Experiences of a Software Engineering Course based on Interactive Learning

Stephan Krusche, Nadine von Frankenberg, Sami Afifi
Technische Universität München, Munich, Germany
krusche@in.tum.de, nadine.frankenberg@tum.de, afifi@mytum.de

Abstract

Learning to apply software engineering requires practical experience, and can not be taught through traditional theory-based lectures. Interactive learning is an approach that combines lectures and exercises into multiple iterations of theory, example, exercise, solution and reflection. It is based on active, computer based and experiential learning and on immediate feedback to improve the learning experience in large classes. It includes hands-on activities with the goal to increase students' motivation and engagement.

This paper describes an interactive learning course design that includes multiple choice quizzes and interactive tutorials as in-class exercises and a team project in which students apply their knowledge in a different setting. Based on this course design, we present a case study with 300 students in 2016. An evaluation shows that students are more engaged and motivated, if they practically apply and exercise the previously learned theory in the classroom. By providing students with both, theoretical foundations and practical exercises, their learning experience improves.

1 Introduction

Software engineering (SE) requires practical application of knowledge (Connolly et al., 2007; Shaffer, 2004), because it is an interactive and collaborative activity (Whitehead, 2007). In particular, project management in SEering is an activity that requires practical experience. The learning experience of students is low when educators disregard the practical relevance of SE and do not handle real problems in a course (Cunliffe, 2002). Interaction with students is limited if the learning activities focus on the educator in front of the classroom. Then, students' participation and motivation are low and the learning outcome decreases.

Educators can apply self-guided learning, personal responsibility, practical relevance and individualization to overcome this problem. Several pedagogic theories have been developed that include these elements: Problem-based learning teaches a subject through the experience of problem solving. Educators support, guide, and monitor this process (Boud and Feletti, 1998). Cooperative learning organizes classroom activities into social learning experiences: Students complete exercises in groups towards a common goal (Johnson et al., 1991). Computer based learning allows students to learn through computer-mediated activities (Garrison and Kanuka, 2004). Experiential learning is the process of learning from experience and reflecting about it (Kolb, 1984). Active learning promotes that students actively participate in the learning process (Bonwell and Eison, 1991). Instead of only passively listening, they are involved in exercises and engaged in solving problems.

We developed a course that includes a mix of these approaches and teaches SE concepts through **interactive learning** (Krusche et al., 2017). The course includes interactive tutorials and quizzes as activating in-class exercises where students immediately receive feedback to reflect about their performance. It also integrates team based exercises in which students apply the knowledge in a different situation to deepen their understanding and to increase their knowledge retention.

While the integration of multiple learning theories and exercise types increases the effort for educators, it can lower their stress and it can lead to higher satisfaction for educators and learners (Ben-Ari et al., 2003). We base our teaching methodology on a Chinese proverb: *"Tell me and I will forget. Show me and I will remember. Involve me and I will understand. Step back and I will act"* (Korthagen et al., 2001). It emphasizes that involving students into the learning process, activating them in the classroom, is the key for their understanding. Self-guided and problem-based learning let students take responsibility to solve a problem on their own, using concepts they learned before.

The paper is organized as follows: Section 2 describes active learning and the Revised Bloom's Taxonomy as foundations of the learning theories in our course. In Section 3, we present the course design that follows an interactive learning approach with an iterative process combining lectures and exercises into short cycles. Section 4 presents a case study about a large software engineering course with 300 students

in which we applied interactive learning. In Section 5, we present the findings of an evaluation of this case study. Section 6 discusses related work and Section 7 concludes the paper.

2 Foundations

Active learning is an educational approach to increase student involvement with the subject being taught. Instead of students acting as receivers of knowledge by passively listening to lectures, active learning puts the emphasis on developing student skills and engaging them in activities. Bonwell and Eison define active learning as "anything that involves students in doing things and thinking about the things they are doing" (Bonwell and Eison, 1991). The active learning approach draws from constructivist learning theories and can be summarized in four main premises (Brophy and Good, 1994):

1. Learners construct their own meanings
2. New learning builds on prior knowledge
3. Learning is enhanced by social interaction
4. Meaningful learning develops through "authentic" tasks

Grabinger and Dunlap emphasize that authentic contexts encourage students to take more responsibility and engage them in learning activities that promote high level thinking processes (Grabinger and Dunlap, 1995). In SE education, an authentic context would be a software project where students have to develop an application: they experience typical development workflows and tools such as software configuration management.

Students experience collaborative learning through learning communities that involve both peer students and instructors. Their learning progress is supported and assessed through realistic tasks such as planning and conducting a meeting. There is broad support for the benefits of active learning on knowledge transfer and student performance (Prince, 2004). Active learning has an improved learning outcome compared to more passive approaches (Michael, 2006).

Bonwell and Eison propose a set of activities that align with the principles of active learning (Bonwell and Eison, 1991). Examples are:

- **Think-Pair-Share:** Students think and discuss about a topic in pairs.
- **Simulation:** Classroom activities resemble real-life situations.
- **Working in group:** Collaborative or cooperative group work requires high involvement of students.
- **Case studies:** Practical examples encourage students to integrate knowledge from class with real-life.

While keeping active learning principles in mind, instructors can use the Revised Bloom's Taxonomy (RBT) to classify curricular objectives and exercises (Krathwohl, 2002). The RBT identifies six cognitive process categories (remember, understand, apply, analyze, evaluation, create) and four knowledge categories, ordered from concrete to abstract knowledge:

- **Factual:** Basic knowledge to acquaint with a discipline and to solve problems.
- **Conceptual:** Connection of basic knowledge in a larger context.
- **Procedural:** Methodology of knowledge application using skills, techniques, and methods.
- **Metacognitive:** Knowledge about the use of particular strategies for learning or problem solving.

The cognitive process dimension together with the knowledge dimension help to formulate learning objectives. Learning activities that require higher order cognitive processes and that lead to acquisition and construction of more abstract knowledge can be classified as active learning. Figure 1 shows the RBT matrix classifying more active and more passive learning approaches.

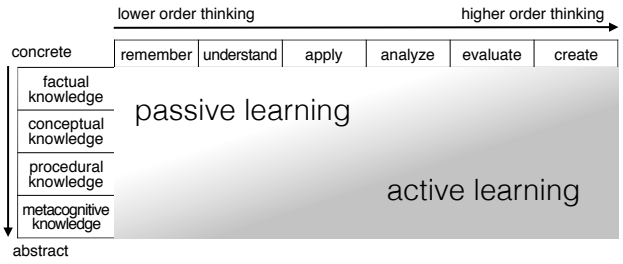

Figure 1: Classification of more active and more passive learning approaches in the matrix of the Revised Bloom's Taxonomy (adapted from (Krathwohl, 2002))

3 Course Design

With the aforementioned pedagogical foundations in mind, we designed a course to teach software project management by mixing lectures with engaging in-class and homework activities, such as hands-on tutorials, multiple choice quizzes, team exercises and team projects. To activate students, we put emphasis on the interactivity of the course.

3.1 Learning Objectives

The course has the following intended learning outcomes: Participants understand the key concepts of software project management. They learn and apply the basic techniques and methods of project organization that are used when complex software systems are developed such as task, issue and meeting management. The course focuses on agile models as preferred software lifecycle, in particular Scrum (Schwaber, 1995) and Kanban (Anderson, 2010).

Students communicate and collaborate in team projects, learn to estimate tasks and to schedule a

project. They learn how to model software life-cycles and how to write an agile contract. They design user interfaces, create prototypes and evaluate these using typical usability heuristics. Student apply software configuration management including change, branch, merge and review management using git (Chacon, 2009) and pull requests (Krusche et al., 2016). They apply build and release management by implementing typical continuous integration and continuous delivery workflows (Krusche and Alperowitz, 2014).

Students do not only get familiar with the theory of each topic, but also get practical experience. They get to know specific cases and learn how to use each concept in different settings.

3.2 Organization

The course is designed for large audiences with more than 100 students. One instructor teaches the course with the help of teaching assistants (TAs). Besides helping the students, the TAs also act as intermediaries between the students and the instructor. To maintain a high level of interactivity, informal communication channels encourage students to interact with other course participants, with their team members and with the instructor. A learning management system is used for formal information sharing.

During class, TAs monitor a question channel of a chat tool, and respond when necessary. This gives students the opportunity to clarify questions through informal communication. People respond and communicate more frequently when using an informal communication tool (Kraut et al., 1990), such as a chat application. During class, TAs can inform the instructor about issues that are of interest for other students. Then, the instructor can clarify issues and answer questions in front of all students. In addition, the instructor encourages students to ask questions in the lecture hall as well.

Lectures and exercises are combined into interactive classes to encourage students to attend. Students are expected to actively participate: they must bring their own laptop, tablet or smartphone and use it in class for computer based exercises. To motivate students to participate in these exercises, students can earn bonus points to improve their grade in the final exam. In addition, students can participate in a team project to apply the learned knowledge in another setting. This team project includes with five team members and is a simplified version of the team projects described by Bruegge and his colleagues (Bruegge et al., 2015): there is no real customer and students have less deliverables, but the applied process model is the same. While the team projects are not mandatory to the students, the instructor encourages them to take part, because students learn important communication and negotiation skills when it e.g. comes to task distribution and meeting management.

3.3 Interactive Learning

Figure 2 shows the iterative process of interactive learning. Each lecture has multiple iterations of the following five phases (Krusche et al., 2017):

1. **Theory:** The instructor introduces a new concept and describes the theory behind it. Students listen, try to understand it and ask questions.
2. **Example:** The instructor provides an example so that students can refer to a concrete situation.
3. **Exercise:** The instructor asks the students to apply the concept in a small exercise. The students submit their solution to the exercise.
4. **Solution:** The instructor provides a sample solution, explains it to the students and discusses exemplary student submissions to provide immediate feedback and guidance.
5. **Reflection:** The instructor facilitates a discussion about the theory and the exercise so that students reflect about the concept.

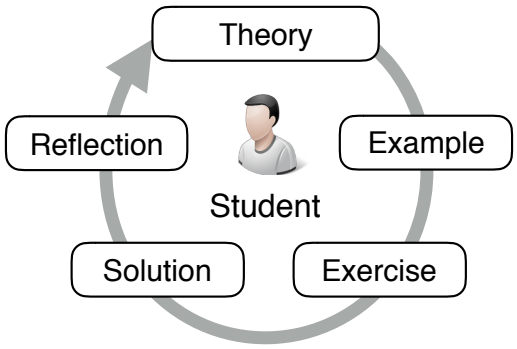

Figure 2: Iterative process of interactive learning performed multiple times during lectures (Krusche et al., 2017)

While the theory helps to build factual knowledge and conceptual knowledge, exercises can build procedural and metacognitive knowledge.

3.4 Exercises

TAs help in the conduction of the exercises: they walk through the classroom, answer questions and provide help in case problems occur or exercise instructions are unclear. The assessment of the submitted solutions can either be automated using tool support, or manually done by the TAs who review the submitted solutions and provide immediate feedback to the students. The degree of automation depends on the exercise type and the solution's format. The course includes individual exercises and team based exercises:

Individual exercises

E1 Quizzes with drag and drop questions or multiple choice questions (automatic evaluation through a quiz system).

E2 Interactive tutorials with step-by-step instructions (automation degree depends on the exercise).

In each class, theory is followed by short in-class quizzes, serving as self-assessment so that students can instantly check whether they understood the main concepts or not. Therefore, quizzes help to increase factual and conceptual knowledge by repeating and connecting the learned theory.

Interactive tutorials include detailed, step-by-step instructions so that even beginners are able to conduct the exercises. They are helpful to experience a concept for the first time. The instructor performs these tutorials live in class so that students can follow on their own laptops. He uploads the presentation slides with detailed screenshots before class, so that students can look up the steps of the exercise on the slides if they cannot follow in the given time.

He asks the students several times during a tutorial how many of them can still follow. If not enough students raise their hand, he waits and explains the current step again in more detail. If more than around 80 % raise their hand, the instructor continues. More experienced students are kept motivated with optional challenges. Interactive tutorials help to build procedural knowledge by involving the students into the methodology of knowledge application and by building skills, techniques and methods to solve particular tasks.

All tutorials are self-contained and do not depend on previous exercises. They are based on the same common problem statement provided in the beginning of the class, so students know the context and can follow exercises more easily. The instructor does not have to introduce a new problem statement in each class and saves time. If students miss a class, they can also catch up with the exercise at home.

Team based exercises

E3 Project teamwork that includes communication and collaboration aspects (automation degree depends on the exercise).

Team based exercises incorporate the concepts of peer learning and cooperative learning. They repeat the topic of individual exercises to deepen and retain the knowledge by applying the learned concepts in a different setting. Students transfer the previously learned knowledge to the concrete team situation and tailor the concepts. This facilitates self-guided learning and promotes the idea of self-organization, an important management aspect.

To create a context for their project, the teams choose a problem statement and a development environment in the beginning of the course. In team based exercises, students need to transfer the previously-learned factual, conceptual and procedural knowledge into a concrete situation. They need to adapt the learned skills, techniques and methods or find new ones to solve the problem collaboratively while taking responsibility because the instructor steps aside. This helps to build metacognitive knowledge.

4 Case Study

The following case study describes the university course "Software Engineering II: **P**roject **O**rganization and **M**anagement" (POM) that implements interactive learning. We evaluated the course in summer 2016, amongst others by means of a questionnaire that included free text fields in which the students stated their thoughts. The detailed evaluation is discussed in Section 5.

4.1 Course Format

The course had a heterogeneous distribution of 300 students with two groups standing out: around half of the students are bachelor students with major in information system who have to take this course in their studies. The other half are master students in computer science and take the course by their own choice. The challenge is to keep the lecture content easy enough for less experienced students, but also stimulating enough for more experienced ones.

The course took place in one semester over 13 weeks in the summer 2016, with a three hour time slot for lectures and exercises between 8:15 am and 11:30 am, including a 15min break. A single instructor taught the course with the help of 9 teaching assistants. Table 1 shows the schedule and the content of each lecture.

Week	Class content
1	Team Formation
2	Project Organization
3	Software Lifecycle Models
4	Agile Methods (Krusche et al., 2014)
5	Prototyping & Usability Management (Bruegge et al., 2012)
6	Proposal Management
7	Branch, Merge & Review Management (Krusche et al., 2016)
8	Contracting & Estimation
9	Continuous Integration
10	Continuous Delivery (Krusche and Alperowitz, 2014) Feedback Management (Krusche and Bruegge, 2014)
11	Risk and Demo Management
12	Global Project Management
13	Project Management Antipattern

Table 1: Overview of the course content

In large courses, students get easily distracted, may no longer pay attention to the lecture, or may engage in off-topic conversations with each other. Therefore, our main goal was to design and structure each class so that students are engaged and motivated using interactive learning.

As additional motivation, students were able to earn bonus points (BP) for participating in exercises. If they earned enough BPs, their grade in the final exam was improved accordingly. Students reported in a survey

that the bonus was a “strong motivation to be active in the course”. In the following, we illustrate the course concept in more detail.

4.2 Quizzes

During each class, multiple choice quizzes gave students the opportunity to revise the covered theory, and to earn BPs. To motivate students to attend class, we performed the quizzes dynamically during the class after certain lecture content was completed. Thus, only students present in class could participate in the quizzes. On average, 181 students participated in the quizzes per class.

We optimized the creation of quiz questions during the course using an iterative feedback approach to minimize misunderstandings and ambiguities. Two TAs created the quizzes based on the lecture content, then all TAs reviewed the questions to find errors and misunderstandings. On average, there were three quizzes per class. A quiz consisted of three questions with three to four answer choices each, an example of a question is shown in Figure 3.

Question: What are key characteristics of the Waterfall Model?

✓ 1. Progress is measured by the number of tasks that have been completed.
✓ 2. At the end of each activity, a verification step prevents the deletion or unwanted introduction of requirements.
X 3. The Waterfall Model allows the repetition of activities if requirements change unexpectedly.
X 4. Traditional managers do not like waterfall-based models, since fixed milestones are bad for progress measurement.

Figure 3: Example question with four answer choices

To prevent cheating during the quiz, students could see their results only after the quiz was closed. Questions and answer choices were randomly interchanged in the learning management system which included an automated quiz grading component that showed the overall quiz performance. As a result, there was no correction overhead.

Most students liked the concept of using quizzes directly after the lecture content, and stated in a survey that it was ”good to use quizzes to see right away what I learned”. The quiz performance of all participating students was available instantly to the instructor, and provided information on how well the students could follow the lecture. After each quiz, the instructor shortly reviewed and discussed questions and answers with the students. This offered the opportunity to repeat key points of the taught theories. Students also had the chance to give feedback about the quiz.

In the beginning of the course, some students reported “too little time to read and answer some questions”. We considered this when designing subsequent quizzes. Other students’ reported inconsistencies in the quizzes and helped to improve the questions. There was controversial feedback concerning the quizzes, as some students felt that they “put too much pressure” on them. Most students appreciated the quizzes as a direct control of their learning outcome. One student reported: “I like the quizzes a lot. They really help me to understand the topics faster and better”. Another one stated in the Slack[1] channel that was open during class: “quizzes wake me up better than coffee” as shown in Figure 4.

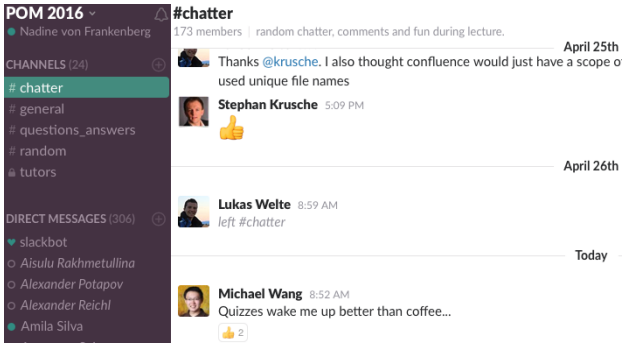

Figure 4: Slack channel with students’ comments during the lecture

4.3 Exercises and Homework

In the first class, students organized themselves into 51 teams, with 5 students per team, to participate in a team project. Each team chose one of three distinct problem statements, and had the task to implement a small mobile app until the end of the course, while applying project management methodologies. The students had to include at least one experienced and one unexperienced team member to give unexperienced students the opportunity to learn from more experienced ones, following the master apprentice approach (Collins et al., 1991). Experienced students could also benefit, since they deepened their knowledge, and were challenged by optional harder tasks.

The first team exercise was an icebreaker in the first class where all teams participated in a small competition, the marshmallow challenge (Wujec, 2010) in the lecture hall as part of the team formation class. The teams had 18 minutes to build the tallest structure using spaghetti sticks, tape, rope and a marshmallow. Figure 5 shows how the teams built the spaghetti towers in the classroom. After the exercise, each team had to measure its own tower according to specific rules and upload the picture to a shared space. The TAs evaluated the tower and the measurement and awarded the best team with small prices.

The instructor performed in-class exercises live on a computer shown on a projector. A second projector showed the corresponding lecture slides with detailed screenshots. The TAs walked through the lecture hall and helped if necessary. However, TAs did not explicitly tell students the solutions to the exercises, but

[1] Slack is a popular free team chat service that we used in class to improve the communication between instructor, TAs and students: `https://slack.com`.

Figure 5: Icebreaker with about 200 students in 51 teams in the lecture hall

rather pointed them into the right direction, so that they worked out the solutions themselves.

Immediate feedback was an important factor for the exercises. Students could review sample solutions on the projector, and were assisted by TAs. The exercise tools and the screenshots in the lecture slides provided additional feedback. If the student's screen looked identical to the screenshot on the slide after a task, or the tool reported a success message, the students knew that they performed the exercise correctly. We used several workflows and tools that are used in industry in order to demonstrate practical usage (Klepper et al., 2015). Most students found this approach helpful, and liked that the "exercises were practical and relevant".

To deepen the students' understanding, each lecture included a team project exercise to learn and apply management concepts, following a learning by doing approach. Students e.g. learned agile methods, and had to apply them throughout the team project. They performed self-organized meetings and documented their meetings, including a meeting-selfie of all participants to add a fun factor to the exercise. Another example is that they formed pairs in class, implemented a small feature, and then reviewed their partner's pull requests to understand the code review workflow (Krusche et al., 2016). Students then followed this pull request workflow when implementing the mobile app in the team projects. While the team projects were not mandatory, students could earn 50 % of the bonus points. Therefore, many students were motivated to participate in the team projects.

Multiple TAs reviewed the exercises that could not be assessed automatically. The students received feedback at the latest two weeks after the submission deadline. As Kothiyal and his colleagues point out, "prompt and descriptive feedback on their [the students] understanding" enables both, students and instructor, to "use this feedback to modify their learning and teaching respectively" (Kothiyal et al., 2013). Students found fast feedback motivating and helpful. For each exercise, students could see the current grading status and deadline so that they had an overview which exercises were due in the current week. They could see which TA graded their exercise to directly communicate with the TA to clarify questions.

5 Evaluation

This section describes the study design, the findings and the limitations of an evaluation of the course POM in summer 2016.

5.1 Study Design

We state the following hypotheses:

H1 Participation: Interactive in-class exercises increase the participation of students.

H2 Improved Learning: The mix of theory, quizzes and exercises in class leads to an improved learning experience for students.

We validate the hypotheses with a quantitative and a qualitative evaluation. In the quantitative evaluation, we measured the participation of students by counting how many students attended class and completed specific exercises.

In the qualitative evaluation, we investigated the students' improvements in topics that we applied in individual and team based exercises using an online survey. We asked about their opinion on the exercise concept. The questionnaire consisted of 14 questions, took about 10 minutes and was not mandatory for the students. It included questions about personal data, the participation in individual exercises, and application of techniques in the team project. We also wanted to know if students improved their skills in techniques and if they felt confident to apply techniques in their next team project. Finally, we asked in an open question how the course can be improved.

We conducted the survey in July 2016 and gave the students two weeks to complete it. We created personalized tokens and asked the 272 students, who completed the final exam of the course, to participate in the anonymous survey. The open source survey tool LimeSurvey[2] guarantees that the answers are anonymous by strictly separating token and answer tables in the database. We received 190 responses, which corresponds to a response rate of 70 %.

5.2 Findings

The quantitative evaluation shows that more students participated in POM in summer 2016 than in a previous instance of the course without interactive learning in summer 2014 or in other courses of the same faculty. Figure 6 shows that the number of participants per class in 2016 was around 80 % in the beginning and around 60 % in the end of the course, although the class started early at 8:15 am in the morning. Figure 7 shows that in the same course in 2014, the attendance rate steadily decreases to less than 20 % until the end

[2] http://www.limesurvey.org

of the course. On average, 71 % of all participants of the course completed the team exercises, whereas 72 % completed the individual exercises. From these numbers, we have first anecdotal evidence that H1 is supported: interactive in-class exercises increase the participation of students.

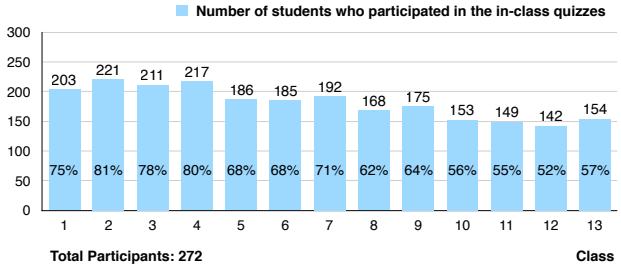

Figure 6: Number of participants per class in POM in summer 2016 **with** interactive learning

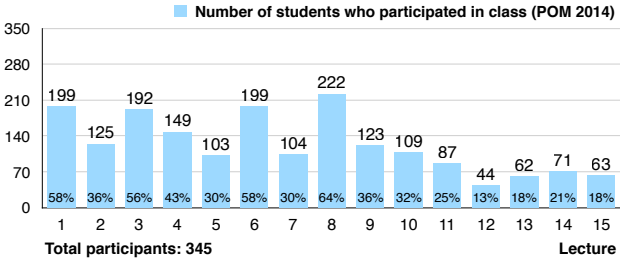

Figure 7: Number of participants per lecture in POM in summer 2014 **without** interactive learning

The qualitative evaluation showed that on average 80 % of the students, who participated in an individual and team exercise, agree (or strongly agree) that they improved their knowledge and that they are confident to apply the knowledge in their next team project. Table 2 shows results of the qualitative evaluation, i.e. whether students agreed to given statements in the qualitative evaluation. 80 % of the students agreed that the use of interactive learning increased their learning success (S1), and 71 % agreed that it improved their understanding of the theory during class (S2).

In-class exercises motivated 76 % of the students to attend the lecture (S3) and quizzes motivated 54 % of the students to listen to the lecturer (S4). Interactive tutorials helped 65 % of the students to learn new concepts and also 65 % were able to deepen their knowledge in team exercises. These answers can be considered as anecdotal evidence that H2 is supported: the mix of theory, quizzes and exercises in class leads to an improved learning experience of the students.

Formulating multiple choice quiz questions and their respective answer possibilities unambiguously proved to be challenging, especially when asking question that are beyond simple definitions. At the beginning of the course, we varied the number of questions per class, the number of answer choices per question, and the time per question. After the quizzes, we presented the correct solutions and discussed them quickly with the students. In the first half of the course the average number of quizzes per lecture session was higher: on average, we had five quizzes with 3 questions each, leading to 15 questions per class.

In an intermediate evaluation, we found that the number of quizzes per class was too high, so we reduced it to three quizzes and nine questions on average per class. The available time per questions was between 60 and 120 seconds depending on its difficulty and length. Additionally, we evaluated each quiz regarding the students' response rate. This helped us to extract weak and misleading factors of the questions and answers, e.g. when many students selected a wrong answer choice due to misinterpretation.

In total, many students reported that this course was their favorite course in the semester and that they wish that more courses would be rich in variety and activation during class.

5.3 Limitations

A limitation of the qualitative evaluation is that personal opinions of students might not reflect the real situation, because they could be subjective. Beginners cannot estimate objectively about their real improvement, and the confidence to apply a concept does not necessarily mean that the student is in fact able to apply it.

Other positive effects of the course, such as the open atmosphere towards feedback, might have a positive influence on the evaluation result. Only if students like interactive exercises, this does not necessarily mean that their skills improve. To alleviate these threats, we additionally evaluated the participation in the lectures and exercises quantitatively in a more objective manner.

6 Related Work

Active learning techniques applied in computer science show an increase in students' learning, engagement, and overall performance. A popular approach is Think-Pair-Share (TPS), where students first work on a problem individually, then in small groups and finally with the whole class.

Kothiyal and his colleagues describe a setting that uses TPS in a large level-1 programming course (Kothiyal et al., 2013). The course included lectures and programming labs. The lectures had two TPS activities, where students first worked on questions individually, and then with a subsequent task in pairs, while an instructor could be asked for help. Finally, class-wide discussions were facilitated concerning the former tasks. The study reports an average of 83 % student engagement for TPS-based courses. This approach shows some parallels to our course setup, since we introduced individual and team exercises, similar

	Statement	Strong Agree	Agree	Neutral	Disagree	Strong Disagree
S1	The mix of theory, quizzes and exercises in class contributed to my learning success	36 %	44 %	13 %	4 %	3 %
S2	The mix of theory, quizzes and exercises improved my understanding of the theory during class	33 %	38 %	16 %	10 %	3 %
S3	In-class exercises motivated me to attend the lecture	35 %	41 %	14 %	6 %	4 %
S4	Quizzes motivated me during class to actively listen to the lecturer	19 %	34 %	23 %	15 %	9 %
S5	Interactive tutorials were particularly helpful to understand concepts that I did not know before	24 %	41 %	24 %	9 %	2 %
S6	Team exercises helped me to apply the concept in a different setting to deepen my knowledge and understanding	16 %	49 %	26 %	6 %	3 %

Table 2: Evaluation results with Likert scale typed responses whether students agree to given statements

to the think and pair phases. The share phase is also present, as students could join discussions with the instructor.

Kurtz and his colleagues describe an active learning approach using microlabs (Kurtz et al., 2014). Students perform 5-10min activities during lectures, either individually or in groups, and submit their answers to an automated grading system, using tablets as delivery mechanism. Students receive constructive feedback, and can revise their answers. The study concludes that microlabs can increase the students' learning gains. This approach can be compared with the in-class exercises we performed. Students had a pre-defined time limit for the exercises and submitted their solutions to an automated grading system, or to an online documentation tool. The key point is, that both approaches are used during lectures.

Campbell and his colleagues describe a flipped classroom approach with video lectures, labs and assignments (Campbell et al., 2014). Similar to our approach, quizzes were used and contributed to the course grade. However, the authors do not give credit for in-class exercises, and report a low lecture attendance rate. Our course design includes homework assignments as team exercises, as well as immediate feedback for in-class exercises to keep students motivated.

Heckman reports, there is "a large increase in student engagement" for the use of in-class laboratories (Heckman, 2015). His approach is similar to our in-class exercises, but not used in large classes.

7 Conclusion

In this paper we described our experiences with an interactive learning course in project management in software engineering. Interactive learning is based on active, computer based and experiential learning: the instructor combines lectures and exercises into multiple iterations of theory, example, exercise, solution and feedback. The course includes multiple choice quizzes and interactive tutorials as in-class exercises and an additional team project where students apply their knowledge in a different setting. This mix supports different knowledge dimensions.

We applied and evaluated interactive learning in a large course with 300 students. We found that interactive learning increases the participation of students. Our findings show that students are more engaged and motivated, if they practically apply and exercise the previously learned theory in class. By providing students with theoretical foundations and practical exercises, their learning experience improves.

References

Anderson, D. (2010). *Kanban: Successful Evolutionary Change for Your Technology Business*. Blue Hole Press.

Ben-Ari, R., Krole, R., and Har-Even, D. (2003). Differential effects of simple frontal versus complex teaching strategy on teachers' stress, burnout, and satisfaction. *International Journal of Stress Management*.

Bonwell, C. and Eison, J. (1991). *Active Learning: Creating Excitement in the Classroom.* ASHE-ERIC Higher Education Reports.

Boud, D. and Feletti, G. (1998). *The challenge of problem-based learning*. Psychology Press.

Brophy, J. and Good, T. (1994). *Looking in Classrooms*. HarperCollins College Publishers.

Bruegge, B., Krusche, S., and Alperowitz, L. (2015). Software engineering project courses with industrial clients. *ACM Transactions on Computing Education*.

Bruegge, B., Krusche, S., and Wagner, M. (2012). Teaching Tornado: from communication models to releases. In *Proceedings of the 8th edition of the Educators' Symposium*, pages 5–12. ACM.

Campbell, J., Horton, D., Craig, M., and Gries, P. (2014). Evaluating an inverted cs1. In *Proceedings of the 45th technical symposium on Computer science education*, pages 307–312. ACM.

Chacon, S. (2009). *Pro git*. Apress.

Collins, A., Brown, J., and Holum, A. (1991). Cognitive apprenticeship: Making thinking visible. *American educator*.

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

Cunliffe, A. (2002). Reflexive dialogical practice in management learning. *Management learning*, 33(1):35–61.

Garrison, R. and Kanuka, H. (2004). Blended learning: Uncovering its transformative potential in higher education. *The internet and higher education*.

Grabinger, R. and Dunlap, J. (1995). Rich environments for active learning: A definition. *Research in Learning Technology*, 3(2):5–34.

Heckman, S. (2015). An empirical study of in-class laboratories on student learning of linear data structures. In *Proceedings of the 11th annual conference on International Computing Education Research*, pages 217–225. ACM.

Johnson, D. et al. (1991). *Cooperative Learning: Increasing College Faculty Instructional Productivity. ASHE-ERIC Higher Education Report*. ERIC.

Klepper, S., Krusche, S., Peters, S., Bruegge, B., and Alperowitz, L. (2015). Introducing continuous delivery of mobile apps in a corporate environment: A case study. In *Proceedings of the 2nd International Workshop on Rapid Continuous Software Engineering*, pages 5–11. IEEE/ACM.

Kolb, D. (1984). *Experiential learning: Experience as the source of learning and development*. Prentice Hall.

Korthagen, F., Kessels, J., Koster, B., Lagerwerf, B., and Wubbels, T. (2001). *Linking practice and theory: The pedagogy of realistic teacher education*. Routledge.

Kothiyal, A., Majumdar, R., Murthy, S., and Iyer, S. (2013). Effect of think-pair-share in a large cs1 class: 83% sustained engagement. In *Proceedings of the 9th annual conference on International computing education research*, pages 137–144. ACM.

Krathwohl, D. (2002). A revision of bloom's taxonomy: An overview. *Theory into Practice*, 41(4):212–218.

Kraut, R., Fish, R., Root, R., and Chalfonte, B. (1990). Informal communication in organizations: Form, function, and technology. In *Human reactions to technology: Claremont symposium on applied social psychology*, pages 145–199. Citeseer.

Krusche, S. and Alperowitz, L. (2014). Introduction of Continuous Delivery in Multi-Customer Project Courses. In *Proceedings of the 36th International Conference on Software Engineering*, pages 335–343. IEEE.

Krusche, S., Alperowitz, L., Bruegge, B., and Wagner, M. (2014). Rugby: An agile process model based on continuous delivery. In *Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering*, pages 42–50. ACM.

Krusche, S., Berisha, M., and Bruegge, B. (2016). Teaching Code Review Management using Branch Based Workflows. In *Companion Proceedings of the 38th International Conference on Software Engineering*. IEEE.

Krusche, S. and Bruegge, B. (2014). User feedback in mobile development. In *Proceedings of the 2nd International Workshop on Mobile Development Lifecycle*, pages 25–26. ACM.

Krusche, S., Seitz, A., Börstler, J., and Bruegge, B. (2017). Interactive learning: Increasing student participation through shorter exercise cycles. In *Proceedings of the 19th Australasian Computing Education Conference*. ACM.

Kurtz, B., Fenwick, J., Tashakkori, R., Esmail, A., and Tate, S. (2014). Active learning during lecture using tablets. In *Proceedings of the 45th technical symposium on computer science education*, pages 121–126. ACM.

Michael, J. (2006). Where's the evidence that active learning works? *Advances in Physiology Education*, 30(4):159–167.

Prince, M. (2004). Does active learning work? a review of the research. *Journal of Engineering Education*, 93(4):223–231.

Schwaber, K. (1995). Scrum development process. In *Proceedings of the OOPSLA Workshop on Business Object Design and Information*.

Shaffer, D. (2004). Pedagogical praxis: The professions as models for postindustrial education. *Teachers College Record*, 106(7):1401–1421.

Whitehead, J. (2007). Collaboration in software engineering: A roadmap. *FOSE*, 7(2007):214–225.

Wujec, T. (2010). The Marshmallow Challenge - TED Talk. Retrieved January 08, 2016 from `http://marshmallowchallenge.com`.