# Use of Games as a Software Reuse Teaching Strategy

**Diego Cardoso Borda Castro**[1,2], **Claudia Maria Lima Werner**[1]

[1]Programa de Engenharia de Sistemas e Computação - COPPE
Universidade Federal do Rio de Janeiro, Brasil

[2]Centro Federal de Educação Tecnológica do Rio de Janeiro, Brasil

{diegocbcastro, werner}@cos.ufrj.br

***Abstract. Introduction****: Software Reuse (SR) is a field rich in theoretical concepts that necessitates practical experience for optimal understanding. Nonetheless, the traditional approach to teaching SR is passive, lacking techniques that engage students or include practical elements in the subject. The use of games has been recommended as a solution for this issue and is already utilized in various Software Engineering disciplines; nevertheless, no games have been identified to teach SR.* ***Objective:*** *Therefore, this work aims to demonstrate the use of games for teaching SR.* ***Methodology or Steps****: The game created aims to teach software components, being one of the most used areas of SR in the job market, and was evaluated by 26 participants.* ***Results:*** *The evaluation demonstrated its pedagogical potential, showing that SR can be taught through games.*
***Keywords*** *Game, Learning, Game-Based Learning, Software Reuse, Software Component.*

## 1. Introduction

To improve attention and engagement, educators strive for innovative learning strategies that combine pleasure with education [Xexéo et al. 2013] such as e-learning, the use of projects in the classroom, inverted classroom, blended learning, gamification, and Game-Based Learning (GBL), the latter being the focus of this work. Games are visual, interactive, and practical, possessing characteristics that hold the user's attention, still being one of the main ways of distraction and pleasure [Xexéo et al. 2013].

Software Reuse (SR) is a discipline with a very extensive theoretical body, including topics such as software product lines, domain engineering or Component-Based Development (CBD) [Chueca et al. 2023] that require hands-on experience for effective learning. However, the conventional way of teaching SR is passive, through classes with slides and readings, what tends to be tiring/boring due to the many concepts involved, making the student feel less motivated and engaged during classes [Navarro et al. 2004].

Companies have already tried to use CBD in their development, but not in a systematic way. Studies indicate that the majority of real-world projects lack sophisticated SR approaches, and the lack of training and education is considered one of the most significant reasons for this failure [Niu et al. 2011]. Games are currently being employed by numerous software engineering subareas to address a variety of topics, such as software testing, project management, software modeling, and other related fields of study [Connolly et al. 2007]. Nevertheless, the area of SR lacks games of this type.

The study attempts to address the deficiencies in SR teaching through the use of games, with the objective of improving student engagement and motivation. The main goal is to transform students into competent developers who are well-prepared for entry into the job market and able to use the taught practices. In order to achieve this objective, a game has been proposed as a practical method of teaching componentization through quality metrics for reuse. The application of the game was evaluated, and it produced positive results.

The remainder of this paper is organized as follows: Section 2 briefly introduces and discusses concepts used throughout the paper, Section 3 describes the game developed with the purpose of teaching CBD, Section 4 demonstrates the results of the proposed game evaluation, and Section 5 concludes with some observations, limitations, and future work.

## 2. Theoretical foundation

Games can include different characteristics, among which the following stand out: they are voluntary social activities, they have an uncertain conclusion, one or more players may participate, players may influence the game's progression through their decisions, they may have conflicting objectives, and they are regulated and limited by rules, among other characteristics [Xexéo et al. 2013]. Due to the various characteristics that compose up a game, it can be as complex as well-known franchises like God of War or as simple as a quiz. It is worth noting that some authors do not consider quizzes, puzzles and similar entertainment activities as games, but as educational activities.

Considering the range of characteristics that a game may possess, models have been developed to assist game developers in structuring their ideas; Mechanics, Dynamics, and Aesthetics (MDA) and Objectives, Challenges, and Rewards (OCR) are two notable concepts. MDA is an approach to understanding games that tries to describe their characteristics through three components: [1] Mechanics are the essential components of a game, such as actions and rules; [2] Dynamics describe the behavior of mechanics acting on data inputs and outputs; and Aesthetics describe the desired emotion when the player interacts with the game [Hunicke et al. 2004]. OCR can be understood as the basic structure of a game, that is, what has to happen in a game, and can be divided into three stages: Objectives are the desires that are intended to be achieved with the game to be created; challenges are the actions that the player must accomplish to achieve the goals; and Rewards are the results players earn by solving a specific challenge [Guardiola 2016].

Games can be used for other purposes beyond entertainment. The focus of this work is on its use to facilitate the transmission of educational content, the so-called "serious games" or "games with purposes" [Maloney et al. 2010]. Serious games have several sub-branches, according to their gameplay (pattern defined through the game rules), objective, and audience. Among them, stand out advergames, health games, business games, and game-based learning, which is the focus of this paper [Djaouti et al. 2011].

GBL can be understood as the use of games with an educational purpose to optimize the learning experience by using game characteristics such as simulations, error-based learning, and problem-solving [Djaouti et al. 2011]. Considering this, various educational activities, including quizzes and puzzles, are converted into games,

incorporating elements such as time constraints, continuous feedback for correct or incorrect answers, and stages dependent on the score obtained [Nascimento e Leite 2022].

Games have a list of characteristics that support teaching; among them, the following stand out: they are based on objectives, have rules that must be adhered to, capture the player's attention, and provide a simulated and motivational environment that does not affect the real world [Djaouti et al. 2011]. Due to these characteristics, many highly cited researchers have been working on the topic of GBL [Gari e Radermacher 2018].

Games are a different and innovative opportunity to catch students' attention and improve retention of matters taught, and it has already been used in some SE disciplines, such as project management, software testing, software modeling, and software processes [Schafer 2017]. However, no evidence was found in the literature on applying games for teaching SR topics in primary, secondary, or tertiary studies [Gari e Radermacher 2018, García-Mireles e Morales-Trujillo 2020].

## 3. Reuse Blocks Game

Components are software units with well-defined interfaces and explicitly specified dependencies. They can have different sizes and be characterized in different ways, from a small piece of code, a software package, a web service, a module that encapsulates a set of functions, or it can even be as large as a system [Sametinger 1997]. A component is considered reusable if its functionality is shared across several applications [Sametinger 1997], that is, if it is used in more than one location within the same project or not. And that is exactly the objective of the game described in this section: to construct reusable software components.

The proposed game was called Reuse Blocks, and was inspired by the Scratch programming environment [Jordine et al. 2014] since it is one of the main current visual languages and already has several users who make use of this development platform [Maloney et al. 2010]. Reuse Blocks makes use of an interface and mechanics very similar to Scratch, using block programming. However, Scratch was developed with the aim of teaching programming to novice users [Jordine et al. 2014], and this game was developed to teach reuse through software components and be used by undergraduate students.

In this game, the user must create the components that are requested in the current phase based on the components/functions that are previously made available by the game, or from components that were created in previous phases. In the end, the component created by the player is evaluated based on metrics, and a score is generated.

The game's score is based on four metrics for the development of software that measure the reuse of a component, aiming at building better components so that the codes created are evaluated and scored according to the quality generated. A metric is a measurement of an attribute of a given entity. It serves to demonstrate and measure evidence of an entity's specific characteristics to improve possible problems and can be used in different dimensions, such as effort, size, and complexity [Fenton e Bieman 2020]. Table 1 shows each of the metrics that are used to estimate the score of the game.

**Table 1. METRICS FOR REUSABLE COMPONENTS**

| Id | Name | Definition | Formula |
|---|---|---|---|
| M1 | Test cases | The number of test cases that the component performed correctly. Divide the number of correct test cases (NC) by the total number of test case (N) [Redolfi et al. 2004]. | X = NC / N |
| M2 | Amount of effort | The amount of effort to generate a component can be derived from the count of distinct operators (n1) and operands (n2) and the total frequency of operators (N1) and operands (N2) [Caldiera e Basili 1991]. | N = (N1 + N2)<br>n = (n1 + n2)<br>Z = N * log2 n<br>Y = (n1/2) * (N2/n2)<br>X = Z * Y |
| M3 | Cyclomatic Complexity | Cyclomatic Complexity, also known as McCabe's Metric, basically consists of counting the flow tests (TF) of a method (if, for, while, case, catch) [Thathsarani 2024]. | X = $\sum$ TF |
| M4 | Reuse Incentive | If the user uses at least one component previously created at a later stage, +1 point will be added to the total score. Encouraging the user to reuse components. | X = $\sum$ Comp |

Another essential feature of the game is the feedback controls. There are two main feedback controls, the first serves to demonstrate whether the player succeeded in his/her goal, obtaining a high score as reward, and the second functions as a quality assistant in real time, validating the code created by the player with each interaction of the player. In an equivalent manner, with each new block of code created by the player, a pre-evaluation of the code is performed, validating its syntax and recommending a new action that the player can do. For example, if the player inserts a while block, the game will evaluate the command and recommend that he/she inserts a stop condition in that while. When that condition is fulfilled, the game will recommend that the while end with a key, giving tips on the next step to be taken by the student.

Reuse Blocks is divided into three parts; the first is the IDE (Integrated Development Environment) of blocks, which is the part where the user must create his/her components through the drag and drop of primitive components (if, else, while, etc) or components created in other phases (reusable components). The second part is the visual IDE, where codes are generated by decoding the components created through the blocks (giving feedback on what is being built in a coded way). Finally, the last part is the result area that provides feedback to the user, showing error messages and punctuation. Figure 1 shows Reuse Blocks and Table 2 presents a description based on the OCR [Guardiola 2016] and MDA [Hunicke et al. 2004] models used for defining and organizing game ideas. The game demonstrated was created for teaching components and was also created based on components, as also represented on the diagram in Figure 1. The suggested game could look like an educational activity, as stated in the theoretical foundation, in which the user must create the elements of each phase using the visual IDE. However, to provide Reuse Blocks game characteristics, features like scoring, time constraints, conditioned phases that are unlocked based on the player's score, and continuous feedback in the case of errors were incorporated.

Finally, Figures 1 and 2 show two solutions to the Fibonacci problem that were created using the for and while operators. These figures demonstrate that when using the for operator, the code becomes less efficient in terms of the effort (M3) and Cyclomatic complexity (M4) metrics. This highlights the importance of using metrics to find a more efficient algorithmic solution.

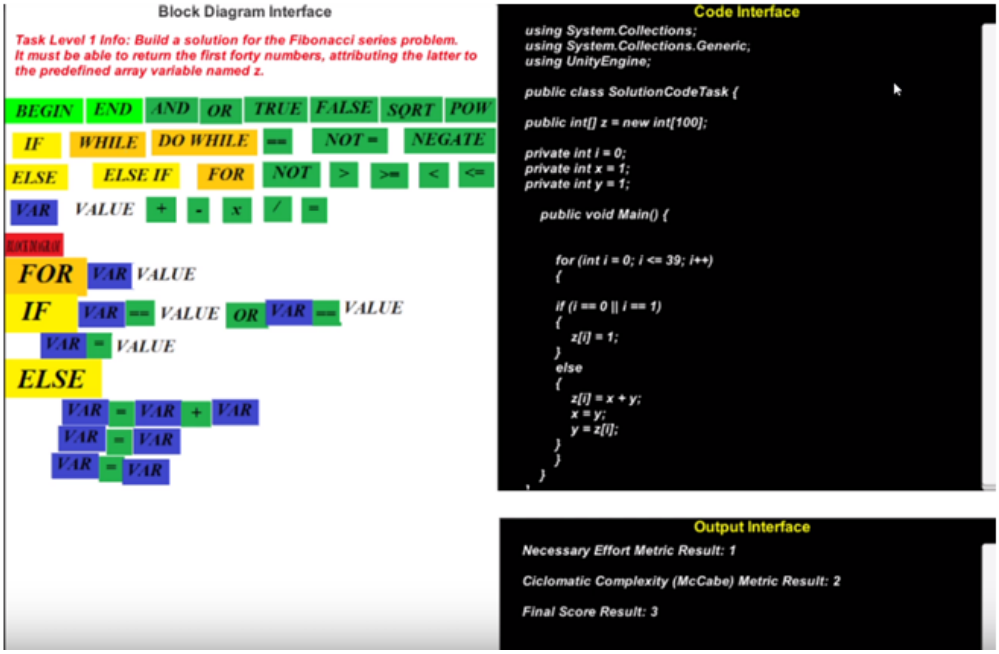


**Figure 1. Reuse Blocks, Fibonacci algorithm implemented with FOR.**


**Figure 2. Reuse Blocks, Fibonacci algorithm implemented with WHILE.**

Reuse Blocks was built with four phases, each of them will be described in more detail in the following. It is worth remembering that the order of the phases was created keeping in mind that the user could develop a component and reuse it later. It is worth mentioning that the phases designed for the game were inspired in classic computing problems such as ordering lists and Fibonacci series. Furthermore, the phases were thought to be issues with simple and fast solutions, where the most essential aspect to evaluate would be the tool rather than the solution.

**Table 2. CONCEPTUAL TABLE.**

| | Game |
|---|---|
| **Objective** | Teach reuse through the development of software components. |
| **Challenge** | Develop the algorithms proposed in each of the phases. |
| **Reward** | The score for each phase is based on the calculation of the metrics in Table 1 . |
| **Mechanics** | The game's basis is through the construction of components, where the player must drag and drop blocks of codes to solve the phase algorithm. It is worth remembering that these created components can be saved and reused in later stages. |
| **Dynamics** | After creating the algorithm, it will be validated through metrics, and a score will be calculated for the player, encouraging the user to create better programs to achieve higher grades. |
| **Aesthetics** | The user must develop his/her own component (expression) based on the algorithm described in the phase, so that he/she can get an adequate grade to advance to the next phase (Challenge). In addition, the game can still be seen as a hobby for the player. |

- **Fibonacci:** Generate the first 40 numbers in the Fibonacci series, a sequence of integers, starting with 0 and 1, in which each subsequent term corresponds to the sum of the previous two.
- **Ordering:** Given an X array, it must be ordered in ascending order.
- **Higher number:** Given an X array, the higher number must be returned.
- **Array transformation:** Given an X array, a new array must be returned where the N element of the resulting array is equal to the nth number of the original array plus the (array size - n) element. Example: given an X array equals to [2, 5, 4, 4, 5, 6], the algorithm should return the array [8, 10, 8].

## 4. Evaluation

Games are characterized by various elements such as goals, rules, restrictions, interaction, challenge, competition, rewards, feedback, and other features. Based on each of the characteristics that originate a game, it is possible to measure its teaching effectiveness. Throughout the literature, 18 factors that influenced teaching directly, such as satisfaction, motivation, interface, usability, and experience were found. These last two are the most discussed in the literature. There are several ways and frameworks to evaluate a game, such as questionnaires, observation, flow models, among others. According to the literature, a framework that addresses most of the factors that influence teaching is the MEEGA questionnaire [Ahmad 2018].

Searching for evidence about the usability and experience provided by Reuse Blocks, a case study was carried out to analyse the game's use for teaching SR to evaluate the experience provided concerning the gains in engagement, motivation, fun, and practice. For this evaluation, some questions from MEEGA questionnaire were used (Section 4.3), which is an evaluation model for educational games that captures information about the player's experiences and usability [Petri et al. 2016].

The entire evaluation took place remotely, with 26 participants categorized into three groups: specialists, who were defined as graduate participants with teaching experience, remote undergraduate students, and observed undergraduate students that used the think-aloud protocol [Jääskeläinen 2010].

Since the evaluation of Reuse Blocks did not occur in the classroom, the questionnaire for experts also had some additional questions to understand the main

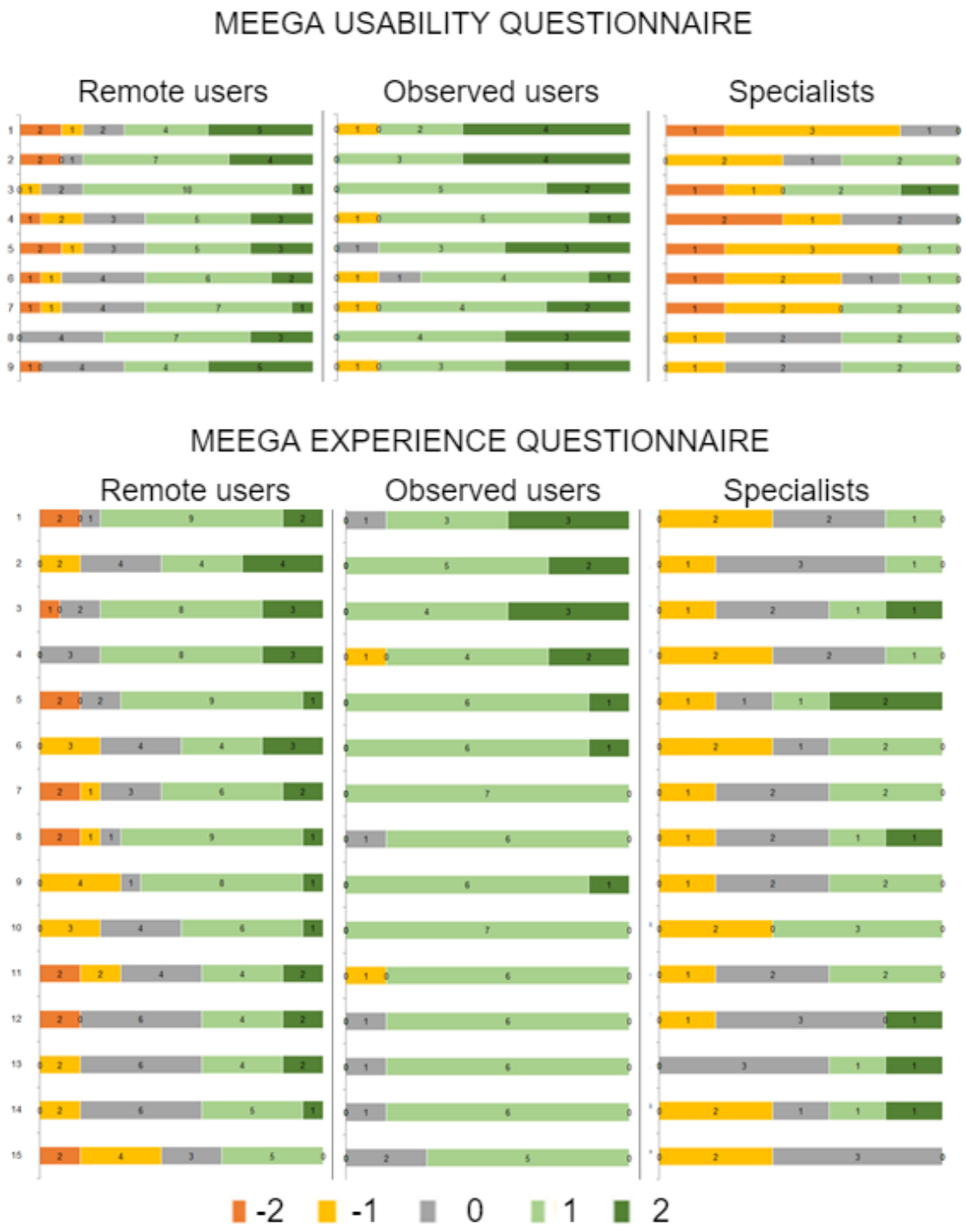


**Figure 3. Meega questionnaire applied in the evaluation of Reuse Blocks.**

problems experienced in teaching SR and try to find out if the proposed game could help to overcome these problems. The educational questionnaire can be found in (Section 4.3) (Educational Questionnaire).

From this evaluation, it was possible to observe that in terms of usability, the game received a median evaluation, with improvements to be made, mainly as regards to

the movement of the blocks. Bearing in mind each of the groups, it is possible to observe that both remote students and those who were observed had a good experience with the game in matters of usability. However, the experts' group ended up having less positive perception, which was already expected, considering that these users would have a more critical view regarding the game's use.

In terms of experience, the game was also rated as average, leaving half of the sample engaged, having strong experiences of challenge, satisfaction, and fun. The main problem pointed out was that many users considered that the existence of many commands in the game made it complex. About the evaluation groups, the group that was observed had a good experience concerning the game. However, the remote and expert groups had somewhat less positive perspectives. This possibly occurred because these groups did not receive any prior introduction to use the game, having to learn based on intuition alone. Even so, the group of remote students scored the game with a good experience provided. Figure 3 shows the information regarding each of the questions in the MEEGA questionnaire.

Despite the problems highlighted by the evaluators, most of them (19 participants) took the initiative to use games for teaching reuse and the Reuse Blocks games. The main points highlighted were to the initiative, the quality of transcription of the code in blocks to text, the use of metrics for punctuation, the visual programming, and the feedback provided by the game, mainly to the construction of code syntax.

### 4.1. Statistical tests

The MEEGA questionnaire [Petri et al. 2016] aims to evaluate a game in terms of usability and experience. However, it does not hypothesize relationships between the attributes of the population with the results obtained by the questionnaire. It is worth remembering that with the MEEGA questionnaire, a characterization questionnaire was carried out to identify information about academic specialization, age group, and experience with games and software reuse. Based on this, some statistical tests were performed to test the relationships between both questionnaires.

In the evaluation, the Likert scale was used as the standard for answering the questions. This scale is characterized by non-continuous values, which makes the application of statistical tests difficult. An additive method was used to solve this problem, which sums up all the responses of a candidate and observes the general result, transforming an ordinal scale into a continuous scale [Nwobi e Akanno 2021].

From the additive method, it is checked that the collected data did not follow a normal distribution. Therefore, we applied the non-parametric Kruskal-Wallis (KW) test to compare the participants and verify whether the characteristics of the characterization questionnaire influenced the answers obtained by the evaluation questionnaire [Nwobi e Akanno 2021]. Usually, a significance level of 0.05 is used for the KW tests, so every value less than this value leads us to consider the possibility of a factor influencing the study outcome. Table 3 shows the values of KW test.

The characterization questionnaire sought to answer four questions: age, academic background, and experience with games and reuse. Based on the information found by performing the tests, it was possible to observe that since most participants were aged between 18 and 28 years, this attribute did not influence the responses. The same occurred

**Table 3. Kruskal-Wallis test**

| Variables | H(chi-square) | Degrees of freedom | p value |
|---|---|---|---|
| Gaming experience | 0,500 | 1 | 0,480 |
| SR experience | 0,077 | 1 | 0,781 |
| Academic education | 5,460 | 3 | 0,147 |
| Age | 5,042 | 4 | 0,283 |
| Sample group | 8,682 | 2 | 0,013 |

for academic specialization and experience in games and reuse, not affecting the study's final result. Finally, the tests indicated that the group attribute (remote users, observed users and specialists) of the sample interfered with the results of the questionnaire, which was already expected.

### 4.2. Threats to validity

Validity threats are potential risks that are involved in the design and execution of studies. These threats can limit the ability to produce reliable results or generalize them to a larger population than those used in the experiments. From a critical analysis of the study, it is possible to find some threats to validity [Ihantola e Kihn 2011]. These threats were split into four types, as follows: [1] Conclusion validity: The study was carried out in three different ways (i.e. with distinct execution protocols) which may have caused a different level of reliability across studies; [2] Internal validity: The study was carried out in 3 different groups that may contain different results; [3] Construct validity: For remote users and specialists, information of study execution was sent by e-mail, which could cause a misunderstanding of some instructions in the tool or the study; and [4] External validity: only 26 participants carried out the study. There is a risk that replicating this study with a larger sample may lead to different results.

### 4.3. MEEGA Questionnaire

**Usability:** [1] The game design is attractive (interface, graphics, boards, cards, etc.). [2] The text font and colours are well blended and consistent. [3] I needed to learn a few things before I could play the game. [4] Learning to play this game was easy for me. [5] I think that most people would learn to play this game very quickly. [6] I think that the game is easy to play. [7] The game rules are clear and easy to understand. [8] The fonts (size and style) used in the game are easy to read. [9] The colours used in the game are meaningful.

**Experience:** [1] The contents and structure helped me to become confident that I would learn with this game. [2] This game is appropriately challenging for me. [3] The game provides new challenges (offers new obstacles, situations, or variations) at an appropriate pace. [4] The game does not become monotonous as it progresses. (repetitive or boring tasks). [5] Completing the game tasks gave me a satisfying feeling of accomplishment. [6] It is due to my personal effort that I managed to advance in the game. [7] I feel satisfied with the things that I learned from the game. [8] I would recommend this game to my colleagues. [9] I had fun playing the game. [10] There was something

interesting at the beginning of the game that captured my attention. [11] I was so involved in my gaming task that I lost track of time. [12] The game contents are relevant to my interests. [13] The game contents are relevant to my interests. [14] It is clear to me how the contents of the game are related to the course. [15] I prefer learning with this game instead of learning through other ways (e.g. other teaching methods). [16] The game contributed to my learning in this course. [17] The game allowed an efficient learning comparing to other activities in the course.

**Educational Questionnaire:** [1] Inside the classroom, have you ever used games for teaching? Comment on the use. [2] In your view, describe the advantages and disadvantages of using games in the classroom. [3] In your view, what is the most significant difficulty in teaching Software Reuse? [4] Do you consider that using the game in the classroom can help you to learn?

## 5. Final Remarks

From the difficulties of teaching SR (motivation/engagement and practice) and the advantages offered by the use of games in higher education, in general (greater motivation/engagement and practice), it was possible to realize that the use of games in the SR teaching could be a good strategy to be researched. However, no game was found to teach this discipline.

In view of this, a game was proposed aimed to teach SR in a practical and fun way through software components. The game was evaluated by twenty-six participants, five of whom were considered experts. This evaluation made it possible to find some problems related to the game's layout and usability. However, the game was deemed didactic and fun, presenting intense experiences of challenge and satisfaction for users.

Limitations can be identified when performing a critical analysis of this research work. Among the main limitations, the following stand out: the Reuse Blocks was not evaluated in the classroom and its evaluation had only 26 participants.

Future works were identified during the progression of this research work. The following stand out: improve the game from the problems observed in the evaluation and evaluating it in the classroom.

## References

Ahmad, M. (2018). Understanding the significance of quality criterion in educational game: A comparative review. In *ICERI2018: Proceedings of the 11th International Conference of Education, Research and Innovation*, pages 643–648. International Academy of Technology, Education and Development (IATED).

Caldiera, G. e Basili, V. R. (1991). Identifying and qualifying reusable software components. *Computer*, 24(2):61–70.

Chueca, J., Trasobares, J. I., Domingo, Á., Arcega, L., Cetina, C., e Font, J. (2023). Comparing software product lines and clone and own for game software engineering under two paradigms: Model-driven development and code-driven development. *Journal of Systems and Software*, 205:111824.

Connolly, T. M., Stansfield, M., e Hainey, T. (2007). An application of games-based learning within software engineering. *British Journal of Educational Technology*, 38(3):416–428.

Djaouti, D., Alvarez, J., e Jessel, J.-P. (2011). Classifying serious games: the g/p/s model. In *Handbook of research on improving learning and motivation through educational games: Multidisciplinary approaches*, pages 118–136. IGI Global.

Fenton, N. e Bieman, J. (2020). *Software metrics: a rigorous and practical approach*. CRC press, 3rd edition.

García-Mireles, G. A. e Morales-Trujillo, M. E. (2020). Gamification in Software Engineering: A Tertiary Study. In *Advances in Intelligent Systems and Computing*, volume 1071, pages 116–128. Springer.

Gari, M. R. N. e Radermacher, A. D. (2018). Gamification in computer science education: A systematic literature review. In *ASEE Annual Conference and Exposition, Conference Proceedings*, volume 2018-June. American Society for Engineering Education.

Guardiola, E. (2016). The gameplay loop: a player activity model for game design and analysis. In *Proceedings of the 13th International Conference on Advances in Computer Entertainment Technology*, pages 1–7.

Hunicke, R., LeBlanc, M., e Zubek, R. (2004). Mda: A formal approach to game design and game research. In *Proceedings of the AAAI Workshop on Challenges in Game AI*, volume 4.

Ihantola, E.-M. e Kihn, L.-A. (2011). Threats to validity and reliability in mixed methods accounting research. *Qualitative Research in Accounting & Management*.

Jordine, T., Liang, Y., e Ihler, E. (2014). A mobile-device based serious gaming approach for teaching and learning java programming. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, pages 1–5. IEEE.

Jääskeläinen, R. (2010). Think-aloud protocol. *Handbook of translation studies*, 1:371–374.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., e Eastmond, E. (2010). The Scratch Programming Language and Environment. *ACM Transactions on Computing Education (TOCE)*, 10(4):1–15.

Nascimento, A. M. d. S. e Leite, B. S. (2022). Uma revisão sistemática da literatura nos anais do sbgames (2010-2022) com jogos digitais educacionais em química. *Simpósio Brasileiro de Jogos e Entretenimento Digital (SBGames)*, pages 929–938.

Navarro, E. O., Baker, A., e Van Der Hoek, A. (2004). Teaching software engineering using simulation games. In *Proceedings of the International Western Simulation Multiconference*.

Niu, N., Reese, D., Xie, K., e Smith, C. (2011). Reuse a" software reuse" course. In *ASEE Annual Conference & Exposition*. American Society for Engineering Education (ASEE).

Nwobi, F. N. e Akanno, F. C. (2021). Power comparison of anova and kruskal–wallis tests when error assumptions are violated. *Metodoloski Zvezki*, 18(2):53–71.

Petri, G., von Wangenheim, C. G., e Borgatto, A. F. (2016). Meega+: an evolution of a model for the evaluation of educational games. Technical Report INCoD/GQS.05.2018.E, INCoD/GQS.

Redolfi, G., de Araujo Spagnoli, L., Bastos, R. M., Cristal, M., e Espindola, A. P. (2004). Especificando informações para componentes reutilizáveis. Technical report, Faculdade de Informática, PUCRS, Porto Alegre, Brazil.

Sametinger, J. (1997). *Software engineering with reusable components*. Springer Science & Business Media.

Schafer, U. (2017). Training scrum with gamification: Lessons learned after two teaching periods. In *IEEE Global Engineering Education Conference, EDUCON*, pages 754–761. IEEE Computer Society.

Thathsarani, N. (2024). A comprehensive software complexity metric based on cyclomatic complexity. In *2024 4th International Conference of Science and Information Technology in Smart Administration (ICSINTESA)*, pages 90–95. IEEE.

Xexéo, G., Carmo, A., Acioli, A., Taucei, B., DIpolitto, C., Mangeli, E., Kritz, J., Costa, L., e Monclar, R. (2013). O que são jogos. *LUDES. Rio de Janeiro*, 1:1–30.