Manifest interest in participating in the study and sign a consent form;
- Fill a characterization profile form to check the knowledge of the student on each topic;
- Participate in training of software testing concepts, educational games and heuristic evaluation; and
- Answer the feedback questionnaire.

4) Instrumentation: This subsection presents the description of the instrumentation used in the feasibility study:

- **Consent Form:** The subjects signed a form to manifest their interest in participating in the feasibility study, voluntarily.
- **Educational Game:** We use the Testing Game to perform the feasibility study. This game is available as a web application and addresses the three software testing techniques.
- **Feedback Questionnaire:** This questionnaire is used to evaluate the training, structure of the feasibility study and to identify suggestions to the study.
- **Characterization of Profile form:** This form aims characterizing the level of knowledge of the subjects. This questionnaire contains questions on: software testing, Java programming language, educational games, evaluation of educational games and heuristic evaluation.
- **Training Material:** We used as training material contents related to software testing, educational games, and heuristic evaluation. In addition, we used a didactic note on software testing.
- **Evaluation Model of Educational Games:** We used the educational game evaluation model proposed by Savi, Wangenheim and Borgatto (2011) [17]. This model has 37 questions that evaluate three dimensions: motivation, user experience, and learning.
- **Usability Heuristics:** We used the set of heuristics proposed by Nielsen (1994) [18] to evaluate the usability of the Testing Game.

B. Execution of the Feasibility Study

1) Execution of Quality Evaluation of the Testing Game: In this evaluation, we invited 15 students to participate in the study. Most students were male (86%) and all students were graduate students (8 master's students and 7 doctoral students). From the characterization of profile form, we identified the profile of the students who participated in this study. Table II shows the percentages (%) of the knowledge levels in software testing and educational games of the students who participated in the evaluation of the quality of the Testing Game.

Table II. Profile of students who participated in the quality assessment of the Testing Game

Knowledge Level	Software Testing (%)	Educational Games (%)
No Knowledge	0%	6,7%
Basic	26,7%	47,6%
Medium	46,7%	20%
Advanced	26,7%	26,7%

Regarding knowledge in educational games, 47.6% of the students have basic knowledge, 26.7% advanced, 20% medium and 6.7% do not have knowledge in educational

games. Regarding knowledge in software testing, 46.7% of the students have a medium knowledge, 26.7% advanced and 26.7% have basic knowledge in software testing. It is important to note that only 13.3 % of the students studied a specific discipline of software testing at an undergraduate degree. In addition, only 6.7% of the students have experience in software testing in industry, specifically, more than 3 years of experience. Only 13.7 % of the students had already used the evaluation model of educational games proposed by Savi, Wangenheim and Borgatto (2011) [17]. However, most of the students had already participated in evaluations of games.

After the students answered the characterization of profile form, we conducted a training, so that the students were able to participate in the study as mentioned before. Finally, the students played the game and used the evaluation model proposed by Savi, Wangenheim and Borgatto (2011) [17] to evaluate the quality of the game. Then, students filled a feedback questionnaire to authorize the use of their data in the study. In addition, we collect information on criticisms and suggestions to the game.

2) Execution of Usability Evaluation of the Testing Game: In this evaluation, we invited 5 students to participate in the heuristic evaluation. These students were selected from the set of subjects who participated in the study that evaluated the quality of the game. From the characterization of profile form, we identified the profile of the students who participated in this study. Table III shows the percentages (%) of the knowledge level in software testing and educational games of the students who participated in the heuristic evaluation.

Table III. PROFILE OF STUDENTS WHO PARTICIPATED IN THE USABILITY ASSESSMENT OF THE TESTING GAME

Knowledge Level	Software Testing (%)	Educational Games (%)
Basic	60%	0%
Medium	20%	60%
Advanced	20%	40%

Regarding knowledge in educational games, 60% of the students have medium knowledge and 40% advanced. However, all students have already participated in an evaluation of educational games. Regarding knowledge in software testing, 60% of the students have a basic knowledge, 20% medium and 20% advanced. Regarding the heuristic evaluation, 80% of the students have already performed some heuristic evaluation. However, all of them attended a specific discipline of software testing in graduation course. In training, students performed a heuristic evaluation in another educational game to learn to use this evaluation technique.

Then, students played the Testing Game and used the set of heuristics proposed by Nielsen (1994) [18] to identify usability problems. Finally, students answered a feedback form on the training, structure of the experiment and critics/suggestions for the game. Overall, the students said that the training was appropriate and the experiment was well structured.

C. Analysis of Results

In this section, we present the results obtained in the feasibility study that was performed to evaluate the usability and quality of the Testing Game.

1) Result of Quality Evaluation of the Testing Game: To evaluate the quality of the Testing Game, we analyzed the motivation, the players' experience and learning. For this purpose, students answered the questionnaire using the Likert scale from -2 to +2, where -2 stated "Strongly Disagree" to +2 as "Strongly Agree". Therefore, it is possible to identify the positives and negatives points of the game from the point of view of the students.

The results related to **Motivation** are presented in Figure 7. Approximately 86.6% of the students evaluated the motivation of the game positively. In Figure 7, we observe that 80% of the students were satisfied with the game. However, 20% of the students had difficulty advancing through the game with their own knowledge. Regarding *confidence*, 90% of the students stated that the game was an easy to understand educational resource, promoting confidence. The *relevance* item obtained the best result of the evaluation in which approximately 91.13% of the students evaluated positively, in other words, the game is appropriate for learning of contents related to software testing. Finally, the *attention* item was evaluated positively by 84.5% of the students. Therefore, we observe that the Testing Game obtained a good evaluation regarding the motivation of the students.

The results related to **User Experience** are presented in Figure 8. Approximately 87.7% of the students evaluated the user experience of the game positively. In the *competence* item, approximately 83.35% students positively evaluated this item because they performed the tasks of the game through their own abilities. Regarding the *fun*, approximately 88.32% of the students evaluated positively. They said they would recommend the game to their colleagues and that they would like keep playing the game. Regarding the *challenge* item, approximately 93.15% of students evaluated positively. This item had the best score in the User Experience category. Finally, 84.46% of students positively evaluated the *immersion* of the game because they felt part of the game and temporarily forgot their day-to-day tasks while they were playing it.

Finally, results related to **Learning** are presented in Figure 9. Approximately 82.23% of the students evaluated the learning of the game positively. It is important to note that the learning item had no negative evaluation. The students stated that the Testing Game can contribute to their experiences in professional life. In addition, they said that the game contributed to their learning in software testing and the game was more efficient than other traditional activities already used to software testing education.

2) Result of Usability Evaluation of the Testing Game: The heuristic evaluation is based on a set of criteria to identify usability problems in interfaces. These criteria are used by experts to visually check an interface and analyze it [18]. The set of heuristics that we used to evaluate the usability of the game can be observed in the Table IV.

Therefore, we performed a heuristic evaluation in which we identified 7 usability problems in the Testing Game. The H03 is the heuristic that identified more usability problems, shown in Figure 10. The identified usability problems are listed in Figure 10 along with their respective rate, and the severity that were suggested by the experts. In addition, during the execution of the study, we observed that the students remained attentive to

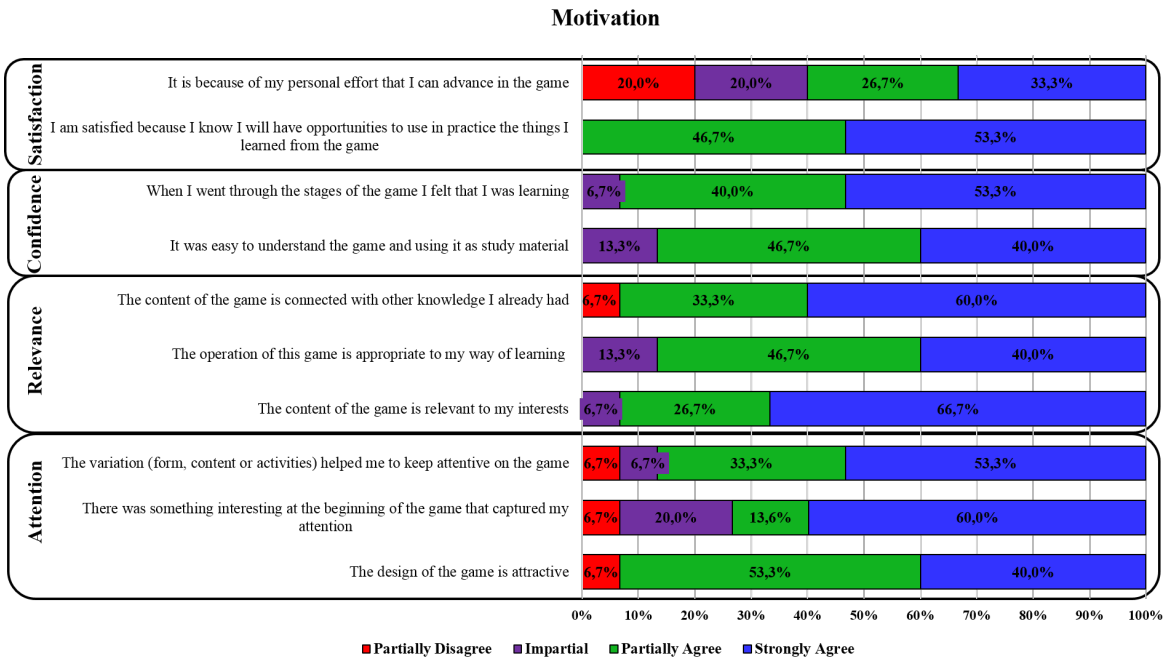

Figure 7. Result of Evaluation of the Testing Game Regarding Motivation

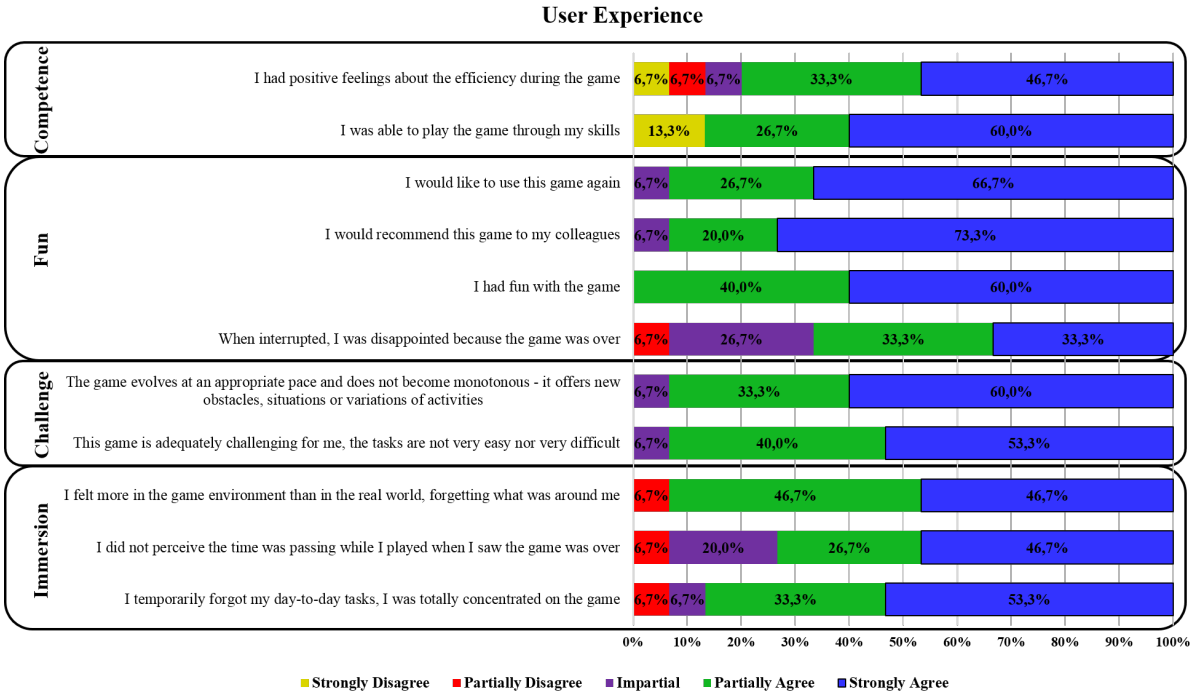

Figure 8. Result of the Evaluation of the Testing Game Regarding User Experience

the challenges proposed in the game.

In general, experts have indicated that it is necessary to have an option to exit the game. In the current version, students can only exit the game when they complete a phase. Another problem indicated by the experts is regarding the possibility of the students to consult the specification of the program tested in the game and to consult the description of the activities in real time. In the game, this action can only be performed at the beginning of each phase. The experts identified that it is necessary to indicate in the game the completed phases, the main errors and the correct answers of the students. Finally, experts have identified the need to create an explanation of the games' commands for performing some actions, as: jumping, running, and shooting.

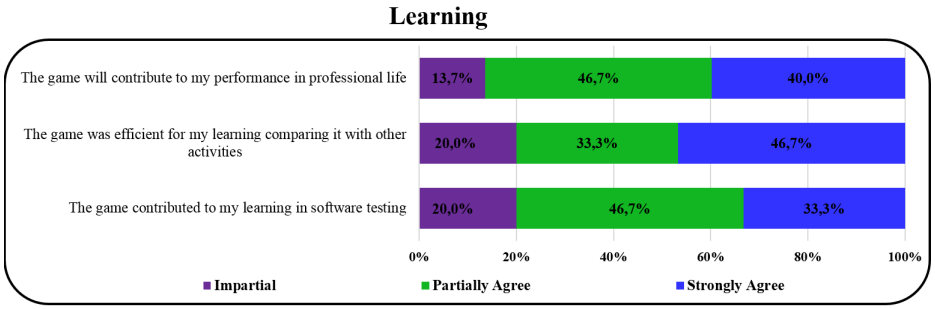

Figure 9. Result of Evaluation of the Testing Game Regarding Learning

Table IV. USABILITY HEURISTICS

Heuristic	Description
H01	Visibility of system status
H02	Match between system and the real world
H03	User control and freedom
H04	Error prevention
H05	Help users recognize, diagnose, and recover from errors
H06	Consistency and standards
H07	Recognition rather than recall
H08	Flexibility and efficiency of use
H09	Aesthetic and minimalist design
H10	Help and Documentation

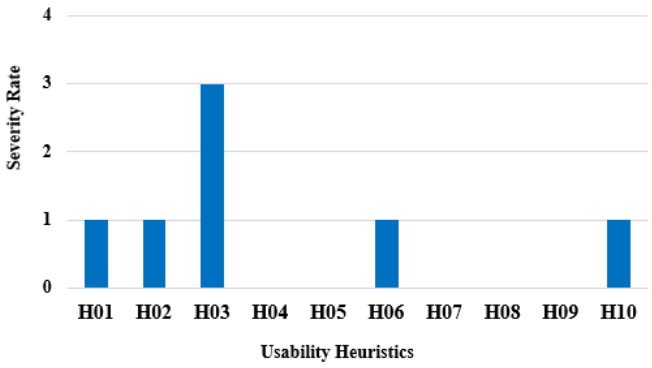

Figure 10. Frequency of problems identified by heuristics in the Testing Game

D. Discussion of Results

Through the results obtained in the feasibility study, we identified answers to the research questions proposed in the feasibility study planning. The RQ_1 is related to the quality of the Testing Game regarding motivation, user experience and learning from the students' point of view. From the obtained results, we observed that the Testing Game has a good quality from the students' point of view. Approximately 86.9% of students evaluated the **Motivation** item positively. The students said that they were satisfied while playing and that the Testing Game was easy to understand, stating that it is appropriate for learning software testing.

The **User Experience** item also had a positive evaluation according to 87.7% of the students who participated in the study. Students learned about software testing with fun aspects. These students said they would like play Testing Game again and would recommend this game to their colleagues because it presents challenging dynamics and that they felt part of the game, in other words, they felt immersed within the game. Finally, 82.33% of the students evaluated the Learning positively, they said that the game contributed to software testing education. In general, the arithmetic mean of the items evaluated, namely: **Motivation**, **User Experience** and **Learning**, indicates that 85.64% of students evaluated the quality of the game positively.

Table V. LIST OF USABILITY PROBLEMS IDENTIFIED IN THE HEURISTIC EVALUATION OF THE TESTING GAME

Heuristic	Usability Problems	Severity Rate
H01	There is no indication that differs the phases that have been completed from the phases that were open and have not been completed	2
H02	Students do not understand that the key should be dragged to the door	4
H03	It is not possible to get out of a phase.	2
H03	It is not possible to consult the program specification and the description of the activities during the execution of a phase.	3
H03	It is not possible to navigate between phase menus and there is no option to return to the main menu	3
H06	It is not possible to know the errors and correct answers of the students in each phase	3
H10	There is no explanation on the game commands	4

The feasibility study also evaluated the usability of the game as proposed in RQ_2. The experts identified 7 usability problems through 5 heuristics. However, we have identified some usability problems in the game, such as lack of explanation on game commands, lack of error feedback, lack of menus for navigating between screens, among other problems. During the evaluation of the game and through the feedback questionnaire, we collected suggestions for improvements for the game. These suggestions for improvements will be developed in the next versions of the game.

V. CONCLUSIONS AND FUTURE WORK

Software testing is a relevant activity to provide evidences of software products quality. However, there is a lack of qualified professionals in this area. In addition, students feel a lack in motivation to learn contents related to software testing. To mitigate this, we have developed an educational game to support software testing education, named Testing Game. This game considers the functional, structural, and defect-based testing and it is available for web platforms.

Through the feasibility study performed, we observed that the Testing Game have good quality regarding motivation, user experience and learning from the students' point of view. It is important to say that the learning category that we evaluated did not verify the learning level of the students with

the use of the game, but verified the students' opinion on whether the game could contribute to their learning. We also evaluated the usability of the game through a set of usability heuristics proposed by Nielsen [18]. This paper contributed to the identification of the state of the art regarding the software testing education, and motivated the creation of a "software testing culture" among students, providing an increase in students' interest to learn contents related to software testing.

In general, we concluded that Testing Game has a good usability and quality. Based on our results, we believe that this game can be used as an instrument to support the software testing education. As future works, we intend to conduct a case study to evaluate students' performance by applying this game in a software testing course. We also intend to consider implementing multiplayer features and integrate software testing tools in future updates for those students can test the programs within the game.

VI. Acknowledgments

We thank CNPq and CAPES for the financial support.

References

[1] M. Delamaro, J. Maldonado, and M. Jino, *Introdução ao teste de software*, 2nd ed. Rio de Janeiro: Elsevier, 2016.

[2] G. Fraser and A. Arcuri, "Whole test suite generation," *IEEE Transactions on Software Engineering*, vol. 39, no. 2, pp. 276–291, 2013.

[3] S. Jia and C. Yang, "Teaching software testing based on cdio," *World Transactions on Engineering and Technology Education*, vol. 11, no. 4, 2013.

[4] J. C. de Souza, D. M.; Maldonado and E. F. Barbosa, "Aspectos de desenvolvimento e evolução de um ambiente de apoio ao ensino de programação e teste de software," in *XXIII Simpósio Brasileiro de Informática na Educação*. Rio de Janeiro, Brasil: SBC, 2012.

[5] P. H. D. Valle, E. F. Barbosa, and J. C. Maldonado, "Um mapeamento sistemático sobre ensino de teste de software," in *XXVI Simpósio Brasileiro de Informática na Educação*. Maceió, Brasil: SBC, 2015, pp. 71 – 80.

[6] W. Wong, A. Bertolino, V. Debroy, A. Mathur, J. Offutt, and M. Vouk, "Teaching software testing: Experiences, lessons learned and the path forward," in *XXIV Conference on Software Engineering Education and Training (CSEE&T)*. Honolulu, USA: IEEE, 2011.

[7] P. H. D. Valle, E. F. Barbosa, and J. C. Maldonado, "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)*. Setúbal, Portugal: IEEE, 2015, pp. 62–68.

[8] M. Thiry, A. Zoucas, and A. C. da Silva, "Empirical study upon software testing learning with support from educational game." in *XXIII International Conference on Software Engineering and Knowledge Engineering (SEKE)*. Miami Beach, USA: IEEE, 2011, pp. 481–484.

[9] L. L. Diniz and R. L. S. Dazzi, "Jogo das sete falhas: Um jogo educacional para apoio ao ensino do teste caixa preta," in *Computer on the Beach*. Florianópolis: Universidade do Vale do Itajaí, 2011, pp. 1–10.

[10] C. I. Bezerra, E. F. Coutinho, I. S. Santos, J. M. Monteiro, and R. M. Andrade, "Evolução do jogo itest learning para o ensino de testes de software: Do planejamento ao projeto," in *XIX Conferência Internacional sobre Informática na Educação (TISE)*. Fortaleza, Brasil: Nuevas Ideas En Informática Educativa, 2014.

[11] T. P. B. Ribeiro and A. C. R. Paiva, "ilearntest: Educational game for learning software testing," in *X Iberian Conference on Information Systems and Technologies (CISTI)*. Aveiro, Portugal: IEEE, 2015.

[12] A. K. T. Barbosa, L. L. E. Neves, and A. C. D. Neto, "Jovetest - jogo da velha para auxiliar no ensino e estudo de teste de software," in *IX Fórum de Educação em Engenharia de Software*. Maringá, Brasil: SBC, 2016.

[13] F. Shull, J. Carver, and G. H. Travassos, "An empirical methodology for introducing software processes," in *VIII European Software Engineering Conference Held Jointly and IX International Symposium on Foundations of Software Engineering (ACM SIGSOFT)*. New York, USA: ACM, 2001, pp. 288–296.

[14] G. J. Myers, C. Sandler, and T. Badgett, *The art of software testing*. New Jersey, USA: John Wiley & Sons, 2011.

[15] P. H. D. Valle, R. F. Vilela, P. A. P. Júnior, and A. C. G. Inocêncio, "Hedeg-heurísticas para avaliação de jogos educacionais digitais," in *XVIII Conferência Internacional sobre Informática na Educação (TISE)*. Porto Alegre, Brasil: Nuevas Ideas En Informática Educativa, 2013.

[16] V. Basili, J. Heidrich, M. Lindvall, J. Münch, M. Regardie, D. Rombach, C. Seaman, and A. Trendowicz, "Gqm+ strategies: A comprehensive methodology for aligning business strategies with software measurement," *Software Metrik Kongress*, 2007.

[17] R. Savi, C. Wangenheim, and A. Borgatto, "Um modelo de avaliação de jogos educacionais na engenharia de software," in *XXV Simpósio Brasileiro de Engenharia de Software (SBES)*. São Paulo: SBC, 2011, pp. 1–10.

[18] J. Nielsen, "Heuristic evaluation," *Usability inspection methods*, vol. 17, no. 1, pp. 25–62, 1994.