ModelGame: A Quality Model for Gamified Software Modeling Learning

Ed Wilson Júnior*
Universidade do Vale do Rio dos Sinos
São Leopoldo, Rio Grande do Sul, Brazil
edwjr7@edu.unisinos.br

Kleinner Farias
Universidade do Vale do Rio dos Sinos
São Leopoldo, Rio Grande do Sul, Brazil
kleinnerfarias@unisinos.br

ABSTRACT

Gamification has been adopted in software development tasks in recent years. This adoption seeks, for example, to improve the engagement of developers while creating UML models or writing code. Empirical studies report that UML models suffer from incompleteness and inconsistency problems. This study conjectures that gamification mechanics can improve learner engagement while learning software modeling, mitigating such problems concerning UML models. The current literature lacks studies that explore gamification and UML model quality in the context of software modeling learning. This article, therefore, proposes *ModelGame*, which is a quality model to support software modeling learning in a gamified way. It serves as a reference framework so that instructors can obtain a parameterized way to evaluate UML models created by learners. The quality of UML models can be improved by applying gamified activities and providing guidelines aware of quality issues. A qualitative questionnaire was answered by 19 instructors who teach software modeling at higher education institutions. The results show that (1) 94.7% recognize that the proposed model can improve the quality of UML models, indicating that they would adopt the *ModelGame* in their learning practices; and (2) 47.4% do not use any gamification mechanics in their classes. The results are encouraging, showing the potential for applying and improving the teaching and learning of software modeling.

CCS CONCEPTS

• **Software and its engineering → Software design engineering**.

KEYWORDS

Model design, learning model, Gamification

ACM Reference Format:
Ed Wilson Júnior and Kleinner Farias. 2021. ModelGame: A Quality Model for Gamified Software Modeling Learning. In *15th Brazilian Symposium on Software Components, Architectures, and Reuse (SBCARS '21), September 27-October 1, 2021, Joinville, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3483899.3483910

1 INTRODUCTION

Gamification has been adopted in software development tasks in recent years. This adoption seeks, for example, to improve the engagement of developers while creating UML models or writing code. Empirical studies [7, 9, 14] report that UML models suffer from incompleteness and inconsistency problems. Lange [14] reinforces that these defects bring potential risks that can cause misinterpretation and communication failure, representing a risk to software quality. Thus, finding formats that favor student learning and consequently in generating increasingly effective UML models can become one of the main challenges faced by instructors that include UML (Unified Modeling Language) as part of software modeling content.

Some studies [3, 12, 25] sought to understand how to apply gamification in software modeling teaching using some elements such as points, emblems and levels. However, instructors and researchers still find limitations when applying, evaluating, and measuring the use of this tool in the learning of software modeling students and, consequently, in the models developed by them, since in the current literature there is no "frame of reference" that guides them. This study conjectures that gamification mechanics can improve learner engagement while learning software modeling, mitigating such problems concerning UML models. The current literature lacks studies that explore gamification and model quality in the context of software modeling learning.

This article, therefore, introduces *ModelGame*, which is a quality model to support software modeling learning in a gamified way. It serves as a reference framework so that instructors can obtain a parameterized way to evaluate UML models created by learners. The quality of UML models can be improved by applying gamified activities and providing guidelines aware of quality issues. A reference framework would help to (1) establish parameters for evaluating UML models created by learners; (2) provide guidelines to improve the quality of these artifacts; (3) to analyze which elements of gamification could be included in each of the phases of modeling using UML; (4) identify intrinsic and extrinsic aspects of students during the modeling stages, to improve the models; (5) to compare validated theories about the inclusion of gamification in software modeling teaching, taking into account the types of learning and methodologies used; and (6) contributing to the identification of gamification use objectives in modeling activities.

A qualitative questionnaire was answered by 19 instructors who teach software modeling at higher education institutions. The results show that (1) 94.7% recognize that the proposed model can improve the quality of UML models, indicating that they would adopt it in their learning practices; and (2) 47.4% do not use any gamification mechanics in their classes. These results are encouraging, showing the potential for applying and improving the teaching and learning of software modeling.

The remainder of the paper is organized as follows. Section 2 presents the main concepts discussed throughout the article. Section 3 discusses the related work, highlighting research opportunities. Section 4 introduces the proposed quality model. Section 5 presents how the quality model was evaluated. Section 6 points out some threats to validity. Finally, Section 7 presents some concluding remarks and future work.

2 BACKGROUND

This section presents the essential concepts for understanding this work, including gamification and software engineering teaching (Section 2.1), and software modeling and model quality (Section 2.2).

2.1 Gamification and Software Engineering Teaching

Gamification aims to use game elements in the context of not game [5], bringing all positive aspects they provide as a way to encourage and engage "players," thereby broadening their motivations.

Werbach [23] classifies gamification into three dimensions: Dynamics, Mechanics, and Components. *Dynamics*include all game aspects related to the emotional responses of "players" (e.g., relationship, progression, and narrative).*Mechanics* offer elements that promote the action of a game — usually elaborated via a rule-based development —, so that the player can interact with such elements, e.g., challenges, feedback, and rewards. **Components** represent the aesthetic elements of gamification, whose goal is to present visual aspects with which players can perform the interaction, for example, points, scores, and emblems (badges).

Knowing that the teaching of Software Engineering should involve students to experience the professional practices of the area so that they can understand which practices and techniques are useful in several different situations [2]. The challenges of teaching new software engineers are not limited to learning programming, but also include paying attention to detail, considering the quality of created models, established schedule and defined budgets [1]. In addition to understanding the technical challenges, these future professionals must be up to date with nontechnical issues, including teamwork, communication and management.

To meet these new demands of the current context, the format with exhibition classes is no longer considered enough and may even become demotivating and ineffective in learning students. In this sense, gamification has been increasingly used in the teaching of software engineering as a way to promote behavioral and psychological changes [11] providing an environment that favors communication, cooperation, feedback, reward, achievement and other recurring elements that are capable of improving performance, efficiency and engagement in educational activities , and can enhance, for example, the learning of software modeling.

2.2 Software Modeling and Model Quality

Software modeling encompasses the set of principles, concepts, and practices that lead to the development of a high-quality system or product. The principles of this activity establish a philosophy that guides the entire software development process.

In this scenario, UML models play a crucial role in software development tasks, for example, documenting project decisions, understanding development details, promoting better communication between teams, and generating greater efficiency in software development [19]. However, these models suffer problems of inconsistency and incompleteness [10, 18], as well as end up being overlooked within the modeling process, as pointed out in some empirical studies in the literature [14, 15]. Class and sequence diagrams, for example, present inconsistencies when sequence diagram objects are not found in the class diagram, consequently developers end up living with inconsistencies throughout the development process.

A research challenge still open is how to evaluate these diagrams, both in industry and in the teaching process, in terms of quality, such as syntactic and semantic, for example.

3 RELATED WORK

The selection of related works was carried out following two steps: (1) search in digital repositories, such as *Google Scholar* and *Scopus (Elsevier)* of articles related to gamification, quality modeling, and modeling learning; and (2) filter selected articles considering the alignment of such works with the objective of the work (Section 4). After selecting the works, they were analyzed (Section 3.1) and compared (Section 3.2), seeking to identify research opportunities.

3.1 Analysis of Related Works

Porto *et al.* (2021) [4]. This work performed a systematic mapping with the objective of characterizing how gamification has been adopted in noneducational contexts of software engineering activities. The main results of this study show that gamification provided benefits for activities such as requirements specification, development, testing, project management, and support process. In addition, he pointed out that the number of publications and new research initiatives has increased over the years, many positive results have been achieved in software engineering activities. Nevertheless, the study reinforced that gamification can still be explored for other tasks in this area, as empirical evidence is very limited.

Marin (2021) [17]. It performed the application of gamification on some topics of a software engineering course to engage students and increase their motivation and argued that, with due motivation, students can better exercise the topics and obtain more solid knowledge. There were five games related to risk management, BPMN modeling, Scrum process, design and inspection of class diagrams, and cosmic functional size measurement to assist in the learning process of the software engineering course. This study also presented the lessons learned about the application of gamification and serious games in software engineering, including limitations or disadvantages.

Jurgelaitis *et al.* (2018) [12]. This work conducted a research to investigate how gamification could be inserted into an Information Systems Modeling course, which covers a range of topics on UML. As a result, an implementation of the gamified system modeling course in the Moodle environment was presented, using additional plugins for the use of the necessary gamified elements. The study showed good results and obtained a positive acceptance by the participating students.

Rodrigues *et al.* (2018) [22]. They investigated the use of games and game elements in software engineering education, through a research that had the participation of 88 instructors of this discipline. The results showed that most instructors are aware of these educational approaches, however, the games were adopted by only 21 participants and game elements were adopted only by 19. Games are most often used to cover "Software Process" and "Project Management". The most commonly used game elements are points, quizzes, and challenges. The results also show that the main reasons for not adopting the resources are the lack of knowledge, information about games relevant to the engineering of teaching software, and the lack of time to plan and include these approaches in the classroom.

Cosentino *et al.* (2017) [3]. They present a model-based approach to learning modeling in a gamified way. The approach includes a new language to model the gamification process itself and an environment where it can be incorporated into current modeling tools to allow instructors and students to design and use a complete modeling framework, including gamification elements. In addition, the approach also had as a proposal to provide support to collect and analyze gamification data, thus facilitating monitoring activities.

Yohannis (2016) [25]. This research presents an exploration of game design as an approach to strengthening the student's mastery in software modeling by developing their abstraction skills. It brought together concepts of gamification development, such as the lens of atoms of intrinsic skill and principles of pedagogical design of various theories and models of learning. The research follows the Design Science Research Methodology and explores the best practices of Model Oriented Engineering. As a result, a modeling game design framework and generation structure and a series of produced games are presented.

Pedreira *et al.* (2015) [21]. They developed a systematic mapping of gamification in Software Engineering based on 29 studies. The mapping revealed that software implementation is the area in which most studies focus, followed by software requirements, few others in different areas, such as project planning and software testing, and even to a lesser extent in activities involving software modeling. However, the highlight of this work was to highlight that gamification in software engineering is still at a very early stage and the evidence on its impact in this field remains inconclusive.

3.2 Comparative Analysis and Opportunities

Five Comparison Criteria (CC) were defined selecting the most relevant variables to assist in the process of identifying similarities and differences between the proposed work and the selected articles. This comparison is crucial to make the process of identifying research opportunities using objective rather than subjective criteria. The criteria are described below:

- **Context (CC01):** Works that explore the use of gamification in software modeling teaching/learning.
- **Participant profile (CC02):** Studies that collected data from participants for screening and profile characterization.
- **Applicability of Gamification in UML (CC03):** Studies that evaluated how gamification can contribute to UML models.
- **Model creation (CC04):** Studies that have developed a model to improve factors that imply the non-adoption of UML.
- **Instructor participation (CC05):** Studies that collected qualitative data through the participation of software modeling instructors.

Table 1 shows the comparison of the selected works, confronting this work. Some gaps and research opportunities are observed: (1) only the proposed work was the only one to fully meet all comparison criteria; (2) although most of them targeted the application of gamification in software modeling teaching, they were not directed to the use of UML; (3) no study has developed a model to evaluate the learning and improvement of UML models developed by students; and (4) most of them did not have the participation of instructors to identify the difficulties and opportunities in the application of gamification in the teaching of software modeling. Thus, the next Section presents a quality model to explore these identified opportunities.

Related Work	Comparison Criterion				
	CC1	CC2	CC3	CC4	CC5
Proposed Work	●	●	●	●	●
Porto et al (2021) [4]	○	○	●	○	○
Marin (2021) [17]	●	○	◐	◐	○
Jurgelaitis et al (2018) [12]	●	○	●	●	○
Rodrigues et al (2018) [22]	●	●	◐	○	●
Cosentino et al (2017) [3]	●	○	●	●	○
Yohannis (2016) [25]	●	○	◐	◐	○
Pedreira et al (2015) [21]	○	○	◐	○	○

● Completely Meets ◐ Partially Meets ○ Does not attend

Table 1: Comparative analysis of the selected related works

4 PROPOSED QUALITY MODEL

This section presents the proposed quality model to support software modeling learning in a gamified way. It serves as a frame of reference so that instructors can evaluate the UML models created by students through gamified activities. Section 4.1 presents a proposal of a generic analytical framework. Section 4.2 details the abstract syntax of the proposed quality model. Section 4.3 explains the quality notions related to the gamified software modeling learning.

4.1 Generic Analytical Framework

Figure 1 presents the generic analytical framework for improving the quality of the models and serves as the basis for the creation of an evaluation scheme. The arrows ("links"), labeled as Evaluation and Gamified Modeling, represent the questions that the evidence must answer; dotted lines represent associations; rectangles represent the Models (rounded corners) or the quality states (square corners) by which these bindings are measured. Ellipses represent the adverse effects that can be generated from the evaluation and use of gamification.

The numbers refer to the key questions and are connected with the concepts and relationships of the abstract syntax of the Quality

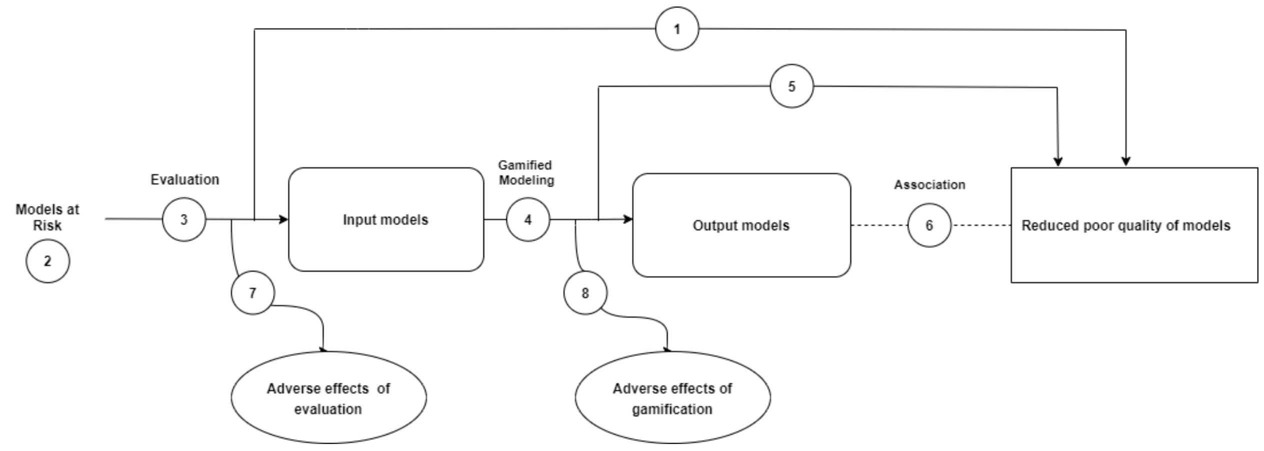
1
5
Models at Risk
2
Evaluation
3
Input models
Gamified Modeling
4
Output models
Association
6
Reduced poor quality of models
7
8
Adverse effects of evaluation
Adverse effects of gamification

Figure 1: Generic analytical framework for gamified software modeling learning.

Model (presented in Section 4.2), as follows: (1) Are there tools that assist instructors in evaluating the models developed by students, thus reducing the poor quality and incompleteness of these artifacts? (2) What is the prevalence of characteristics that cause models to be at risk? (3) Are there notions of quality to evaluate the models as a way to define parameters when performing their correction? (4) Applying the use of gamification in models that need intervention would be a way to identify factors that could generate models with high quality levels? (5) Does the application of gamification improve the quality of the model? (5.a) How are the models without gamification evaluated in relation to those with gamification? (5.b) Are there reasons to expect that gamification models can have better quality results than those that are generated without gamification? (6) Is the output model really effective when associated with reducing the poor quality of the model? (7) Does the absence of evaluation result in adverse effects? (7.a) is the evaluation acceptable for the model? (7.b) What are the potential harms, and how often do they occur? (8) Does gamification result in adverse effects on models?

Fact is that it is not enough just to include this "toolbox" in the UML learning process, it is necessary to provide the instructor with a model (guide) that can serve as a reference to evaluate the quality of diagrams elaborated through gamified activities. For example, the instructor could create models predefining inconsistencies by making use of these questions raised to evaluate the models created by the students. The set of questions serves as the starting point for this evaluation. Knowing that the adaptation of the gamification approach requires a significant effort [20], in this study we present The ModelGame as a way to identify factors that contribute to the quality of these artifacts and, consequently, to the students' learning.

4.2 Abstract Syntax

Following the specification pattern of the UML metamodel, Figure 2 presents the abstract syntax of the proposed Quality Model for gamified software modeling learning (ModelGame). It identifies the main concepts and relationships. The numbers represent the notions of quality that are discussed in Section 4.3. The following are detailed each of these concepts and relationships.

Domain. The first concept presented in this study is the domain, which corresponds to a specific context of the application to be developed to solve the problem. In this process, the design template represents the solution given to the domain.

Association

- `contextualizes`: Challenges[*]

Each contextualise refers to the domain that will serve as the basis for the challenges launched.

Challenges. This concept represents the phase in which the problem is contextualized (domain-based), as well as what will be the missions, phases, scenarios, and other elements presented to the players, in this case the students, who must use the principles of software engineering to perform the modeling and reach the final goal.

Association

- `influences`: Design Model[*]

Each influence represents that the proposed challenge interfered in aspects of the design model, causing the user to seek to make a continuous improvement.

Modeling Language. Software modeling is an important step for development to happen in a way that adheres to the requirements established by the requester, for this, there is the modeling language, which offers a standardized way to document and design software. Through the use of modeling languages, it is possible to achieve a high level of understanding about the software in question, improving the communication between all those involved in the process, thus avoiding implementation errors. It points out that software engineers use these languages to communicate design decisions and verify the feasibility of implementing the intended design. The UML was consolidated as the Modeling Language in the paradigm of object orientation, in which it is possible through visual notation generated from the diagrams- presented later in this study as Design Models- to perform the representation of various perspectives of the system.

Association

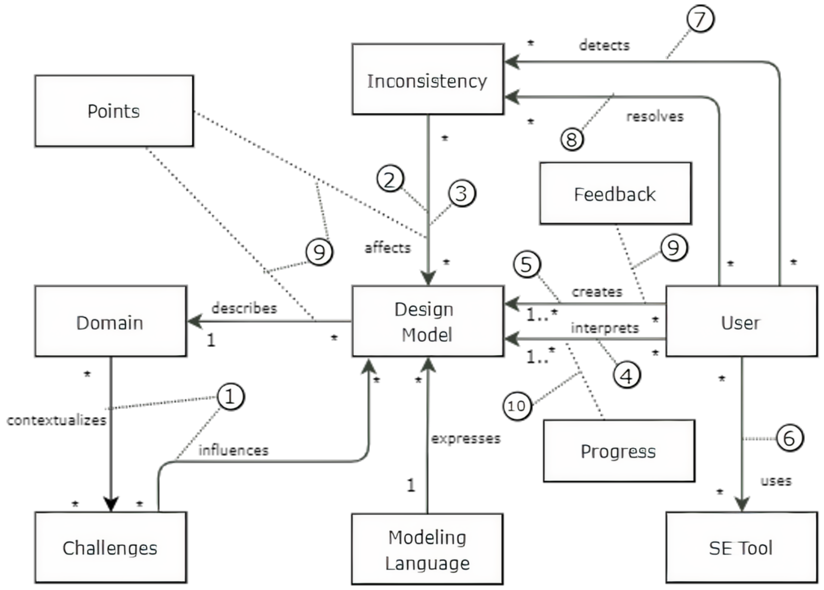

Figure 2: Abstract Quality Model Syntax.

- `expresses`: Design Model[*]

Performs the representation of the intended design templates, in which the Modeling Language should be applicable to the domain type.

User. This concept corresponds to the individual who performs the interpretation of the developed design models, whose objective is to be able to understand the domain in question. In the gamified context, the user has the role of player and it is he who performs the whole process, being able to perform the interpretation of existing models or even creating new ones. The user can also identify and resolve inconsistencies that arise from compositions between models.

Association

- `creates`: Design Model[1..*]

Represents the process in which the user creates a design template, which can be one or more.

- `interprets`: Design Model[1..*]

In this association, the user performs the interpretation of the design template. When interpreting the model, paths for the resolution of inconsistencies can be identified.

- `detects`: Inconsistency [*]

Represents the user's discovery of design model inconsistencies, for example, those that are generated from identifying conflicts, whether a class is abstract or not.

- `resolves`: Inconsistency [*]

Each resolves equates to the resolution representation of the inconsistencies by the user that happens after he analyzes and determines the best alternative to perform this action.

- `uses`: Modeling Tools [*]

Determines that the user can use modeling tools to generate/update design models.

Association

- `Without a directed relationship`.

Modeling Tool. This concept represents the applications that are used to carry out the construction of design models. There are several tools available, online and desktop, and it is up to the user to choose the one that will best meet their needs and adapt to the context in question, that is, they work in any domain that is being considered.

Design Model. The design model refers to a visual notation (diagram) to represent static and dynamic aspects. These models are built according to a specific objective or task and tend to facilitate the logical interpretation of the software in several aspects. The most popular diagrams are Use Cases and Classes, the first being static and representing a set of actions generated from functional requirements (use cases) and presenting the interactions generated with external users (actors). The second is a static diagram and makes the representation of the logical structure of the software involving the classes, their attributes, methods, and relationships between them [19].

Association

- `describes`: Domain[1]

Each describes makes the representation of a specific domain and means that every design model must describe it.

Inconsistency. It corresponds to the defects found in the models developed by users. They may occur because of the nonidentification and correction of possible conflicts and even an erroneous interpretation.

Association

- `affects`: Design Model[*]

This association indicates that with each occurrence of the affect, a problem is presented harming the quality of the design model.

Points. This concept represents one of the most used game mechanics in software engineering and functions as a quantitative reward for each action developed, in which it is possible to regulate the number of rewarded points of the player, defined here as user, based on the importance of each action. Through this concept, it is possible to stimulate competition, collaboration, and creativity among users, stimulating learning. Points appear as a derivation of the association affects, since when each inconsistency error is identified or not, the user will receive a score and the association describes, because the points will also be applied when making connections between the model and the domain.

Progress. The concept of progress emerges as a factor that makes the user able to perceive its evolution in the process, in this case, software modeling. Progress emerges as a derivation of the association interprets, making the user know when they have performed a correct interpretation of the proposed design model or what still needs to be improved.

Feedback. Feedback has the role of making the user realize that the proposed goal can be achieved and follow its evolution, including analyzing how to change or creating new strategies to achieve the goal. This concept emerges as a derivation between the associations it creates, causing the user to receive a return to the model creation process.

4.3 Quality Notions

As discussed in Section 2, gamification can bring important elements for learning software modeling and, therefore, the objective of this section is to produce the notions of quality of the model of this study. The ModelGame is composed of ten counts, four of which are proposed in this study - scope, use, motivational and engagement - extracted from the main benefits that the gamification elements presented in Figure 2 can bring to the models. The others are adaptations of previous works [6, 14, 15], they are, syntactic, semantic, social, effort, detection and resolution.

Scope Quality (1). It seeks to determine how much the proposed challenge is contextualized with the design model, as well as the definition of the domain, problem, competencies, concepts, behaviors and attitudes that will be developed throughout the process.

Syntactic Quality (2). This notion makes the representation of the process of correction of the design models that are produced by the modeling language, because if it is not used correctly, inconsistencies will arise. It is important to insert this notion of quality into our study, since during the process of developing the models, users may come across the composition of two class diagrams, for example.

Semantic Quality (3). It is necessary to verify that the design model and the problem domain match, so this notion performs this type of analysis. Communication problems may occur between users if the semantic elements of the model are affected.

Social Quality (4). Design models are used to communicate between members of a team to inform all established decisions about software development [8]. If divergent interpretations occur, this communication will be greatly impaired.

Quality of Effort (5). This notion refers to the production challenges of the model that will be generated, including factors such as time and cost.

Quality of Use (6). To produce design templates, users can use unusual tools such as paper, whiteboard, and more. However, most of the time they choose to use formal tools (CASES) and can be online or desktop. This notion corresponds to the level of ease and applicability of the models elaborated when making use of these tools, it is also important to contribute to communication between users through collaboration-related functionalities.

Detection Quality (7). This notion is referenced to the process of locating inconsistencies, since when users arise, they should perform traceability of them quickly. If the detection is complicated, it could hinder the process of correcting the models.

Resolution Quality (8). It corresponds to the level of quality related to the effort that users take to look for alternatives to solve the identified problem.

Motivational Quality (9). This notion refers to the motivational factors involved during the learning and development of design models, which can be intrinsic and extrinsic. Elements of gamification such as points, feedback and progress bring the user a degree of satisfaction in continuing their discovery and transformations throughout the process.

Quality of Engagement (10). The user in tracking their progress can feel committed to the objective in question, and this notion represents the measurement of the level of commitment of them during the development of design models.

5 EVALUATION

This section describes the methodology followed to evaluate the proposed quality model. This methodology follows well-established empirical guidelines [24]. Section 5.1 details the objective and research questions (RQ). Section 5.2 presents the questionnaire formulated to evaluate the proposed quality model. Section 5.3 explains the context and selection of participants. Section 5.4 describes the presentation of the Model. Section 5.5 presents the analysis of the collected data.

5.1 Objective and Research Questions

The objective **(O)** of this study is twofold: **(O1)** Introduce ModelGame as a tool for teaching Software Modeling; and **(O2)** Analyze the applicability of the quality model regarding the improvement of UML models.

To analyze the different facets of the objectives, two Research Questions **(RQ)** have been formulated:

- **RQ1:** How do instructors evaluate the use of gamification in software modeling?
- **RQ2:** What is the acceptance of ModelGame by software modeling instructors?

5.2 Questionnaire

Data was collected through an online questionnaire created through Google Forms[1] following well-established guidelines described in [24]. This strategy was chosen because the questionnaire could be applied quickly and easily collect data from individuals in geographically diverse locations. The questions of the questionnaire

[1]Questionnaire: https://forms.gle/qjaFDpErEtGdLuWw6

were concerned with examining the research gaps of previous studies and apprehending the structures of the previously developed questionnaire.

Part 1: Participant profile. The first part of the questionnaire consisted of collecting data that are related to the characteristics and opinions of the participants. The creation of the participant profile through this data is important to make the selection of possible users of ModelGame. Without this profile, participants with an inadequate profile may generate inconsistent assessments. Participants were asked to provide more general information, such as age, education level, academic background. Information about the time of experience in teaching was also considered, including teaching software modeling and level of knowledge about UML models.

Part 2: TAM questionnaire. The second part addressed questions about the usability and acceptance of the technique, aiming to explore q3. To this end, this part of our questionnaire is based on the technology acceptance model (TAM) [16]. This part contained nine questions, which were answered through the Likert Scale, including Totally Agree, Partially Agree, Neutral, Partially Disagree, and Totally Disagree. The questions formulated (Q) dealt with several topics, including perceived ease of use (Q1-3), perceived utility (Q4-7), attitude towards use (Q8), and behavioral intention to use (Q9).

5.3 Selection of participants

The participants were selected based on the following criteria: instructors and/or professionals working in the teaching of software modeling in higher education institutions in Brazil. Using this criterion, we sought to select participants with academic training and practical experience in teaching. This finite set of all possible participants represents the target population [13]. This population represents those people who are in a position to answer the questions formulated and to whom the results of the survey apply [13]. In all, 19 people (n) answered the questionnaire. The participants were invited via e-mail to participate in the study and each of them previously received the explanation/training about the model proposed through the researcher and there was no doubt, they could leave for the next step that consisted of completing the TAM questionnaire. We discussed the experimental process in the next section.

5.4 Experimental Process

Figure 3 presents the experimental process used in this study, which is composed of three phases discussed below:

Phase 1: Presentation. It has an activity, *presentation*, in which the researcher explained to the participants through a video detail about the quality model. This process took place individually and in a standard way, where space was also made available for participants to answer possible doubts about the proposed study and model, lasting an average of 20 minutes.

Phase 2: Application of the TAM questionnaire. It has two activities, the first being *Collect demographic data*. The participants answered a list of questions (input) so that we could collect their characteristics and opinions about the ModelGame. The demographic data collected (output) became the result of this activity. The second activity *Apply TAM questionnaire (input)*. Participants received a list of questions about the perception of ease of use, perceived utility, attitudes, and intention of behavior, in relation to the ModelGame. Qualitative data (output) were generated, regarding the usability and acceptance of the Model under the perspective of professionals who teach software modeling. This questionnaire followed the guidelines of the TAM [16].

Phase 3: Analysis and result report. It has two activities. The first, *Analyze data* sought to perform a thorough analysis of the data collected through the questionnaire and the researcher's perception regarding the participants' doubts during the presentation stage. For this, the collected data were analyzed separately, as well as confronted, aiming to perform a triangulation of them. Subsequently, there was an *Evaluation data*, as a way to understand in a more depth the context, the perceptions of the participants in relation to the proposed model as well as its applicability.

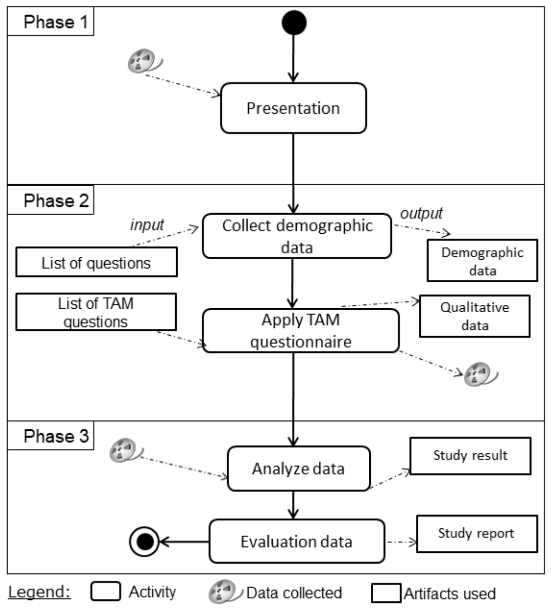

Figure 3: The experimental process.

5.5 Result Analysis

5.5.1 Profile data of the participants. Table 3 describes the profile data, reporting the characteristics and opinions of the participants. These data were collected from May 18 to June 5, 2021. In total, we had 19 participants. Our participants are between 20 and 49 years old, most of them have a degree in Computer Science (52.6%), Information Systems (26.3%) or Systems Analysis (21.1%) and are specialists (36.8%), masters (36.8%) and doctors (15.8%). About the working time in teaching, the majority (42.1%) they have been teaching for more than 8 years and teach disciplines related to software modeling, including software engineering, systems analysis and software projects. A total of 47.4% have a full level of knowledge about UML and almost half of them (47.4%) has not yet used gamification in the teaching of software modeling. Therefore, we consider

	Totally agree	Partially agree	Neutral	Partially disagree	Totally disagree
Perceived ease of use					
I found the quality model easy to use	8	9	2	0	0
I found the quality model easy to learn	10	9	0	0	0
I found the quality model easy to master	6	12	0	1	0
Perceived usefulness					
The model would make it easier to understand which elements of gamification can be used in modeling .	12	5	2	0	0
Using the quality model would help increase productivity.	9	8	2	0	0
The model would provide an understanding of how to mitigate the incompleteness of UML diagrams.	5	8	5	1	0
The model would help compare theories about gamification in software modeling teaching.	13	4	2		
Attitude towards use					
Using the Quality Model for Gamified Software Modeling Learning is a good idea.	13	5	1	0	0
Behavioral intention to use					
I would use the quality model in software modeling classes.	10	7	2	0	0

Table 2: Collected data related to TAM questionnaire.

that although small, our sample is adequate to carry out an initial evaluation of the proposed approach.

Characteristic and Opinion (n=19)	Answer	#	%
Age	< 20 years	0	0.0%
	20-29 years	4	21.1%
	30-39 years	8	42.1%
	40-49 years	5	26.3%
	> 49 years	2	10.5%
Education	Undergraduate*	0	0.0%
	Specialization*	7	36.8%
	Master*	7	36.8%
	PhD*	3	15.8%
	Others	2	10.6%
Undergraduate course	Information Systems	5	26.3%
	Computer Science	10	52.6%
	Computer Engineering	0	0.0%
	System Analysis	4	21,1%
	Others	0	0.0%
Time of experience in teaching	< 2 years	4	21.1%
	2-4 years	2	10.5%
	5-6 years	3	15.8%
	7-8 years	2	10.5%
	> 8 years	8	42.1%
Experience in teaching software modeling	< 2 years	3	15.8%
	2-4 years	5	26.3%
	5-6 years	3	15.8%
	7-8 years	2	10.5%
	> 8 years	6	31,6%
Level of knowledge about UML models	Beginner	2	10.5%
	Junior	5	26.3%
	Full	9	47.4%
	Senior	3	15.8%
Used gamification in teaching	Yes	9	47.4%
	No	9	47,4%
	Maybe	1	5.3%
Gamification can contribute to the quality of the models of UML diagrams generated by students	Totally agree	10	52.6%
	Partially agree	8	42.1%
	Neutral	1	5.3%
	Partially disagree	0	0.0%
	Totally disagree	0	0.0%

Table 3: The profile data of the participants.

5.5.2 RQ1: How do instructors evaluate the use of gamification in software modeling? Table 3 presents the collected data related to the RQ formulated. First, we begin the analysis by verifying how instructors visualize gamification in software modeling teaching. Although most of them (47.4%) have not yet used gamification elements (scores, challenge, emblem, among others) in their classes, most (52.6%) totally agree and (42.1%) partially agree that the use of these can contribute to the quality of the models developed by the students.

We consider the percentage of instructors who have not yet used gamification in their classes to be high and this may be tied to factors such as lack of knowledge, information about the tool, and even time to plan and include these approaches [22]. Although they were based on software modeling teaching context, previous studies [3, 4, 12, 17, 25] they did not count on the participation of instructors and we understand that this participation is fundamental to understand the perceptions of these professionals since they will be at the forefront of the use of gamification.

The ModelGame proposed in this study could help them insert gamification into their classes, according to the software modeling learning design [25], based on the assumption that for this, it is necessary to develop a better understanding of the tasks, activities, skills and operations that the different elements of gamification can offer and how they can correspond to the desired learning outcomes by developing a more concrete and motivating presentation that can involve students and facilitate deep learning with UML.

Summary of RQ1: *The results show that (47.4%) of the participating teachers have not yet made the use of gamification elements, however, the majority (94.7%) agrees that the use of these elements can contribute to the quality of the models developed by the students. These results indicate that ModelGame could help teachers to insert gamification into their classes.*

5.5.3 RQ2: What is the acceptance of the ModelGame by software modeling instructors? Using the TAM questionnaire, we tried to evaluate the ease of use, perceived usefulness, attitude, and behavioral intention to use the Quality Model. Table 2 shows the data obtained. Our data obtained show that no one disagreed that the ModelGame is easy to use, learn, and master. On the contrary, almost 90% of participants find the model easy to use (42.1% totally agree and 47.4% partially agrees and 10.5% neutral), learn (52.6%

fully agree and 47.4% partially agree) and master (31.6% fully agree, 63.2% partially agree and 5.3% partially disagree).

The results are also favorable considering the perception of utility. Most participants realized that the ModelGame would make it easier to understand which elements of gamification can be used in each of the phases of modeling using UML(63.3% totally agree, 26.3% partially agree and 10.5% neutral), increase productivity (47.4% fully agree, 42.1% partially agree and 10.5% neutral), and the use of the quality model would provide an understanding of how to mitigate the incompleteness of UML diagrams (26.3% agree totalmen 42.1% partially agree, 26.3% neutral and 5.3% partially disagree). Still in the useful aspect, we tried to know if the quality model would help to compare validated theories about the inclusion of gamification in software modeling teaching (68.4% totally agree, 21.1% partially agree and 10.5% neutral).

Considering the attitude towards use, participants believe that using the ModelGame is a good idea (68.4% totally agree, 26.3% partially agree and 5.3% neutral), just as they are confident and would use the Model in software modeling classes (52.6% totally agree, 36.8% partially agree and 10.5% neutral). These findings show the potential for acceptance by people with profiles similar to those of participants. The results are encouraging and show the potential to use the proposed approach in the educational scenario.

Summary of RQ2: *The results are positive regarding the ease of use, perceived usefulness, attitude and behavioral intention of using ModelGame by participating teachers and show its potential for use in the educational scenario.*

6 THREATS TO VALIDITY

This section discusses the possible threats to the validity of the study.

Internal validity. The main point affecting the internal validity of our study concerns the total time used for the exploratory phase. To mitigate this threat, we performed the video recording of a pilot explaining the operating details and objectives of the ModelGame. In relation to the methods used, the threats related to internal validity relate to how we extract the perceptions of the discussions and whether they represent the perceptions of teachers about the use of the Model. We try to reduce this threat by applying the TAM questionnaire.

External validity. We identified threats related to external validity, such as the number of participants who never applied the use of gamification. This study was limited to 19 participants (teachers) from various educational institutions, of which 9 (47.4%) never used any element of gamification in their classes, this factor can interfere in the data, since the model intends to evaluate the quality of UML diagrams from gamified activities.

Conclusion validity. Threats related to the validity of the conclusion are related to treatment and outcome. We try to make the reduction by combining quantitative and qualitative data through different resources. These data were obtained through audio and questionnaires. We analyze this data to answer the research questions.

7 CONCLUSIONS AND FUTURE WORK

This study proposed an initial quality model (ModelGame) that serves as a reference framework for instructors for qualitative evaluations of UML models developed from gamified activities, the application of an empirical study with 19 participants was carried out to understand their vision in relation to gamification and the acceptance of the proposed Model. It was identified that most have not yet used gamification in their classes, but agree that their use can contribute to the quality of the models developed by the students and were open to using the model. Our findings can enhance the adoption of new teaching practices through gamification, resulting in the improvement of software modeling learning using UML, and consequently the creation of models developed by students. These approaches can stimulate students' immersion in the design of systems as future professionals during learning.

Finally, we hope to carry out in the future a series of experimental studies to analyze each stage of application of the ModelGame and that this work represents a first step to better support the application of empirical studies on models of evaluation of the use of gamification in software modeling. We also hope that the questions described throughout the article will encourage other researchers to extend our study to different modeling languages and teaching methodologies.

REFERENCES

[1] Rick Adcock, Edward Alef, Bruce Amato, Mark Ardis, Larry Bernstein, Barry Boehm, Pierre Bourque, John Brackett, Murray Cantor, Lillian Cassel, et al. 2009. *Curriculum guidelines for graduate degree programs in software engineering.* ACM.

[2] Mark Ardis, David Budgen, Gregory W Hislop, Jeff Offutt, Mark Sebern, and Willem Visser. 2015. SE 2014: Curriculum guidelines for undergraduate degree programs in software engineering. *Computer* 48, 11 (2015), 106–109.

[3] Valerio Cosentino, Sébastien Gérard, and Jordi Cabot Sagrera. 2017. A model-based approach to gamify the learning of modeling. CEUR Workshop Proceedings.

[4] Daniel de Paula Porto, Gabriela Martins de Jesus, Fabiano Cutigi Ferrari, and Sandra Camargo Pinto Ferraz Fabbri. 2021. Initiatives and challenges of using gamification in software engineering: A Systematic Mapping. *Journal of Systems and Software* 173 (2021), 110870.

[5] Sebastian Deterding, Miguel Sicart, Lennart Nacke, Kenton O'Hara, and Dan Dixon. 2011. Gamification. using game-design elements in non-gaming contexts. In *CHI'11 extended abstracts on human factors in computing systems.* 2425–2428.

[6] Ana Fernández-Saez et al. 2012. A systematic literature review on the quality of UML models. *J. Data. Manage* 22, 3 (2012), 46–70.

[7] Kleinner Farias et al. 2012. Evaluating the impact of aspects on inconsistency detection effort: a controlled experiment. In *International Conference on Model Driven Engineering Languages and Systems.* Springer, 219–234.

[8] Kleinner Frias et al. 2014. Towards a quality model for model composition effort. In *29th Annual ACM Symposium on Applied Computing.* 1181–1183.

[9] Kleinner Farias et al. 2015. Evaluating the effort of composing design models: a controlled experiment. *Software & Systems Modeling* 14, 4 (2015), 1349–1365.

[10] Kleinner Farias et al. 2019. UML2Merge: a UML extension for model merging. *IET Software* 13, 6 (2019), 575–586.

[11] Juho Hamari, Jonna Koivisto, and Harri Sarsa. 2014. Does gamification work?– a literature review of empirical studies on gamification. In *2014 47th Hawaii international conference on system sciences.* Ieee, 3025–3034.

[12] Mantas Jurgelaitis, Vaidotas Drungilas, and Lina Čeponienė. 2018. Gamified Moodle course for teaching UML. *Baltic journal of modern computing* 6, 2 (2018), 119–127.

[13] Barbara A Kitchenham and Shari L Pfleeger. 2008. Personal opinion surveys. In *Guide to advanced empirical software engineering.* Springer, 63–92.

[14] Christian Franz Josef Lange. 2007. Assessing and Improving the Quality of Modeling: A series of Empirical Studies about the UML. (2007).

[15] Odd Ivar Lindland, Guttorm Sindre, and Arne Solvberg. 1994. Understanding quality in conceptual modeling. *IEEE software* 11, 2 (1994), 42–49.

[16] Nikola Marangunić and Andrina Granić. 2015. Technology acceptance model: a literature review from 1986 to 2013. *Universal access in the information society* 14, 1 (2015), 81–95.

[17] Beatriz Marín. 2021. Lessons Learned About Gamification in Software Engineering Education. In *Latin American Women and Research Contributions to the IT Field*. IGI Global, 174–197.

[18] Kleinner Oliveira, Alessandro Garcia, and Jon Whittle. 2008. On the quantitative assessment of class model compositions: An exploratory study. *1th ESMDE at MODELS* (2008).

[19] OMG. 2017. UML: Infrastructure specification. https://www.omg.org/spec/UML/2.5.1/PDF.

[20] Sofia Ouhbi and Nuno Pombo. 2020. Software Engineering Education: Challenges and Perspectives. In *IEEE Global Engineering Education Conference*. 202–209.

[21] Oscar Pedreira, Félix García, Nieves Brisaboa, and Mario Piattini. 2015. Gamification in software engineering–A systematic mapping. *Information and software technology* 57 (2015), 157–168.

[22] Pedro Rodrigues, Mauricio Souza, and Eduardo Figueiredo. 2018. Games and gamification in software engineering education: A survey with educators. In *2018 IEEE Frontiers in Education Conference (FIE)*. IEEE, 1–9.

[23] Kevin Werbach and Dan Hunter. 2012. *For the win: How game thinking can revolutionize your business*. Wharton digital press.

[24] Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. 2012. *Experimentation in software engineering*. Springer Science & Business Media.

[25] Alfa Yohannis. 2016. Gamification of Software Modelling Learning.. In *DS@ MoDELS*.