Climb Your Way to the Model: Teaching UML to Software Engineering Students

Teaching Case

Naomi Unkelos-Shpigel[1,2(✉)], Julia Sheidin[1,2], and Moran Kupfer[2]

[1] Software Engineering Department, Ort Braude College, Karmiel, Israel
{naomius, julia, moran}@braude.ac.il
[2] Information Systems Department, University of Haifa, Haifa, Israel

Abstract. Unified Modeling Language (UML) courses are an essential part of software engineering curricula. There is increasing evidence that embedding active learning techniques in courses in general, and in UML courses in particular, increases students' motivation and performance. In this paper, we contribute to this body of work by presenting model for embedding active learning in an undergraduate UML course. The model was used throughout a course, providing interesting results, and promoting students' participation and motivation. We present our insights and plans for further inspection of the model.

Keywords: UML · Active learning · Motivation

1 Introduction

Software engineering and information systems students are required to take a software-modeling course, as part of their training as software designers. The course usually consists of learning several UML (Unified Modeling Language [7]) diagrams. In some cases, the course is taught before the students have practiced any object-oriented language (the design-first approach [2]).

The first UML course presents several challenges to both instructors and students [3]. First, for understanding and fully implementing diagrams, a lot of practice is needed. Students are usually required to submit 3–4 homework tasks per semester. As they need a lot of training in understanding and designing, simply performing the tasks is hardly enough training. Second, as the teaching method is traditionally frontal, the instructor cannot get a clear picture of students' perceptions, and in particular, their misconceptions. Third, as UML diagrams usually consists of a large number of details, and several modeling variants are available, solving questions with the students on the board is usually not enough to cover many solution options. Fourth, as the modeling requires a cognitive resource, students lack the motivation of performing the task [5].

Several teaching mechanisms and tools were offered in recent years to improve student's motivation and understanding. These methods all rely on active learning during the lecture and recitation. In example, the flipped classroom [5] is a learning method that includes both learning at home and performing practical activities in the

H. A. Proper and J. Stirna (Eds.): CAiSE 2019 Workshops, LNBIP 349, pp. 40–46, 2019.
https://doi.org/10.1007/978-3-030-20948-3_4

classroom. However, this method requires a great deal of time investment from both students and teachers. Additional teaching method are in order.

This paper presents the summary of a teaching case of a UML modeling course called "Climb Your Way to The Model". The teaching method drew inspiration from the onion model for collaborative learning [9], the 21st century skill movement [4] and motivation theories such as the Self Determination Theory (called SDT henceforth), [8], and the 4C's Model [6]. These motivation theories discuss how to motivate employees to take active part in the work, and to encourage them to strive for more productive behavior, mainly by encouraging intrinsic and extrinsic motivation [8], and by achieving a state where the worker is immersed into the task [8]. The Kahoot! Application[1] was used each lecture, to test students' knowledge from the previous lecture, and to present the results to all the class in a gamified manner. The Moodle environment[2] was used to perform additional collaborative exploratory assignment during class. Google forms were used for a reflection.

Leveraging on the principles of collaborative and gamified tools for education, our work fills this gap by asking the following research questions: (1) How can we promote software engineering students' achievements and participation in UML design courses? (2) What are the benefits of embedding active learning techniques in UML courses?

In the rest of paper, we first discuss the background for the teaching case. We then introduce the teaching method (Sect. 3). Section 4 presents students' responses to the teaching method. We conclude with a discussion of the findings and further work.

2 Scientific Background

2.1 Active Learning in UML Teaching

Trying to deal with the challenge of teaching an abstract model such as UML, different research works described tools and techniques for active learning. In example, Hansen et al. [4], used collaborative tools, involving instructors and students, to achieve a more usable and functional UML diagrams. However, their research claim that a previous knowledge in object-oriented programming is needed. Briggs [1] presents another case of an experiment for active learning in UML teaching, using simple tasks given to the students, with no special tools required.

Yet, these examples lack the aspect of students' reflection on the task, and a rigorous reasoning for the structure of the task, resulting in an increase of students' motivation and performance.

2.2 Cognitive Theories

Several cognitive theories address the topic of encouraging motivation for increasing the participation and motivation during the semester in learning tasks. Here we briefly present two of the most influential theories in this field.

[1] https://kahoot.com/.

[2] https://en.wikipedia.org/wiki/Moodle.

The 4 C's of 21st century skills [6] are some of the learning strategies in today's environment. They state that if students want to compete in today's global society, they must be proficient: (1) communicators, being able to identify the information needed and how to use or leverage it effectively; (2) creators: can create creative design and possess out of the box thinking ability; (3) critical thinkers: analyze mistakes and improve their thought processing, and (4) collaborators: work effectively with diverse teams, making necessary compromises to accomplish a common goal, and assuming shared responsibility for collaborative work.

The Self Determination Theory (SDT) [8] presents a continuum of motivation types, from intrinsic motivation that emerges from the employee, to extrinsic motivation created by rules and regulation in the workplace. Although intrinsic motivation is considered to be linked to positive human behavior, SDT suggests that proper use in extrinsic motivation can lead to motivated behavior. SDT suggests that there are three basic needs that need to be fulfilled in order to increase motivation: *Competence* - seek to control the outcome and experience mastery; *Relatedness* - will to interact, be connected to, and experience caring for others; *Autonomy* - desire to be causal agents of one's own life and act in harmony with one's integrated self.

The proposed solution is described in the next section.

3 The Course Structure

The course is a mandatory course given in one of the leading academic colleges in Israel, at the department of software engineering. We covered five main UML diagrams in our course (see Table 1).

Table 1. Subjects and activities during the course

	Subject	Active learning task
1	Requirement engineering methods	Lecture: Collaborative task - software that failed Recitation: Data collection practice - pairs
2	Use Case diagram	Lecture: Use case modeling - pairs Recitation: Use case modeling - pairs
3	Activity diagram	Lecture: Activity diagram - pairs Recitation: Activity modeling - pairs
4	Class diagram	Lecture: Use case modeling and Class - pairs Recitation: Class modeling - pairs
5	Sequence diagram	Lecture: Class and sequence diagram - pairs Recitation: Exam questions solving, and peer review - individuals
6	Statechart diagram	Lecture: Statechart diagram - pairs Recitation: Statechart modeling - pairs

We teach the course for three years, and gradually embedded various individual and group activities during the classes. The course is taught in the design-first approach [2], meaning – before the students learnt to program in an object-oriented language.

Building this model, we relied on the principles of the 4 C's model – students were treated as *creators* and *communicators*, responsible for dealing with new and complex problems by teamwork. They were also collaborators, as they changed teams each class. Finally, they were also *critical thinkers*, as we asked them to refer to other students' work from the previous lecture. We also provided them with conditions for the three basic needs of SDT – they worked on each exercise with little instruction (autonomy), worked in small teams and helped each other to understand the task (relatedness), and dealt with tasks that were more challenging than what they learn during the frontal class (competence).

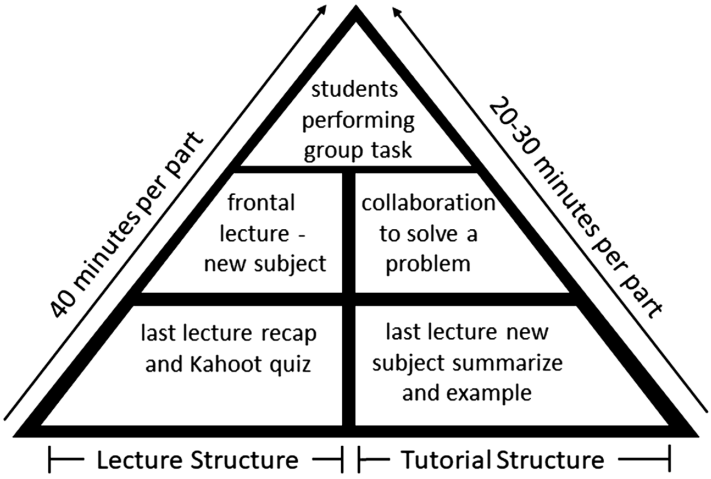

Fig. 1. The two-way climb model

This paper refers to a semester, which included 175 students, all in the second year of their studies. The course had a weekly three academic hours lecture, followed by a two academic hours tutorial (not at the same day) and consisted of three home works and a final exam. 8% of the grade are given for attending at least 80% of all class activities (in both lectures and recitations).

Each lecture consisted of three academic hours, which were divided, according to "Climb Your Way to The Model" model we designed, into three parts. Two parts were frontal, and the last one active learning (see Fig. 1). The first hour was devoted to recap of the last lecture, performing Kahoot quizzes, and reviewing students' models form the previous week; the second hour consisted of frontal teaching, presenting a new subject; the third hour was dedicated to active learning. Students worked in small groups, dealing with the new model they have just learned, and trying to produce a solution.

Each tutorial consisted of two academic hours, which were divided into three parts. One frontal, one collaboration and the last one active learning (see Fig. 1). The first 30 min were devoted to lecture summarize and exercise example of the new subject learned in the last lecture. The next 30 min consisted of collaboration to solve an exercise with discussion on different approaches to solution. The last 20 min included independent problem solving, where students worked in small groups, dealing with an exercise where they had to produce a solution on their own.

4 Insights and Students Perceptions Toward the Task

Several insights arose during the semester, which we categorized according to our research questions:

RQ1 – we noticed that indeed there was a significant increase in both student participation and solution quality:

- *Student attendance* – at the previous to our teaching method semester, where the teaching was frontal, many students stopped coming to class, usually at week seven. In this semester, we experienced a high rate of attendance throughout the semester.
- *Solutions complexity* – as students solved various problems during the classes, their solutions included interesting and even complex variants, even after a class or two on a subject. In example, they used constraints in class diagrams, inheritance in use case diagrams, and loops and multiple objects in sequence diagrams [6], in state chart [6], they used nested states and variables.
- *Students as thinkers, teachers as mentors* – in both lectures and recitations, during the active learning session, we served as mentors rather than teachers. In many cases, the students came up with interesting variants that were different from our solution, so we could encourage them to follow their line of thinking and construct an interesting new solution. In the following lecture, we shared these variants with the entire class. This method had two advantages: we exposed the entire class to multiple solutions and discussed the advantages and drawbacks of each solution.

RQ2– we asked the students to fill a survey, reflecting on the different activities they performed in the course. 52 students filled the survey. Here we had several interesting insights (see Table 2):

- *Activities cs. Kahoot* – Though Kahoot! Quizzes were very popular in previous semester, 70% of the students this semester thought they were redundant. 72% of the students said the class activities were helpful for their learning.
- *Promoters of iterative work and team work* – 55% of the students felt that the tasks they performed helped them to enrich their knowledge each week. 70% of the students felt that team work was better than solving the task on their own.

The quotes also comply with the cognitive theories we relied on. The students expressed in their feedback the characteristics of SDT – they expressed capability to perform the task (competence); they felt that they had solved the problems on their own

Table 2. Reflection (representative sample) on the activities from the student' perspective

Quote	SDT	4C's
The activities helped me to develop a technique for modeling	✓	✓
Reviewing my peers was useful, as I understood how much time to spend on a question, and how to plan the solution		✓
I learnt from both my correct and incorrect solutions	✓	
The class activities helped me realize that every student has his unique way to solve the problem	✓	✓
The class activities encouraged me to look at the problem from different perspectives		✓

(autonomy), and they felt related to the task and to their peers, understanding that there were different ways to solve the same problem (relatedness).

5 Conclusions and Future Work

We conducted a teaching case of a UML modeling course, inspired by existing cognitive theories. 175 students, all in the second year of their studies, took a part at this mandatory course given in one of the leading academic colleges in Israel, at the department of software engineering. Our observations indicate that there are benefits of embedding active learning techniques in UML courses in sense of improving students' achievements and increasing the participation and motivation during the semester.

Our findings support the students need for being more active during the semester in order to improve their learning skills. It adds to the growing evidence of the benefits of promoting students' work during class - solving problems, creating, communicating, collaborating and being a critical thinker. However, we further realized that the workload students face needs to be more balanced throughout the semester and personal feedback following each assignment is in order. We plan to further embed these activities in this course, providing students with continuous feedback for their performance. We also plan to adapt this model to additional courses, which require abstract thinking.

References

1. Briggs, T.: Techniques for active learning in CS courses. J. Comput. Sci. Coll. **21**(2), 156–165 (2005)
2. Cooper, S., Dann, W., Pausch, R.: Teaching objects-first in introductory computer science. ACM SIGCSE Bull. **35**(1), 191–195 (2003)
3. Chrysafiadi, K., Virvou, M.: Student modeling approaches: a literature review for the last decade. Expert Syst. Appl. **40**(11), 4715–4729 (2013)
4. Hansen, K.M., Ratzer, A.V.: Tool support for collaborative teaching and learning of object-oriented modeling. ACM SIGCSE Bull. **34**(3), 146–150 (2002)

5. Jensen, J.L., Kummer, T.A., Godoy, P.D.D.M.: Improvements from a flipped classroom may simply be the fruits of active learning. CBE—Life Sci. Educ. **14**(1), ar5 (2015)
6. National Education Association - Alexandria, VA.: Preparing 21st Century Students for a Global Society: An Educator's Guide to the "Four Cs". National Education Association (2012)
7. Pilone, D., Pitman, N.: UML 2.0 in a Nutshell. O'Reilly Media Inc., Sebastopol (2005)
8. Ryan, R.M., Deci, E.L.: Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. Am. Psychol. **55**(1), 68 (2000)
9. Unkelos-Shpigel, N.: Peel the onion: use of collaborative and gamified tools to enhance software engineering education. In: Krogstie, J., Mouratidis, H., Su, J. (eds.) CAiSE 2016. LNBIP, vol. 249, pp. 122–128. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-39564-7_13