

Towards an Open Repository for Teaching Software Modeling applying Active Learning Strategies

Williamson Silva^{1,2}, Bruno Gadelha¹

¹ Instituto de Computação (IComp)
Universidade Federal do Amazonas
Manaus, AM - Brazil
williamson, bruno
{@icompu.ufam.edu.br}

Igor Steinmacher²

² School of Informatics, Computing and
Cyber Systems
Northern Arizona University
Flagstaff, AZ - USA
igor.steinmacher@nau.edu

Tayana Conte¹

¹ Instituto de Computação (IComp)
Universidade Federal do Amazonas
Manaus, AM - Brazil
tayana@icompu.ufam.edu.br

ABSTRACT

Modeling is a core topic in Software Engineering Education. Nevertheless, students face difficulties while learning software modeling. To teach software modeling effectively in computing courses, instructors who usually employed traditional methods could use active learning strategies. However, instructors are reluctant to change their teaching approaches due to several barriers that hinder the application of active learning strategies. Besides, relatively little research addresses how to mitigate them. The objective of this research is to help instructors implementing active learning strategies when teaching software modeling with UML diagrams. To achieve this objective, we conducted a Design Science Research (DSR). We proposed an artifact called OpenSMALS, an Open Repository for teaching Software Modeling applying Active Learning Strategies. OpenSMALS provides specific guidelines on how instructors can apply these strategies and helping instructors to identify the active learning strategies best suit their teaching context. We performed four DSR Design Cycles—in four different universities—to evaluate and evolve OpenSMALS. Our results show that OpenSMALS satisfactorily reduced the barriers faced by instructors, and it achieved an appropriate maturity level to be adopted by other instructors.

CCS CONCEPTS

• **Social and professional topics** → Computing education; *Computing education programs*; Computer science education; • **Software and its engineering** → **Software notations and tools** → **System description languages**; Unified Modeling Language (UML); Specification languages.

KEYWORDS

UML, Modeling Education, Software Engineering Education, Active Learning Strategies.

ACM Reference format:

Williamson Silva, Bruno Gadelha, Igor Steinmacher and Tayana Conte. 2019. Towards an Open Repository for Teaching Software Modeling from Active Learning Strategies. In *Proceedings of 42nd International Conference on Software Engineering - Software Engineering Education and Training (ICSE-SEET)*, May 23-29, Seoul, Republic of Korea. ACM, New York, NY, USA, 11 pages.

<https://doi.org/10.1145/3377814.3381709>

1 Introduction

Software modeling is a key concept in Software Engineering Education (SEE) [1-2]. According to Agner et al. [2], software modeling enables an in-depth understanding of specific concepts or problems using different detail levels. However, instructors and researchers reported that students face difficulties while learning software modeling because of the complexity of its concepts when compared to other aspects of Software Engineering [3,4]. Besides, students find it difficult to abstract real-world concepts and representing them in a model [3,4].

The challenges faced by students may arise from the way that the software modeling has been taught [5]. According to Connolly et al. [6], students often find it challenging to work on problems that do not have a well-defined solution, in which problems are ambiguous and vague, or when they need to apply the classroom examples to different scenarios or domains. To learn how to solve modeling problems, students need hands-on experience with practical scenarios, which could come from participating in actual software projects, simulations, role-playing, case studies, or other experiential learning activities [7-10].

Therefore, instructors need to adapt their pedagogical strategies to provide students with a challenging environment that actively involves them in the learning process [11]. Educational researchers suggest using Active Learning (AL) aiming to provide students with new experiences and learning opportunities, improving students' overall learning [12,13]. According to Bonwell and Eison [14], AL is typically defined as learning that requires students to engage cognitively and

*Article Title Footnote needs to be captured as Title Note

†Author Footnote to be captured as Author Note

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

ICSE-SEET'20, May 23-29, 2020, Seoul, Republic of Korea

© 2020 Copyright is held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-7124-7/20/05...\$15.00

<https://doi.org/10.1145/3377814.3381709>

meaningfully with the courseware. So, they are “*involved with the information presented, really thinking about it (analyzing, synthesizing, evaluating) rather than just passively receiving it*” [15]. The goal of AL is to provide opportunities for learners to critically think about the content through a range of activities that help prepare learners for the issues of professional situations [16].

The literature offers extensive evidence that AL strategies are more effective than a completely passive lecture in undergraduate Science, Technology, Engineering, and Mathematics courses [17,18]. Despite this evidence, the quantity and quality of evidence supporting AL strategies have not increased faculty and instructor adoption rates [18]. This can be explained by the fact that instructors face several barriers that hinder their use of AL strategies: (a) the time spent preparing to teach a class using AL strategies is too long [19]; (b) the instructors find it hard to cover the syllabus using these strategies [20]; (c) the potential negative students' responses when introducing new teaching strategies in class [21,22]; and (d) the lack of information about how to implement active learning strategies [23]. Regarding this last one, instructors have so many conflicting demands on their time, and few instructors have the time to immerse themselves in the education research literature. Minimizing the barriers to apply AL strategies in their classroom is the first step for helping to change the way the instructors teach software modeling [22].

In our research, the goal is to help the instructors applying AL strategies when teaching software modeling with UML diagrams. To achieve this goal, we applied a Design Science Research (DSR) approach, which has been widely applied to create and evaluate new artifacts in Information Systems and Educational Research [24]. In this paper, we propose an artifact called OpenSMALS (**Open** Repository for teaching **S**oftware **M**odeling from **A**ctive **L**earning **S**trategies). OpenSMALS is an open repository that aims to help instructors to implement AL strategies in their teaching context. The repository focuses on addressing the practical challenges faced by instructors by providing specific guidelines on how instructors can apply the strategy, modeling scenarios offered by other instructors, assessment questionnaires, and others. We conducted four DSR Design Cycles to evaluate and evolve OpenSMALS in four different universities. The empirical studies' results revealed limitations and opportunities for improvement in OpenSMALS. Initial evidences show that OpenSMALS helps the instructors to implement AL strategies.

Thus, the contributions of this paper include: (i) the creation of an open repository that explicitly and affordably guides instructors on how instructors should implement each of the AL strategies in the classroom; (ii) a set of OpenSMALS artifacts that contributed to instructors' identification and use of AL strategies; (iii) empirical evidence that OpenSMALS contributed to aiding instructors from four different universities on how to implement AL strategies during teaching software modeling. We believe that SEE community and researchers will take advantage of this work to improve their support for teaching modeling with UML, ultimately leading to more contributions to OpenSMALS.

The remainder of this paper is organized as follows: Section 2 presents the related works; Section 3 describes the research method used to create and evaluate OpenSMALS; Section 4

presents Design Cycles used to evaluate OpenSMALS; Section 5 presents implications; and Section 6 presents conclusions.

2 Related Work

There is a rich literature presenting ways to teach software modeling that makes it clear that the SEE community has dedicated a significant amount of effort to develop new pedagogic strategies that make teaching more attractive to students [25]. The active learning strategies have been gaining acceptance as higher education continues to shift toward student-centered learning [17]. Instructors can use these strategies to engage students in classroom activities by making them active stakeholders of the learning process. Below, we present some studies in which the authors applied AL strategies to teach software modeling.

García-Holgado et al. [26] implemented an active learning methodology based on projects (Project-Based Learning – PjBL), working with analysis models (use case model and problem domain models). The authors noticed an increase in the student success rate from 41.71% to 63.89%. Although they only applied it in a specific context, the experience can be adapted to similar content in other degrees and universities. Fioravanti et al. [16] also reported their experience applying a PjBL. They combined the strategy with project management for creating an environment in which students deal with managers and other stakeholders. The authors reported that including PjBL in the classroom brings innovation and dynamism in teaching modeling. In general, students were enthusiastic and had positive perceptions about PjBL and the importance of using real problems.

Similarly, Silva et al. [27] carried out an empirical study to investigate the influence of two active learning strategies: Problem-Based Learning and Learning from Erroneous Examples when teaching UML diagrams. They concluded that the diagrams created using the two methods presented similar levels of correctness and completeness and that both methods (a) helped in understanding the concepts of diagrams, (b) improved the interaction among the team.

Scanniello and Erra [28] used a strategy based on Think-Pair-Square while creating use case diagrams. In this strategy, the students started with individually modeling (Think), then worked in pairs (Pair), and finally worked in a group with an even number of students (Square) of four or more. The authors realized that diagrams improved as students moved from one stage to another, and students improved their ability to work collaboratively.

Although the literature addresses the use of active learning strategies in a software modeling context, we observed that there is a lack of studies reporting how instructors have conducted the process of selecting appropriate AL strategies nor pointing to a place or repository where instructors can find courseware, they need to apply AL strategies. This is essential due to the number of specific barriers that hinder their use of AL strategies.

3 Applying DSR to develop OpenSMALS

To understand how we can support instructors to implement active learning strategies when teaching software modeling, we

conducted research using the Design Science Research (DSR) approach [24]. DSR helps to create and evaluate new artifacts as they are developed and used to solve practical problems of general interest [29]. DSR approach is an iterative process, which combines both behavioral and Design Science paradigms, and comprises three interlinked research cycles [29]: the Relevance Cycle, the Design Cycle, and the Rigor Cycle. Figure 1 summarizes the primary information related to the DSR cycles in this research. There are intersections between the Relevance and Rigor cycles and the Design Cycle – i.e., the Design Cycle takes into account the Relevance Cycle (e.g., the artifact should meet the established requirements) and the Rigor Cycle (e.g., the development of the object should be grounded in scientific theories and methods).

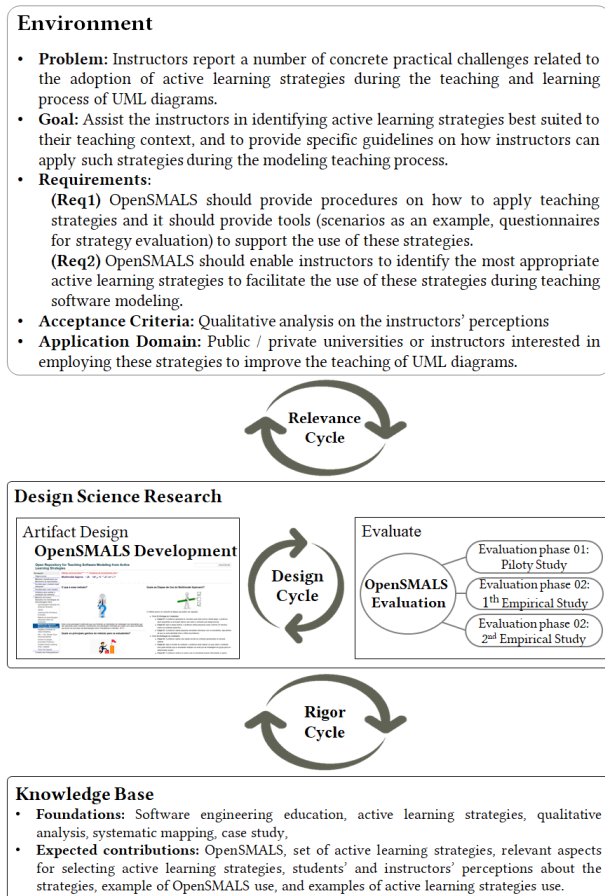


Figure 1. Overview of the Design Science Research cycles in this work – based on Hevner and Chatterjee [24].

The **Relevance Cycle** involves defining the motivation/problem to be addressed, the research requirements (e.g., the opportunity/ problem to be solved), and the acceptance criteria for the ultimate evaluation of the research results. The output from the DSR must be returned to the environment for study and evaluation in the application domain [24]. Our primary motivation for developing this research is related to the amount of evidence about the success of using active learning strategies

in the teaching process. Despite this evidence, the resistance of the academic staff often meets them [30,31]. These strategies demand that instructors invest time and effort to develop new learning materials, integrate modern technologies, and confront unexpected conditions [31]. When weighing the future advantage with the anticipated investment of effort, the common tendency is for many instructors to reject the desired change [30-].

For this reason, instructors often become not eager to master an innovative teaching strategy [32]. Given this problem, we defined two requirements that should be considered as criteria to develop and evaluate the OpenSMALS:

(Req1) OpenSMALS should provide procedures on how to apply teaching strategies and the tools to support the use of these strategies.

(Req2) OpenSMALS should enable instructors to identify the most appropriate active learning strategies to facilitate the use of these strategies during teaching software modeling.

We based these requirements on attributes that can contribute to the acceptance and adoption technologies developed [33]. To assess these requirements, we conducted a qualitative analysis of the instructors' perceptions who applied OpenSMALS.

In the **Design Cycle**, the goal is to develop and to evaluate a concrete solution that addresses the explicated problem and fulfills the defined requirements [24]. The development of artifacts may be driven by requirements, while the results of the application of this artifact on the problem context may corroborate or question the validity of these requirements. In the present work, the artifact we propose is a repository that aims to assist instructors in implement active learning strategies called OpenSMALS (**Open** Repository for teaching **S**oftware **M**odeling from **A**ctive **L**earning **S**trategies). OpenSMALS portal evolved based on evidence collected from a pilot study and two empirical studies, each one of them resulting in a version of OpenSMALS. Before the first Design Cycle, a pilot study was conducted as an illustrative case to assess if the solution was feasible and could solve the identified practical problem [24]. After that, we carried out two empirical studies (first and second DSR Design Cycles) in four different universities. In the third design cycle, we do not perform a new empirical study. We evolved OpenSMALS based on the improvement suggestions. The empirical studies conducted will be explained in detail in Subsection 3.1.

Finally, the **Rigor Cycle** refers primarily to generating and using knowledge [24]. The Rigor Cycle is achieved grounding theories and methods along with the domain experience and expertise from the foundation's knowledge base into the research and adding knowledge generated by the study to contribute to the growing knowledge base [29]. In this work, the main foundations are knowledge related to SEE, AL strategies, and a Systematic Literature Mapping. The main contribution to the knowledge base is OpenSMALS itself, as a new open repository that helps instructors identify AL strategies best suited to their teaching context. Additionally, (i) the process of using of OpenSMALS in real settings serves as an example for other instructors, (ii) the investigation performed to develop OpenSMALS, (iii) the set of AL strategies that can be used to teach UML diagrams, (iv) the aspects

to be considered when implementing the strategy, and (v) the knowledge related to the analysis of impacts of the strategy in teaching UML diagram.

3.1 Evaluating OpenSMALS

We carried out an empirical study to verify whether OpenSMALS produces the desired effects and whether a new iteration over the Design Cycle is needed [24].

3.1.1 Planning

We executed the study in four different Brazilian universities: Federal University of Amazonas (UFAM), State University of Maringá (UEM), Federal Technological University of Paraná (UTFPR) and Uninorte Laureate Universities. The study involved five instructors (I01, I02, I03, I04, and I05), and 163 students. Table 1 shows an overview of the instructors.

Before starting with the actual cycles of DSR, we decided to perform a pilot study. According to Nunamaker and Briggs [34], a Design Science researcher can perform a pilot study first as a useful basis for research. For the **pilot study (first cycle of Design Science)**, we selected an instructor (I01) that was teaching the Software and Analysis and Design course. This course is part of the 5th term (junior year) of the curriculum and is the first contact of students with software modeling in the major. The instructor had more than 17 years of experience in the software industry, with five years related to practical experience and teaching software modeling.

After that, we carried out two empirical studies. Empirical studies can be carried out to improve researchers' understanding of the application of technology and enable its refinement [35]. In the **first empirical study (second cycle of Design Science)**, we selected two other instructors (I02 e I03) who employed OpenSMALS in their courses. I02 and I03 have more than 16 years of experience in the software industry experience and teaching software modeling.

In the **second empirical study (third cycle of Design Science)**, I02 and I03 employed OpenSMALS once again in their courses. The instructors used OpenSMALS to choose other active learning strategies for teaching other UML diagrams. Besides, we selected two other instructors (I04 and I05). I04 has more than 17

years of experience in teaching software modeling, and I05 has three years of experience.

3.1.2 Execution

We followed the steps depicted in Figure 2, and each step is explained in the following. Before starting the use of OpenSMALS, the moderator delivered the consent form, and all the instructors signed it. After that, the instructors used OpenSMALS to define the teaching strategy they would use in the class, from a list available in the web portal.

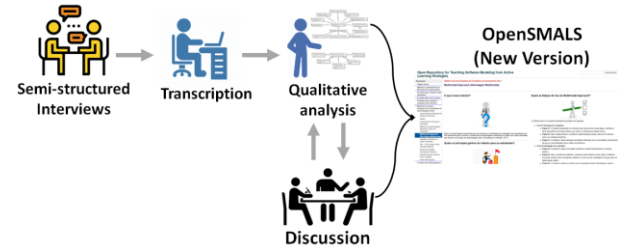


Figure 1. Overall method followed.

Table 1 presents the strategies selected by each instructor. After choosing the strategy, the instructor gathered more information about the chosen strategy to apply it. After the students performed the activity, the instructor discussed the solutions with the students, making them reflect on the problem and the solution they modeled. In this way, students could discuss and ask questions about the diagrams. After the instructor applied OpenSMALS, we conducted semi-structured interviews, which consisted of a mixture of open-ended and specific questions designed to elicit foreseen and unexpected information types [34]. We designed our interview script following the literature recommendations [37]. Interviews are commonly used in empirical software engineering and as part of Design Science Research [38]. Table 2 presents the script with the main questions used during the interview. The interviews were performed individually and lasted between 15 and 30 minutes. During the interviews, the instructor could express their perceptions about the use of OpenSMALS, and talk about the challenges faced in using this tool. We carried out this process for the pilot study and empirical studies.

Table 1. Information about the instructors and the class.

Evaluation	University	Instructor	Major	Course Name	Chosen Strategy	UML Diagram	# students
Pilot study	Class A (UFAM)	I01	Information Systems	Software Analysis and Design	Inspection-based strategy	Class diagram	14
					Think-pair-square	Sequence diagram	
First Empirical Study	Class B (UFAM)	I02	Computer Science	Introduction to Software Engineering	Positive examples Negative examples Learning based on similar systems	Use case diagram Use case description Diagram and textual description	28
	Class C (UEM)	I03	Computer Science	Software Engineering	Negative examples	Class diagram	
	Class D (UFAM)	I02	Computer Science	Software Analysis and Design	Inspection-Based strategy	Sequence Diagram	
Second Empirical Study	Class E (UEM)	I03	Computer Science	Software Engineering	Erroneous examples	Sequence diagram	35
	Class F (UTFPR)	I04	Information Systems	Software Engineering	Positive examples	Class diagram	16
	Class G (Uninorte)	I05	Information Systems	Software Engineering	Similar systems	Use case diagram	14

Table 2. Semi-structured script.

Part I - Participant Background
Q1 - How long have you been teaching in Software Engineering / Software Modeling?
Q2 - Do you have experience in the software modeling industry?
Q3 - In your classes, how do you teach software modeling for students?
Part II - About the using of OpenSMALS
Q4 - What is your perception about OpenSMALS?
Q5 - What is your perception of the strategies provided in OpenSMALS?
Q6 - What is your perception of the strategy information that is provided by OpenSMALS?
Q7 - Did you have any lessons learned after using OpenSMALS?

3.1.3 Data analysis

Our analysis was exploratory, aiming to generate new insights and provide a better understanding of the problem and proposed solution. We qualitatively analyzed the transcription of the answers following coding procedures [39]. The goal of the qualitative analysis was to identify the difficulties and benefits perceived by instructors after using OpenSMALS. By analyzing the instructors' additional comments, we created codes (relevant concepts for the understanding the perception about the strategies) related to the instructors' answers (participants' quotes). After this, we analyzed and grouped the codes according to their properties, forming concepts that represent categories and subcategories. The first author conducted the analysis and discussed it with the other three authors in multiple meetings. This discussion was performed to mitigate any potential bias in the coding process. The results of the qualitative analysis and the discussions carried out gave us insights to improve OpenSMALS.

4. Design Cycles

In this section, we present the Design Cycles performed to evaluate and evolve OpenSMALS.

4.1. First Design Cycle: Development of OpenSMALS (version 1)

Drawing from DSR [29], we proposed an open repository to support instructors in the identification of active learning strategies best suited to their teaching context. OpenSMALS reuses the knowledge provided by other proposals, relying primarily on active learning strategies used for teaching software modeling. To support instructors choosing the appropriate teaching strategies, we considered the strategies identified and curated in a Systematic Mapping (SM) conducted previously. From the SM, we identified the strategies that leverage any active learning strategy in their theoretical background. All the strategies made available in OpenSMALS are presented in Table 3. After identifying the strategies, we organized the knowledge about each strategy by creating a conceptual model described as a class diagram (see Silva et al. [46]).

Table 3. Selected strategies for composing OpenSMALS.

Name of strategies	# Authors
Learning based on similar systems	Stoecklin et al. [40]
Jigsaw	Portilho and Campos [41]
Learning from Erroneous Examples	Silva et al. [27]
Inspection-Based strategy	Kinjo et al. [42]
Multimodal Approach	Thevathayan and Hamilton [43]
Negative Examples	Bolloju et al. [44]
PBL + LBL Double Track Teaching Strategy	Wang et al. [45]
Positive Examples	Bolloju et al. [44]
Problem Based Learning Adapted	Silva et al. [27]
Think-Pair-Square	Scanniello and Erra [28]

To explain the conceptual model proposed, we developed a questionnaire based on the knowledge represented in the conceptual model. Based on the questionnaire responses provided by instructors, the OpenSMALS recommends a set of strategies according to the needs of the instructor interested in applying them. To achieve this, we created a decision tree with three levels. Each level has questions that guide the instructor to choose the most appropriate strategies. All possible questions and answers for this decision tree can be seen in Silva et al. [46]. In the web portal, we grouped information that is important so that the instructor can employ the chosen AL strategy (Figure 3).

Figure 3. Web portal with information on strategies.

4.1.1 Results of pilot study

We analyzed the instructor's perception (I01) about the use of OpenSMALS on choosing an active learning strategy to support the teaching UML diagrams (class diagram and sequence diagram). We present the main findings in Table 4.

The pilot study provided some causes of the negative results obtained with this instructor. Considering these issues, we

improved OpenSMALS to accommodate the suggestions, resulting in a second version of the OpenSMALS portal.

Table 4. Instructor I01's perception of OpenSMALS (v1).

<p>The suggestion of the questionnaire OpenSMALS was too generic</p> <p>"[...] I think that there were few questions and that these were too general [...] the answers I had were not according to my questions, I did not have that perception. If you had more targeted questions to know the characteristics of the class, the characteristics of the time the instructor has available to apply the strategies. I would use it again." – I01</p>
<p>The information available in OpenSMALS was enough to understand how to apply the strategies</p> <p>"OpenSMALS provide detailed steps on how to execute strategies with students [...]" – I01</p>
<p>OpenSMALS could suggest strategies according to the moment when the instructor wants to apply the strategy</p> <p>"OpenSMALS could ask me something like: do you want to use the strategy to teach the students to model, or do you want to use it to train more actively the lecture's content?" – I01</p>

4.2 Second Design Cycle: Development of OpenSMALS (v2)

Evaluating our result from the perspective of Req2 (OpenSMALS should enable instructors to identify the most appropriate active learning strategies), it is not possible to say that it was achieved in the pilot study. Thus, we revisited the conceptual model and analyzed the information collected for each strategy once again. As a result, we created a new version of the decision tree (a part of it is presented in Figure 4). The full version of this tree can be found in a Silva et al. [46]. We regrouped the questions according to the moment that the instructors expect to

use the strategy: at the beginning of the learning and after having taught the concepts about the models.

In the first moment, the portal directed the instructors to strategies that could be used from the beginning of student learning experience, so that students are responsible for their own learning. At that moment, instructors acted only as mediators, i.e., guiding students while using such strategies. In a second moment, instructors could choose strategies that they could apply after presenting the basic concepts about the model for the students. We defined these moments in this second version of the questionnaire because, according to Erickson [47], these moments are the most appropriate to introduce new strategies to improve the acquisition of new skills and knowledge. These moments of learning reveal to the instructors the new possibilities that students can achieve [48,49].

Regarding Req1 (OpenSMALS should provide procedures on how to apply AL strategies, and it should provide tools to support the use of these strategies), we observed that the information available on the portal was sufficient. Based on instructor I01's suggestion, we applied post-modeling questionnaires to evaluate the students' learning experience. Then, we added the students' perceptions about strategies in OpenSMALS (v2).

4.2.1 Results of first empirical study

We performed an empirical study to understand how instructors applied OpenSMALS. By qualitatively analyzing the data collected from interviews with instructors (I02 and I03). We identified the situations in which the difficulties and ease of use of OpenSMALS occurred. We found three benefits perceived by instructors in using OpenSMALS (v2) (Table 5).

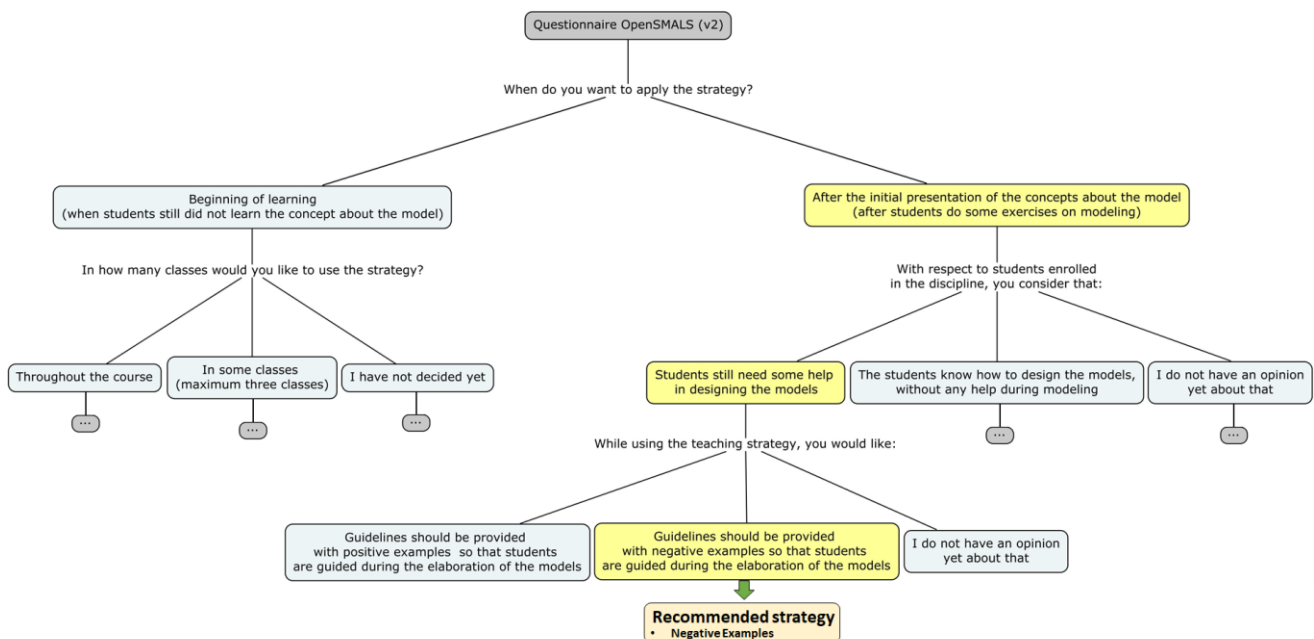


Figure 4. Extract of questions of questionnaire's decision tree – final version.

Table 5. Benefits perceived by instructors by using OpenSMALS (v2).

<p>OpenSMALS assists in class planning "OpenSMALS is very objective because it asks you how many classes you want to use. This is good because it will help you in your lesson planning [...] this enables you to tailor strategies according to your monthly planning" – I02</p>
<p>OpenSMALS recommends strategy according to the teaching context "OpenSMALS recommended something according to the options I choose in the questionnaire [...] I also found it interesting because it did not recommend me just one [strategy]." – I03</p>
<p>OpenSMALS suggests strategies that can be used independently of the diagram being taught "you can adapt the strategy recommended by OpenSMALS to any teaching context, whether it is for the use-case diagram or the class diagram." – I02</p>

The codes related to the above categories indicated that the instructors had a good perception of OpenSMALS (v2). However, the instructors (I02 and I03) identified some difficulties (Table 6).

Table 6. Difficulties perceived by instructors by using OpenSMALS (v2).

<p>OpenSMALS does not tell how the instructor should conduct student organization "I was left wondering if the students could work in groups or not." – I02</p>
<p>It is not explicit in OpenSMALS how to present the strategy for students "the descriptions of the examples are clear, in theory. But, for example, when I chose the strategy and saw that it had only text, I took those rules and related pieces of class diagrams to illustrate each of the negative problems. If I had an illustration for each of those problems, for people like me who like to illustrate, it would be cooler." – I02</p>
<p>There was a bias of the facilitator due to the help provided during the implementation of the strategy "if there were no facilitator, I would not have been able to finish on time, and it would have given me a little more work." – I03</p>

We also identified two codes that describe improvements in OpenSMALS (v2) (see Table 7).

Table 7. Improvement suggestions recommended by instructors for OpenSMALS (v2).

<p>Recommendation of the strategy should take into account the background of the instructor "if I were a freshman instructor in the first or second year of teaching and then I had three alternative strategies, I think I would get lost on which to choose. So, I think if I had something also related to the instructors' experience, maybe that would guide me better." – I03</p>
<p>Create a repository with worked examples that can be used as a basis for other instructors "a suggestion would be to have a repository with examples for using the method from materials that can be customized, which is something that instructors are always looking for." – I02 "there comes a time when you are out of time and have no choice but to give the traditional class. But if you can to set up a scenario bank and offer it as an add-on to OpenSMALS, I think it's going to be pretty cool." – I03</p>

The results of the previous Design Cycle indicated that OpenSMALS (v2) provided useful results regarding its use for instructor assistance compared to the first version. The results showed that instructors considered this new version (v2) useful and feasible to use. We noted in this study that the requirements for evaluating OpenSMALS were met. We perceived that, after improving the questionnaire, instructors reported that it

suggested AL strategies according to the context of teaching (so, we achieved Req2). We created a new version OpenSMALS based on improvements suggested by the instructors.

4.3 Third Design Cycle: Development of OpenSMALS (v3)

The first empirical study (Subsection 4.2.1) results provided us with evidence to make improvements that addressed the difficulties identified in the OpenSMALS. In Table 8, we present the improvements implemented in OpenSMALS (v3), see Figure 5.

Table 8. Suggestions identified in OpenSMALS (v2) and that to be improved in a new version.

Codes	Instructor	Implemented Solution
[Suggestion 1] Avoid the bias of the facilitator during the implementation of the strategy.	I02; I03	Create a repository where all the information needed by the instructor is available.
[Suggestion 2] Create a repository with worked examples that can be used as a basis for other instructors	I02; I03	Show in OpenSMALS how the instructor can organize the students.
[Suggestion 3] OpenSMALS could show how the instructor can conduct student organization.	I03	Add an appendix to OpenSMALS showing other ways to present strategies for students.
[Suggestion 4] OpenSMALS could explicit how to present the strategy for students	I02; I03	Add information about using other instructors in OpenSMALS.
[Suggestion 5] Recommendation of the strategy should take into account the background of the instructor	I03	

We had in mind that the goal of OpenSMALS was to be used by any instructor who wanted to minimize their effort to apply active learning strategies in their courses (**Suggestions 1 and 2**). Thus, we hypothesized that adopting the suggestion of improvement proposed by the instructors would help us to minimize this difficulty. We decided to turn OpenSMALS into an open repository in which all the artifacts that are necessary to implement the strategies are available to the instructors. The artifacts provided for instructors are:

- **Modeling scenarios:** if the instructors do not have scenarios available to apply strategies, the instructors can use some of the scenarios provided by the portal (Figure 6.a);
- **Evaluation questionnaire and a data collection worksheet:** if the instructors wish to evaluate the students' learning experience and want to present such results to the students to reflect on their learning (Figure 6.a).
- **Focus Group frame:** if instructors wish to gather qualitative data, we provide a focus group frame and instructions on how to conduct it with students (Figure 6.b).

Regarding **Suggestion 3**, for each strategy, we added a field showing that students can be grouped according to the learning objectives that the instructor expects students to reach: in groups (three or more students), in pairs, or individually. Regarding **Suggestion 4**, we included a document and a presentation on how to present and use the strategies in the class. This document can be found at the bottom of the page of each strategy. The instructor can choose how best to present these problems to students.

Open Repository for Teaching Software Modeling from Active Learning Strategies

Navigation [Teaching Methods Based on Active Learning Strategies >](#)

[Home](#)

Strategies classified by moments of learning

[Choose the most appropriate strategy here](#)

[Choose your scenario here](#)


[Artifacts to help evaluate strategies](#)

Teaching Methods Based on Active Learning Strategies

- Learning based on similar systems
- Jigsaw
- Learning from Erroneous Examples
- Método de Aprendizagem utilizando Dados de Inspeções
- Multimodal Approach
- Negative Examples**
- PBL + LBL Double Track Teaching Method
- Positive Examples
- Problem Based Learning (PBL) Adapted
- Think-Pair-Square


Negative Examples

What is this strategy?



Negative Examples is a teaching method based on an active learning strategy that provides problems that are associated with negative modeling descriptions and examples so that they are incorporated into the modeling teaching environment.

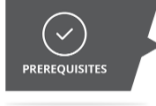
What are the main gains of the strategy for students?



This method allows students to:

- Understand the main difficulties other students had when modeling a particular model.
- Check from the examples worked in class, which are the primary defects that the other students committed to developing a model.
- It allows students to realize the defect and not realize it again.

What are the prerequisites for using this strategy?




- Students need to know the model notation and have previously solved modeling exercises.
- If the instructor wants to use real scenarios, the time for modeling can vary from one to two classes, each class with 1h40min.

Strategy Classification: Strategy for problem-solving.

References
 Bolloju N, Schneider C, Vogel D. Asymmetrical effects of using positive and negative examples on object modeling. In Information Systems Development 2011 (pp. 85-96). Springer, New York, NY.


What are the steps to applying the Negative Examples Strategy?



The strategy has a set of steps that can be followed:


- **Step 01:** The instructor introduces the concepts about the model during class and then provides a scenario for students to perform the modeling..
- **Step 02:** Students build a model based on the scenario delivered by the instructor.
- **Step 03:** The instructor conducts training on the difficulties that are most common during modeling of the requested artifact (below are some defects about some models to assist instructors).
- **Step 04:** The instructor gives the students a set of models with the primary defects marked.
- **Step 05:** The instructor asks students to model a new diagram based on training and a

What is the time for applying the strategy?



To apply the strategy, a class of 1h40min is required.

Come check out some assessments made with this strategy?



[Click here](#) to see the evaluations performed with this strategy.
 Have you used this method? How about sharing with us what was the students' perception of the method? Just [Click Here](#).

Figure 5. A new version of OpenSMALS portal (v3)

For **Suggestion 5**, we added a page about the instructors. In this field, each instructor will be able to provide information about the class that implemented the strategy.

4.3.1 Results of second empirical study

The second empirical evaluation aimed to understand if it was possible to reduce the problems identified in v2 by implementing the solutions proposed. After analyzing the interviews with the instructors who used v3 (I02, I03, I04, and I05), we identified some of the positive and negatives points perceived when using this version of OpenSMALS (Table 9). The Improvement Suggestions for OpenSMALS (v3) category contains codes that describe improvements and new features described by instructors for OpenSMALS (v3), see Table 10.

The results of this third Design Cycle showed the instructors better accepted OpenSMALS (v3). Besides, we could link the reasons behind this improvement with the solutions implemented

based on the feedback gathered on the use of OpenSMALS (v2). We also noted that the barrier related to the support to apply the strategies was also solved in this version (v3) of OpenSMALS (Req 1). With OpenSMALS supporting the instructor in the process of how to apply the strategies, the instructor could participate more actively in the teaching process with the students. As expressed through instructor reflections, establishing a routine with active strategies seemed to set the tone and expectation for students to engage during class. By understanding the causes of the difficulties perceived by the instructors, we have been able to define strategies to overcome these difficulties.

We noticed several positive points on the use of OpenSMALS, reported mostly by instructors with many years of industry experience and teaching software modeling. However, some drawbacks and suggestions for improvement were still reported by the instructors. We use these points to improve OpenSMALS by generating its fourth version.

Table 9. Benefits and difficulties perceived by instructors who used (v3).

<p>[Benefit 1] Instructors chose the strategies based on the suggestions provided by the questionnaire</p> <p>“since I liked the OpenSMALS recommendations from the previous version, now I answered the questionnaire, and it recommended me two strategies, negative examples, and erroneous examples.” – I03 (who has previously had previous experience with OpenSMALS)</p> <p>“I think this works very well when you have a very specific thing to apply; I do not want to see the other strategies; I want a recommendation of a strategy that makes the student learn a certain model” – I05</p>
<p>[Benefit 2] OpenSMALS recommends the strategy according to the teaching context</p> <p>“although I have not looked at all the other strategies, I think he suggested a good strategy; it was quite what I wanted to use with my students.” – I04</p>
<p>[Benefit 3] The stages of use supported the instructors in the application of the strategies</p> <p>“I read the descriptions and found them very easy to understand (...) OpenSMALS describes the steps in a clear way (...) I applied the strategy with my students using only the steps of using strategy, and I found it very easy.” – I03</p>
<p>[Benefit 2] The material provided by OpenSMALS helped instructors during the implementation of the strategies</p> <p>“OpenSMALS provided me with several supporting documents to implement the strategy, (...) we can use the scenarios and a strategy assessment form.” – I02</p>
<p>[Difficulty 1] The students’ and instructors’ perceptions did not influence the choice of strategies</p> <p>“I looked at these perceptions, but I did not pay attention to those perceptions, and I used the strategy because I want to have this experience.” – I03</p>
<p>[Difficulty 2] It is not explicit in OpenSMALS the instructor’s effort to apply the strategies</p> <p>“before applying the strategy, I kept thinking in the effort to prepare the material, and OpenSMALS does not provide this information.” – I04</p>

Table 10. Improvement suggestions recommended by instructors for OpenSMALS (v3).

<p>OpenSMALS needs to present other ways how instructors can use the strategies</p> <p>“I think it would be worthwhile also to put some customizations of how other instructors used the strategies, e.g., we suggest that it be applied like this, but we have reports from other instructors who applied otherwise... that’s pretty cool for other instructors to see that there is not only one way to apply the method.” – I05</p>
<p>OpenSMALS could provide the oracles of the scenarios</p> <p>“I believe that instead of just leaving the scenario, also leave one of the possible solutions for instructors to use in the classroom.” – I05</p>
<p>OpenSMALS could provide scenarios for specific models</p> <p>“It could also have specific scenarios for specific diagrams, for example, scenarios for sequence diagram, class diagram, among others.” – I05</p>

4.4 Fourth Design Cycle: Development of OpenSMALS (v4)

In the previous Design Cycle (using v3), the instructors reported that OpenSMALS is not explicit in estimating the effort that the instructor will have to apply such strategies. In this sense, we talked to each of the instructors who applied the strategies and asked how much effort—in hours—they spent to prepare all the material for the class. Then, for each of the strategies used, we added a section called “*How long does it take to implement the strategy,*” with the time that previous instructors took to prepare the materials.

Concerning “*providing scenarios for specific diagrams,*” we have improved the scenarios that were available in OpenSMALS. For each scenario, we created questions related to the UML diagrams, like the class diagram, use case diagram, a sequence diagram, and others. We provided possible solutions to each of the problems that are required in each scenario. Besides, given the open nature of the portal, and to increase the number of scenarios available in OpenSMALS, instructors who used the strategies before kindly provided new scenarios, so that they could be incorporated into OpenSMALS portal and could be used by other instructors.

Due to the acceptance of OpenSMALS by the instructors and the improvements made to the portal, in this fourth Design Cycle, we do not perform a new empirical study. We only evolved OpenSMALS based on the improvement suggestions. The OpenSMALS portal’s final version can be founded at <<https://sites.google.com/site/activelearningmethods/>>.

5 Implication

Software Engineering Students: According to instructors’ reports, by using the strategies suggested by OpenSMALS, the students were more engaged and encouraged to participate in the activities. Besides, some of the instructors reported that students performed well in disciplines, that is, designing correctly and completely diagrams. In particular, I03 stated that —after grading the elaborated models and administering an exam— “*looking proportionately at the number of errors of the exercises and in the exam compared to the previous classes, these students committed less mistakes considering those classic errors.*” Studies report that students who learn through active learning strategies have more

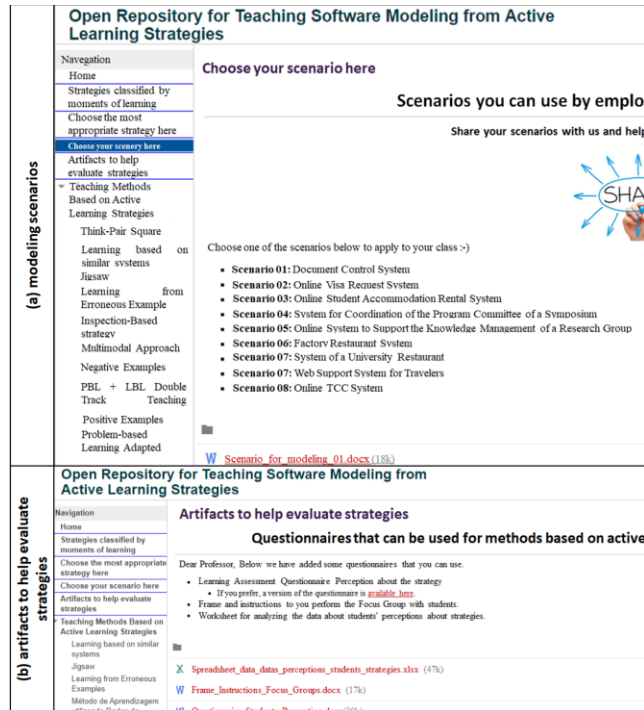


Figure 6. Web pages with artifacts that instructors can employ in their courses.

varied and in-depth skills than students from other universities where active learning strategies are not adopted/used [50]. According to Kim [51], these strategies also help the students to better self-assess and to be aware of whether they are more of a thinking-, action- or rather people-focused personality. The regular feedback helps them to know their own strengths and to work on their weaknesses. Therefore, these points imply that active learning strategies allow students to expand their learning and move beyond a conventional Software Engineering course.

Software Engineering Instructors: Using the active learning strategies suggested by OpenSMALS, instructors could achieve their strategic goals of having a hands-on approach during teaching. Each semester, instructors encounter new challenges and problems. However, by employing these strategies, the knowledge passed on to students greatly enriches daily teaching. In particular, I03 indicated that when using such strategies in the classroom, *“I feel updated, because one of the things you notice is that after a while working on the same thing, every year that same thing, in the same way, gives a feeling of stagnation. And that sense of seeking a different way of teaching also gives me greater motivation. In addition to professional motivation, in order to improve teaching, I also feel personal motivation.”* In addition, I02 stated that *“the active learning strategies already bring a huge gain simply because they change the way you teach modeling (...) Only you present in different ways, you already help students a lot with this.”* This means that more instructors can be motivated and encourage the use of active learning strategies in their subjects by learning more about the benefits of using these strategies with students. When the instructors introduce this type of strategy in the classroom, we expected the students to participate and become more involved in the class. These strategies can facilitate communication between instructors and students through rapid and constant feedback that is a characteristic feature of this type of teaching strategy. In this way, OpenSMALS can assist in this process, either through the use of strategies or feedback questionnaires answered by the students.

Researchers in Software Engineering: There are many gaps in mapping the strategies and barriers faced by instructors, which can be explored in future research. More research is necessary to investigate the moment of application of strategies and what are the most appropriate strategies to teach a particular diagram. As quoted by I03, *“Is there a most appropriate way using negative examples first and then erroneous examples to teach modeling for students?”* It would be of great interest to analyze how these strategies positively influence student learning. In particular, it would be interesting to understand the motivation and demotivation factors influencing the instructors in using OpenSMALS.

6 Conclusion

The use of active learning strategies has been gaining prominence in computing courses. However, instructors indicate several barriers that hinder their use of AL strategies. To help instructors by implementing these strategies when teaching software modeling with UML diagrams, we conducted research

using the Design Science Research approach. The artifact we developed and evaluated / evolved was OpenSMALS, an open repository that helps instructors to identify AL strategies best suited to their teaching context. OpenSMALS was designed to address the practical challenges faced by instructors during the application of these strategies. To this, it provides specific guidelines on how instructors can apply each strategy, modeling scenarios offered by other instructors, assessment questionnaires, and others.

We performed a pilot study and two empirical studies to understand how instructors apply OpenSMALS in their teaching context. Finally, we decided to evolve OpenSMALS for an open repository, where instructors can search and share their experiences using active learning strategies. We made available on the portal all the artifacts needed to apply each strategy. Thus, instructors can apply the strategy without the need of the researchers. We decided to make it available because educational literature frequently does not always provide details on how to implement active learning strategies [22]. As a result, instructors sometimes are unable to identify the critical elements for successfully implementing a classroom strategy [23]. Due to this context and the existing barriers, most instructors decide to follow the traditional teaching. OpenSMALS portal is an effort to create a tool that explicitly and affordably guides instructors on how each of the active learning strategies should be implemented in the classroom (with accountability, logic development, and apprehension reduction) during teaching software modeling. We highlight that proper implementation of active learning strategies does not guarantee improvements in student learning. However, it increases the success chances over the traditional approach [17].

Although we analyzed data from various sources, different classrooms, and instructors, we are aware that each instructor who applied the OpenSMALS has his/her singularities, and the instructor might be inherently biased. Because that, currently, we are working on new studies with more instructors and in other teaching contexts to identify new factors that may influence the use of OpenSMALS, as well as new barriers faced by instructors when employing active learning strategies. Besides, we believe that applying Active Learning is particularly challenging in larger courses. In this sense, we also intend to carry out new empirical studies to investigate whether OpenSMALS is scalable to more extensive courses with hundreds of students.

ACKNOWLEDGMENT

We would like to acknowledge the financial support granted by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001, and process 175956/2013, FAPESP through process number 009/2017; CNPq processes 311494/2017-0, 430642/2016-4, 423149/2016-4, 204081/2018-1/PDE. We would like to thank all the instructors who participated in the execution of the empirical studies, and help us to improve OpenSMALS, namely Thelma Colanzi Lopes, Reginaldo Ré, and Jacilane Rabelo.

REFERENCES

- [1] H. Störrle, How are Conceptual Models used in Industrial Software Development? A Descriptive Survey, in *21th International Conference on Evaluation and Assessment in Software Engineering (EASE'17)*, pp. 160-169, 2017.
- [2] L. T. Agner, T. C. Lethbridge, and I. W. Soares, Student experience with software modeling tools, in *Journal of Software & Systems Modeling*, pp. 1-23, (2019).
- [3] Z. Ma, An approach to improve the quality of object-oriented models from novice modelers through project practice, in *Frontiers of Computer Science* 11(3), (2017) 485-498.
- [4] V.Y. Sien, An investigation of difficulties experienced by students developing unified modelling language (UML) class and sequence diagrams, in *Computer Science Education* 21(4) (2011) 317-342.
- [5] R. Szmurlo, and M. Śmialek, Teaching software modeling in a simulated project environment, in *9th Int. Conf. on Model Driven Engineering Languages and Systems*, pp. 301-310, 2006.
- [6] T. M. Connolly, M. Stansfield, and T. Hainey, An application of games-based learning within software engineering, *British Journal of Educational Technology* 38 (3) (2007) 416-428.
- [7] S. Kurkovsky, S. Ludi, and L. Clark, Active Learning with LEGO for Software Requirements, in *50th ACM Technical Symposium on Computer Science Education*, pp. 218-224, 2019.
- [8] K. Al-Tahat, An innovative instructional method for teaching object-oriented modelling, in *International Arab Journal Information Technology* 11(6) (2014) 540-549.
- [9] Y. T. Lin, Impacts of a flipped classroom with a smart learning diagnosis system on students' learning performance, perception, and problem-solving ability in a software engineering course, in *Computers in Human Behavior* 95 (2019) 187-196.
- [10] E. M. Choi, Applying inverted classroom to software engineering education, *International Journal of e-Education, e-Business, e-Management and e-Learning* 3(2) (2013), 121-125.
- [11] S. A. A. de Freitas, W. C. Silva and G. Marsicano, Using an Active Learning Environment to Increase Students' Engagement, in *29th International Conference on Software Engineering Education and Training (CSEET'16)*, pp. 232-236, 2016.
- [12] T. Briggs, Techniques for active learning in cs courses, in *Journal of Computing Sciences in College* 21 (2) (2005) 156-165.
- [13] M. Prince, Does active learning work? A review of the research, in *Journal of Engineering Education* 93 (2004) 223-231.
- [14] C. C. Bonwell and J. A. Eison, Active learning: Creating excitement in the classroom, in Washington, DC: School of Education and Human Development (1991).
- [15] A. King, From sage on the stage to guide on the side, in *College Teaching* 41 (1993) 30-35.
- [16] M. L. Fioravanti, B. Sena, L. N. Paschoal, L. R. Silva, A. P. Allian. E. Y. Nakagawa, S. R Souza, S. Isotani and E. F. Barbosa, Integrating Project Based Learning and Project Management for Software Engineering Teaching: An Experience Report, in *Proc. of the 49th ACM Technical Symposium on Computer Science Education*, pp. 806-811, 2018.
- [17] S. Freeman, L. S. Eddy, M. McDonough, M. K. Smith, N. Okoroafor, H. Jordt and M. P. Wenderoth, Active learning increases student performance in science, engineering, and mathematics, in *Proceedings of the National Academy of Sciences* 111(23), 8410-8415, 2014.
- [18] J. M. Fraser, A. L. Tinman, K. Miller, J. E. Dowd, L. Tucker, and E. Mazur, Teaching and physics education research: bridging the gap. Reports on Progress in Physics 77 (3) (2014) 1-17.
- [19] S. Tharayil, M. Borrego, M. Prince, K. A. Nguyen, P. Shekhar, C. J. Finelli and C. Waters, Strategies to mitigate student resistance to active learning, in *International Journal of STEM Education* 5(1) (2018) 1-16.
- [20] S. E. Shadle, A. Marker and B. Earl, Faculty drivers and barriers: laying the groundwork for undergraduate STEM education reform in academic departments, in *International Journal of STEM Education* 4(8): (2017) 1-13.
- [21] M. T. Chi and R. Wylie, The ICAP framework: Linking cognitive engagement to active learning outcomes, in *Educational Psychologist* 49(4) (2014) 219-43.
- [22] J. Michael, Faculty perceptions about barriers to active learning, in *College teaching* 55(2) (2007) 42-7.
- [23] M. Borrego, S. Cutler, M. Prince, C. Henderson, and J. E. Froyd, Fidelity of implementation of research-based instructional strategies in engineering science courses, *Journal of Engineering Education* 102 (2013) 394-425
- [24] A. R. Hevner and S. Chatterjee, Design Research in Information Systems, Ed. Springer New York 2010.
- [25] L. T. Agner and T. C. Lethbridge, A survey of tool use in modeling education, in *20th Int. Conf. on Model Driven Engineering Languages and Systems (MODELS)*, p. 303-311, 2017.
- [26] Garcia-Holgado A, Garcia-Peñalvo FJ, Rodríguez-Conde MJ. Pilot experience applying an active learning methodology in a Software Engineering classroom, in *Global Engineering Education Conference (EDUCON)*, pp. 940-947, 2018.
- [27] W. Silva, I. F. Steinmacher and T. U Conte, Is It Better to Learn from Problems or Erroneous Examples?, in *30th Int. Conf. on Software Engineering Education and Training (CSEE&T)*, pp. 222-231, 2017.
- [28] G. Scanniello and U. Erra, Distributed modeling of use case diagrams with a method based on think-pair-square: Results from two controlled experiments, in *Journal of Visual Languages & Computing* 25(4) (2014) 494-517.
- [29] A. R. Hevner, A Three Cycle View of Design Science Research, in *Scandinavian J. Inf. Syst* 19(2) (2007) 87-92.
- [30] D. Pundak and S. Rozner, Empowering engineering college staff to adopt active learning methods, in *Journal of Science Education and Technology* 17(2) (2008) 152-63.
- [31] F. Zellweger, Overcoming Subcultural Barriers in Educational Technology Support, in *18th annual conference of the Consortium of Higher Education Researchers*, pp. 1-14, 2005.
- [32] Eickholt, J., Barriers to Active Learning for Computer Science Faculty. *arXiv preprint*, pp 1-13, 2018.
- [33] D. Bolchini, and F. Garzotto, Quality of Web usability evaluation methods: an empirical study on MiLE+, in *International Workshop on Web Usability and Accessibility*, pp. 481 - 492, 2007.
- [34] J. Nunamaker and R. Briggs, Toward a Broader Vision for Information Systems, in *ACM Transactions on MIS* (2:4) (2011) 20:1-20:12
- [35] F. Shull, J. Carver and G. H. Travassos, An empirical methodology for introducing software processes, *ACM SIGSOFT Software Engineering Notes* 26(5) (2001) 288-296.
- [36] M. D. Myers and M. Newman, The qualitative interview in IS research: Examining the craft, in *Inf Organ* 17(1) (2007) 2-26.
- [37] P. Runeson and M. Höst, Guidelines for conducting and reporting case study research in software engineering, in *Empirical Software Engineering* 14(2) (2009) 131-134.
- [38] C. Wohlin and A. Aurum, Towards a decision-making structure for selecting a research design in empirical software engineering, in *Empirical Software Engineering* 20(6) (2015) 1427-1455.
- [39] A. Strauss and J. Corbin, Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory. 2nd ed. SAGE Publications, London, 1998.
- [40] S. Stoecklin, D. D Williams, and R. Swain, Understanding object-oriented systems specifications using familiar systems, in *Int. Conf. Software Engineering: Education & Practice*, pp. 10-15, 1998.
- [41] J. A. P. S. Portillo and P. G. Campos, The jigsaw technique: experiences teaching analysis class diagrams, in *Mexican Int. Conf. on Computer Science*, pp. 289-293, 2009.
- [42] T. Kinjo, Y. Ohgame and A. Hazeyama, An Object-Oriented Modeling Learning Support System Using Inspection Comments, in *Journal of Information and Systems in Education* 5(1) (2006) 66-75.
- [43] C. Thevathayan and M. Hamilton, Imparting software engineering design skills, in *Proc. Nineteenth Australasian Computing Education Conference*, pp. 95-102, 2017.
- [44] N. Bolloju, C. Schneider and D. Vogel, Asymmetrical effects of using positive and negative examples on object modeling, in *Information Systems Development*, pp. 85-96, 2011.
- [45] Y. Wang, R. Su and G. Li, Research on PBL and LBL Double Track Teaching Model in Unified Modeling Language Teaching Based on Outstanding Engineers, in *Informatics and Management Science*, pp. 381-386, 2013.
- [46] W. Silva, B. Gadelha, I. F. Steinmacher, and T. U Conte, Open_Repository: Support_Material, Available in <<https://doi.org/10.6084/m9.figshare.10074101.v1>>.
- [47] F. Erickson, Taught cognitive learning in its immediate environments: A neglected topic in the anthropology of education, in *Anthropology & Education Quarterly* 13(2) (1982) 149-180.
- [48] J. Dewing, Moments of movement: active learning and practice development, in *Nurse education in practice* 10(1) (2010) 22-26.
- [49] S. De Weerd, F. Corthouts, H. Martens and R. Bouwen, Developing professional learning environments: model and application, in *Studies in Continuing Education* 24(1) (2002) 25-38.
- [50] M. Shaw and J. E. Tomayko, Models for Undergraduate Project Courses in Software Engineering, in *SEI Conference on Software Engineering Education*, pp. 33-71, 1991
- [51] M.-S. Kim, A team building algorithm based on successful record for capstone course, in *Global Journal of Engineering Education*, pp. 243-248, 2017.