2016 IEEE 29th International Conference on Software Engineering Education and Training

Project-based Learning with Examples from Industry in University Courses

An Experience Report from an Undergraduate Requirements Engineering Course

Marian Daun, Andrea Salmon,
Thorsten Weyer, Klaus Pohl
paluno – The Ruhr Institute for Software Technology
University of Duisburg-Essen
45127 Essen, Germany
{marian.daun, thorsten.weyer, klaus.pohl}
@paluno.uni-due.de

Bastian Tenbergen
Department of Computer Science
State University of New York
Oswego, USA
bastian.tenbergen@oswego.edu

***Abstract*— A significant challenge within university education, especially with regard to the teaching of highly theoretical topics like requirements engineering, is to maintain students' interest and motivation whilst addressing the core concepts that will enable students to work in industry upon graduation. It has long been established that experience-based learning can aid in both these feats: On the one hand, providing students with industrial case examples rather than "dry" academic assignments can increase student interest and motivation. On the other hand, a case example-centric classroom approach can yield a rich learning environment which fosters collaboration, communication, and self-directed exploration of the instructed principles. In previous work, we have reported on our experience in changing a graduate requirements engineering course towards using case examples based on real industry projects. As more and more curricula change in advance of project-based teaching paradigms, this paper discusses results from the long-term application of such a course design in a graduate setting. In addition, this paper reports our findings from the replication in an undergraduate requirements engineering course indicating that project-based learning techniques foster different teaching goals in graduate and undergraduate settings.**

Keywords—project-based learning; problem-based learning; industry-oriented teaching; undergraduate students; industrial case example; case example

I. INTRODUCTION

Education in software engineering at the university level has two major aims: On the one hand, software engineering programs are aiming at producing academically skilled graduates, who are familiar with theoretical concepts, principles, and techniques. In principle, these graduates ought to be able to conduct research in the field, if they chose to pursue an academic career. This implies a comprehensive theoretical education focusing and rehearsing the fundamental underpinnings of software engineering. On the other hand, higher education in software engineering seeks to produce graduates, which can work in challenging industry projects, where they are asked to produce pragmatic solutions, drawing upon their theoretical education. However, from an industry perspective, the largely theoretical university level software engineering education is not appropriate to produce industry-ready graduates (cf. [1, 2]). Recent graduates often lack an understanding of the complexity of industry projects [3, 4, 5] and seek the one, definitive correct solution rather than exploring solution alternatives. Furthermore, passive instruction may lead to a lack of social skills necessary to work in teams, which is a core capability graduates must possess for real-world development projects [6]. This results in the need for additional on-site training of graduates until their academic knowledge can be applied in industrial settings [6].

In recent years, university education has successively moved away from traditional, theory-driven instruction, where exams are administered at the end of the semester [1, 7, 8], towards a more object-centric [9] and project-oriented approach [10, 11], where students are encouraged to explore solutions by themselves. The benefit in doing so lies in the fact that traditionally, poor retention in theory-driven instruction (see, e.g., [1]) is combatted by more exciting individual knowledge discovery and problem solving (see, e.g., [12, 13]).

In previous work [14], we have reported on findings from the introduction of interactive and industry-oriented teaching techniques to a graduate requirements engineering course. Our experience has shown that moving from a less theory-driven approach to a project-oriented approach had a profoundly positive impact on students' insight into industry challenges, their self-reported motivation, and ultimately student performance on final exams.

In this paper, we report on our experiences from the application of the project-based learning approach, which makes use of examples from industry in an undergraduate requirements engineering course. In addition, we will outline experiences gained after four years of application of the industry-oriented teaching approach. While the long-term application of the approach in a graduate setting confirms our findings from previous years, application in an undergraduate course indicates discrepancies between graduate and undergraduate teaching. In particular, we will discuss

2377-570X/16 $31.00 © 2016 IEEE
DOI 10.1109/CSEET.2016.15

weaknesses resulting from undergraduate students' preoccupation with quantifiable grades, hence suggesting that the project-based approach mainly holds value for industry-readiness as opposed to improving academic achievements.

In the following, Section II introduces related work with regard to problem-based learning, project-based learning, and industry orientation in university education. Subsequently, Section III will briefly summarize the approach under investigation and previously reported findings from a graduate requirements engineering course, which we have previously reported in [14]. Section IV discusses major findings from the introduction of the teaching approach under investigation in an undergraduate requirements engineering course. Section V discusses the new findings before Section VI concludes this paper.

II. Related Work

In literature, shortcomings of passive instruction are already broadly discussed. In particular, it is argued, that a lack of student involvement negatively impacts the students' learning abilities [7, 8]. Furthermore, short-term rewards (i.e. passing an exam) are commonly accomplished through rote memorization [12, 15] and may lead to poor long-term knowledge retention [1, 7]. As prominent counter-measures, problem- and project-based learning are widely suggested.

A. Problem-based learning

Problem-based learning has been suggested [7] as a possible remedy, as it requires students to completely grasp a problem domain and then derive a suitable solution for some challenge therein. Yet, unless applied in a wide learning environment (e.g., in coordination with other courses offered in the same major), such a "mile-deep and inch-wide" approach may impair students' industry-readiness after graduation (cf. [1]), as students may become experts in some problem within the problem domain, but not in the problem domain at large. In addition, problem-based learning requires a certain degree of self-discipline in the sense that students ought to be able to completely immerse themselves in an assigned topic on their own [16]. In paradigms where groups of students are assigned to individual problems and tasked with presenting their findings to the class, the class's overall learning success depends on each group's success in understanding the problem domain, solving some task, and preparing their findings in a pedagogically beneficial way. While this is generally accepted as highly beneficial for students, the cognitive load on students might impair learning efficiency (i.e. the "guidance-fading effect", see [17]). Furthermore, it is often perceived as taxing for instructors to ensure high course benefit and fair evaluation for all students [18]. Furthermore, while industry-readiness might be fostered for individual student groups with regard to the assigned topic, for the remainder of the class, this instruction paradigm might not lead to an increased ability to solve industrial challenges with other, non-assigned problems.

B. Project-based learning

Project-based learning (e.g., [10], [11]) has been proposed as an approach to foster creativity, student enthusiasm (e.g., [12], [15]), and familiarity with problem domains at large. Students work on case examples in a self-directed manner by making use of provided material and progress is tracked based on accomplished milestones. Project-based learning has striking advantages with regard to students' problem solving abilities [13], albeit it may be difficult to instruct complex theoretical relationships in a sound fashion [19]. Especially in technical domains, such as mathematics or computer science, it may be difficult to instruct a variety of largely independent but closely related concepts and topics (e.g., "automotive engine control units" and "modeling continuous behavior using SysML activity diagrams") in a fashion where all concepts and topics are equally honored in an overarching project (e.g., [20], [21]). Hence, in many computer science curricula, courses often make use of many smaller projects, rather than one overarching one. Yet, in this case, these topics may often be quite academic in nature and may not adequately resemble industry challenges (e.g., [7], [22]).

C. Industry Orientation

Industry-oriented higher education (e.g., [7], [22]) features case example-oriented instruction much like project-based learning. In contrast to project-based learning, however, the focus is not as much on creativity, but on problem-solving based on excerpts from real life industry projects. Different approaches to similar problems are discussed in plenum in order to foster learning from others' experiences and to improve communication skills within the project domain. Involvement of real stakeholders in particular is used to allow for a realistic learning experience (e.g., [23], [24]). However, achieving industry orientation is traditionally difficult, as there is little immediate reward for industry practitioners to collaborate in instruction (see, e.g., [25]), as their time is typically quite limited. In addition, there might be clearance issues regarding the dissemination of intellectual property and there is no guarantee for successful outcome or immediate benefit for the company. One way to counteract this challenge is to have instructors act as stakeholders [26], but this requires a high sensitivity and experience of the instructors regarding their ability to foresee student challenges, maintain the acted role throughout the semester, and carefully guide the knowledge discovery process in such a way that the students achieve teaching goals and perform satisfactorily. Moreover, this also requires a large amount of in-depth industry knowledge, which unless the instructor has an industry background, may not be attainable. Furthermore, the instructor will continuously have to switch between both rolls (i.e. stakeholder and teacher). Hence, there is a threat that students will take statements given by the stakeholder falsely as instruction or solution given by the teacher.

D. Combining these Paradigms

It becomes clear that project-orientation, especially with industry involvement, seems to have a clear positive impact on students' problem solving skills and on student motivation [27]. However, all of these instructional paradigms have in common that they require some level of preparation, let it be academically prepared problems, class or student projects, or industry cooperation of some kind. The instructors are required to produce lecture material, academic examples, or project

milestones (i.e. "objects" in the sense of the Oswego method, see [9]) that make up a valuable course curriculum and guide the knowledge discovery process and learning experience in a motivating, yet industry-ready way. In our teaching approach, we aim to combine these learning paradigms by focusing on industrially relevant challenges, allowing for a self-directed knowledge discovery process, and by fostering the exploration of solution alternatives rather than sample solutions. This is done by making use of industrial case examples. Their application in our teaching approach is outlined in the next section.

III. Approach Under Investigation

As outlined in Section I, we have previously discussed our teaching approach and presented qualitative and quantitative findings from its application in [14]. In this paper, we present findings from applying the teaching approach in an undergraduate requirements engineering course (see Section IV). To do so, this section will briefly summarize the concrete approach under investigation. In addition, we will briefly discuss former experiences from the application to a graduate-level requirements engineering course, which were confirmed in the long-term application of the approach. This will later on allow for comparison between undergraduate application presented in Section IV with findings from the graduate course.

A. Applying Industrial Case Examples in Requirements Engineering Eduacation

The proposed teaching approach focusses on the application of real industrial case examples in university teaching in combination with project-orientation. The approach relies on well-known teaching techniques, e.g., lecture-style instruction paired with seminar-style guided project exploration. Most significantly, our teaching approach makes use of realistic case examples of systems that have been built (in a similar fashion) by our industrial partners. These case examples have been taken from an industrial research project and are focused such that they represent typical and contemporary industrial software engineering and requirements engineering challenges while at the same time removing as many company-specific and domain-specific issues as possible. On the one hand, this was intended to protect our partners' intellectual property while at the same time, allows for the case examples to be worked on without the need for expert knowledge in, e.g., automotive or avionic engineering. We explicitly chose safety-critical domains as the impact and importance of proper requirements engineering becomes obvious with "layperson" knowledge. In addition, these domains rely on strict engineering processes with strong emphasis on requirements engineering and systems in these domains are customarily not developed in an unstructured or "lightweight" fashion. We extensively discussed current as well as general industrial problems with our industry partners and ensured appropriateness of case examples. In addition, this leads to an in-depth understanding of industrial problems and challenges necessary for instructing the material.

Our teaching goals were to improve students' industry-orientation, method competence, and problem-solving skills: The students shall gain awareness of industrially relevant problems and, at the same time, foster an in-depth understanding of requirements engineering theory. In addition, students shall also be able to use the academic concepts to solve real-world problems on their own. To do so, we relied on a combination of several key elements:

- **Theory-centric meetings.** The courses consisted of a series of traditional lectures, as lecture-based teaching is the common teaching approach for advanced theoretical concepts. However, rather than solely lecturing, we provided lecture material beforehand, and, in the fashion of flipped classroom [27], encouraged students to interject any question at any point.
- **Biweekly, voluntary assignment sheets.** We designed biweekly assignment sheets consisting of easy to moderately difficult theoretical problems. Assignments were discussed in dedicated, weekly tutorial sessions, but were never collected nor graded. Students were, however, encouraged to use these assignments sheets to study up on the discussed material by themselves, and were provided with detailed feedback if they chose to submit their assignment sheets. Assignment sheets served as a rehearsing mechanism for the final exam at the end of the course, which determined the grade, and were aimed to strengthen method competence for immediate application in the case studies.
- **Industrial case examples and case studies.** Students were grouped into teams and each team was provided with a single industrial case example. These case examples were worked on over the entire semester, as students were asked to complete a series of milestones that obligatorily had to be submitted for review and critique (thereby forming a semester-long case study). The milestones represented major development phases and comprised techniques and concepts covered during several lectures. To help students find their own solutions for problems, they were allowed to revise or resubmit milestones as often as necessary. To give insights in the multiplicity of problems and possible solutions, each team had to present their milestone accomplishments in the tutorial sessions.

We neither provided sample solutions for the assignment sheets nor for the case studies in order to encourage students to come up with individual solutions on their own. Instead, assignment sheets and project milestones were discussed in class, where solutions to individual problems were developed in the tutorial sessions in plenum. The tutorial instructors guided the discussions, provided further suggestions, insights, and assisted when teams got stuck. For more detailed account on our teaching approach, please refer to [14].

B. Past Experiences from the Graduate Requirements Engineering Course

In this section, we briefly summarize our previous findings from applying this teaching approach in a graduate requirements engineering course (see [14]) and provide some additional results from continued application in the following years. In Section IV, we will then provide new experience from

a replication in an undergraduate setting, and compare our findings to the graduate-level findings.

For the last years, about fifty students participated in the course each year. Students can be considered mostly first and second year Masters' level students enrolled in degree programs for either software engineering or business information systems at the faculty of business administration and economics at the University of Duisburg-Essen, Germany. In both degree programs, the course is offered for elective credit. Attendance in neither the lectures nor the tutorials is compulsory. The course aims at teaching advanced methods and techniques pertaining to the documentation, and analysis of textual and model-based requirements. The course comprises 15 weeks, each week offering one lecture and one tutorial session, each of which lasts at least 90 minutes. The following material is covered: goal- and scenario modeling, essential systems analysis, requirements validation and management (e.g., prioritization, negotiation). The grade is determined by a final exam. However, in order to be admitted for the exam, successful completion of the tutorial session is mandatory.

Until 2011, we applied a rather rigid, lecture and assignment sheet-based instruction style, where for every unit of theory instructed in the lecture, an assignment sheet would follow, which was graded and discussed in class. Successful completion of all assignment sheets (i.e. scoring at least 60% of points over all assignment sheets) was a hard admission criterion to be allowed to take the final exam. Starting in 2012, we applied the teaching paradigm from Section II. Experiences include the following:

- **Stronger focus on theory in classroom discussions.** The classroom discussions moved away from technical questions regarding the assignment sheets (i.e. wording of assignments or usage of diagram syntax) towards more content-centric topics such as diagram semantics, notational alternatives that convey the same meaning, or possible misinterpretations by specific stakeholders.
- **More active student involvement.** Discussions frequently evolved beyond the scope of the case studies, investigating related topics. Students showed a higher interest in how "things would be in a real-life development project". We furthermore observed that student participation increased: While before only about 20% of students would actively involve themselves, almost the entire class would share ideas, comments, agreement, and disagreement in the new setting. In fact, we observed an overall and considerable increase of student enrollment in the course in subsequent years.
- **High degree of voluntary work.** Although we expected that at best only a handful of students would work on the voluntary assignment sheets, we were surprised to see that almost all students asked questions pertaining thereto in almost all meetings, tutorial and lecture. Furthermore, students made extensive use of the ability to present preliminary solutions of future milestones to the entire class in order to receive peer feedback and instructor comments. Moreover, we observed an increased intrinsic effort in student solutions.

In addition to these qualitative experiences, we were able to deduce quantitative results from mandatory course evaluations. Course evaluations are conducted among all students of a course on the authority of the university's teaching quality assurance program. Evaluations are administered by the course instructors and consist of about 20 Likert-scale questions. Quantitative results include the following:

- **Improved student satisfaction with the course.** From 2012 to 2013, students evaluated the course better than in previous years. In particular, students' self-reported learning experience was rated higher than before. However, presumably due to higher enrollment numbers and associated decrease in instructor-to-student ratio, a decrease in student satisfaction with the course was noticeable in 2013.
- **Higher satisfaction as opposed to other department courses.** While the overall teaching quality at the department where the graduate requirements engineering course is taught is very high, students rated their satisfaction with the course higher than the department average. In particular, self-guided learning and theory comprehension showed an improvement over the remaining department courses, which can be attributed to the teaching paradigm. Even in 2013, where the course was evaluated as least successful, the course was rated better than department average.
- **Improving exam scores.** Figure 1 shows the student exam results in percentage of maximum attainable points. Light grey boxes indicate the years prior to adoption of the new teaching paradigm, while dark grey boxes indicate years after adoption. For reasons regarding the completeness of our remarks we added the results from the years 2014 and 2015 to the figure. These years have not been investigated in our past report, but as can be seen they confirm our previous findings. The figure shows that the exam results improved dramatically over the years. It must be noted that the improvement already started before introduction of the new setup. Yet, in general, the exam results after introduction of the new teaching approach can be considered better than in previous years. Furthermore, there is no considerable change between the years after introduction. However, a slight increase in average grades from 2012 on can be detected.

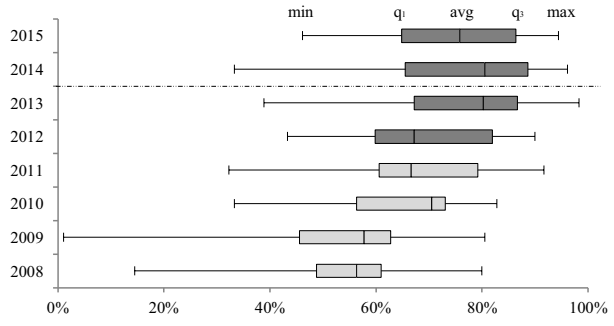

Fig. 1. Development of the Student Exam Results in the Graduate Course

As discussed in [14], we cannot attribute the increasing trend in exam grades solely to our teaching approach. However, it can clearly be seen that our teaching approach has no detrimental effect on the student's ability to solve problems in requirements engineering. Together with the consistently positive qualitative experience regarding student involvement and individual learning success, we are confident in the positive benefit of our teaching approach and intend to maintain this style of instruction in graduate settings.

IV. New Findings from the Replication within an Undergraduate Requirements Engineering Course

Driven by the strong positive impact of our teaching approach, we aimed to implement it in an undergraduate requirements engineering course as well. In this section, we briefly outline the differences of the undergraduate equivalent to the graduate course, present qualitative experience as well as quantitative results, and compare our findings to those from Section III.B.

A. Structure of the Course

The teaching approach was implemented in an undergraduate requirements engineering course at the University of Duisburg-Essen in Germany in Fall/Winter 2014. The format of the undergraduate requirements engineering course was equivalent to that of the graduate course (see Section III and [14]): The course also consisted of one lecture and one tutorial session per week, each of which lasted about 90 minutes. The lecture taught theoretical concepts while the tutorial applied these concepts. Successful completion of the tutorial was necessary to be admitted the grade-determining final exam. Topics covered in the course comprised fundamentals of requirements engineering, textual requirements documentation, introduction to requirements modeling, and modeling requirements in the static-structural, functional as well as the behavioral perspective of system description, see [28]).

Before the adoption of the new teaching approach, successful completion of the course comprised sufficient points in graded assignment sheets and a tool examination intended to introduce common requirements engineering tools. After the adoption in 2014 when the tutorial was switched to the new teaching approach, successful completion entailed sufficient involvement and completion success in the case studies. The case examples were the same ones from the graduate course, but the milestones were adapted to account for the topics taught in the lecture. The assignment sheets from the previous years were still provided, and, if voluntarily submitted, assignment sheets were graded and returned. Yet, in contrast to previous years, the scores on the assignment sheets didn't impact the final grade nor admission to the exam. Students were mainly third and fourth year undergraduates enrolled in Bachelor programs in software engineering, business information systems, or less often business administration. About 140 to 180 students enroll in the undergraduate requirements engineering course on average. In consequence, rather than two tutorial sections of 25 students (like in the graduate course), the undergraduate course consisted of a total of six tutorial sections with up to 30 students each. In each tutorial section, the student population was separated into four to six teams with five to six team members. Every team was given a choice what case example to work on, which means that some case examples were worked on by multiple teams in the same tutorial section.

B. Qualitative Experiences and Results from the Undergraduate Course

To a large degree, experiences and results from the undergraduate course matched those from the graduate course. In particular, the observed knowledge discovery process was quite similar. However, some differences remain. These qualitative experiences can be differentiated into two categories. Firstly, while some experiences were basically similar to those in the graduate setting, it seemed that the effects were more pronounced. For example, students were far more confused at the beginning of the semester. Secondly, we observed challenges in the undergraduate course that we did not encounter in the graduate course at all. For example, undergraduate students seemed to be overburdened with the teamwork and the necessary initiative on their own. In the following, we discuss these categories of findings in more detail.

1) Issues of higher impact in undergraduate education than in graduate education

Primarily, application of the teaching approach in an undergraduate setting suggests three major differences from applying the teaching approach in a graduate setting. These issues mainly became apparent from student interaction. They predominantly deal with student-voiced concerns regarding their evaluation and their final grade. While these observations can also be made in the graduate course, in the graduate course only a negligible number of students were affected, while in the undergraduate course, the following three issues were a recurring theme.

Issue 1.1: Students expressed concerns regarding exam admission. At beginning of the semester, students were particularly concerned with how to achieve exam admission. As outlined in Section III.A and IV.A, successful completion of the tutorial session (i.e. successful involvement and completion of case study milestones) was mandatory to be admitted to the grade-determining final exam. In other department courses there is often a clear quantifiable criterion. This was also valid for the course before introduction of the new teaching approach: Exam admission was achieved by successful completion of the tutorials, i.e. by scoring a certain number of points on graded assignment sheets. In the new approach, successful completion in tutorial sessions was achieved by active involvement in the case study, where active involvement was supervised and enforced by the instructor.

Students required constant assurance in the beginning weeks of the course that obtaining admission was really "that easy". The absence of strict regulations how to succeed and how to fail seemed to make students uncomfortable. As a result, many students inquired often on how well they did in terms of coverage and correctness of the case study milestones. Since in requirements engineering, there rarely are absolute truths (with regard to customer satisfaction), we relied on evaluating the student groups' respective learning success and

knowledge exploration process rather than enforcing strict pass/fail criteria. In consequence, no overall best practice sample solution was given. To ensure fair treatment of students we allowed handing in revised versions of the milestones throughout the entire semester and gave detailed feedback on parts which could be improved. In fact, we did not require student solutions to be formally and semantically perfect, but asked students to look deeply into the problem and explore different solutions. Hence, all students regularly handing in milestones and revisions eventually received exam admission.

In practice, the concerns of the great majority of students could be diminished after the first milestone was handed in and feedback was provided by the instructor. Yet, a handful of students remained concerned throughout the semester.

In contrast to the undergraduate course, the same concerns were raised by students in the graduate course. However, graduate students seemed to appreciate the guided knowledge discovery process over strict quantifiable criteria more, while undergraduate students to a large degree were afraid of this treatment.

Issue 1.2: Student expressed concerns regarding the grading in the final exam. Frequently and throughout the semester, students expressed concerns regarding the exam itself. Students often inquired about the types of questions and assignments on the exam and asked what would be an acceptable solution in the exam. In a few cases, students mentioned informally after class to the instructor that "it is difficult to guess what answer would appease [the instructor]".

In fact, during class interactions, it became noticeable that many students' approach to taking exams is to adjust to the types of questions asked by certain instructors and finding a strategy to memorize and recite the instructors' most favorable statements and writing down pure sample solutions. The new teaching paradigm interferes dramatically with such an approach, as it focuses on exploration on the problem domain rather than regurgitating information. Considering that students consistently uttered their appreciation of the close relation to industry challenges and expressed the value of the knowledge discovery process fostered by our teaching approach for their life after university, we find the implications of students' preoccupation with how to pass an exam alarming. Our teaching approach contrasts with students' experiences from other courses. In other courses, students seem to be primed to memorize concepts and sample solutions, as written exams often only test recollective memory, thereby leaving little space for individual solutions and innovative student ideas. Our teaching approach focuses mainly on knowledge discovery: A wrong answer could be just as acceptable as a right answer, as long as the reasoning is sound. In the final exam, we applied the same principle, to which students were able to adapt only with difficulty.

This was particularly the case for the undergraduate course. Students frequently doubted that the case studies were an adequate preparation for the final exam. The instructors had to point out often that assignment sheets were not only discussed in class, but could also be voluntarily submitted for feedback on method competence, thereby allowing for extra preparation. This offer was only used once: Only a single student submitted some assignment sheets for feedback, close to the final exam. In the graduate requirements engineering course, students were naturally also concerned with the way exam questions were asked and graded. Yet, contrastingly, the graduate students seem more willing to believe the instructor that the course is an adequate preparation for exam and displayed a higher level of voluntary work in additional preparation. Furthermore, graduate students seem to be more familiar with the idea that memorization on its own will not necessarily lead to good final grades. Nevertheless, from our discussions with students, we assume that even in graduate education the majority of courses still relies on memorization of the instructor's book, script, or general ideas.

Issue 1.3: Students appreciated embedded system case examples. Albeit students appreciate the insights given into embedded industry and embedded systems development at the end of the course, the use of embedded system case examples came quite unexpected to many of the participating students. In other courses throughout the department, the focus is strictly limited to either programming languages or to applications in the business area (e.g., information system development, e-business, etc.). Hence, embedded systems are not a concept students are commonly familiar with and at a first glance were often considered boring. Yet, in the end, the case examples seemed to allow students to gain insights from another point of view, and once familiar with the differences to more traditional types of systems (e.g., information systems), students seemed to enjoy the problem domain of, e.g., how software can help make a car move. In our opinion, such case examples leverage everyday knowledge.

While in the graduate course, students often commented the innovative aspect of the case examples (e.g., an autonomously flying drone), graduate students had seen embedded systems case examples before and mostly commented on the specific system (e.g., a drone vs. a heart rate sensor). In the undergraduate course, students predominantly commented on the type of system (e.g., an embedded system like automotive adaptive cruise control vs. an information system like a library information system). In contrast to the previous years, we felt that the domain of embedded systems was, albeit initially unfamiliar to undergraduate students, served as a richer domain to illustrate and practice concepts of requirements engineering in particular and software engineering at large. As consequence, we consider courses outside the students' subject area as a very valuable part of software engineering curricula and encourage mandatory courses. Currently, most students do not elect such offered courses but rely on soft skill courses such as languages and negotiation skills.

2) Issues specific to Undergraduate Education

One clear observation taken from the undergraduate course was that there was a considerably higher spread between those students actively involved in the classroom and those students merely being "physically present" without applying themselves. This was a general trend across all tutorial sessions in the undergraduate course and gave rise to the following findings.

Issue 2.1: Teamwork did not come naturally to all students. Student motivation in undergraduate courses is

traditionally difficult to achieve and keep up. The new teaching approach appeared to give those students who would typically participate in classroom activities, an opportunity to "hide" in the back row and only participate when prompted. Albeit the majority of teams self-assimilated well and was able to find an agreeable level of cooperation, over the course of the semester, some teams would occasionally complain that some team member wasn't contributing to the team's progress. In these cases, the student was approached by the instructor and encouraged to participate. Upon further complaints, the team typically focused on their own work and excluded the uncooperative student from their milestone.

This is a new finding in the undergraduate course that we did not experience in the graduate course. In the graduate course, all students applied themselves with enthusiasm and contributed to the team's progress. In the undergraduate course, this was true for the vast majority of students, but about half the teams complained about at least one student who would not engage in the case study. However, it was to note that the prevalence of disengaged students was somewhat lower than in the previous years: In the years prior to the introduction of the new teaching approach, a similar higher percentage of students would not make an effort to contribute to class discussions, would not hand in mandatory assignment sheets in a timely manner, or would simply stop attending the course. Since introduction of the new teaching approach, we made fewer such observations, which is in line with the general increase in student motivation that we consistently observed since applying the teaching approach in the graduate course.

Issue 2.2: Necessary countermeasures against declining attendance and participation. In a few cases, we noticed that entire teams seemed to consist of students who would not take the case study seriously. These teams would eventually stop attending the tutorial session. In these cases, milestone presentation often showed very poor performances by the respective teams, leading to rather harsh remarks by other students attending the tutorial session. As soon as such a trend became obvious for a specific team, the instructor of the respective tutorial session intervened and exhorted the team to increase their effort. As result, many teams improved their engagement with their material and presence in tutorial session to a satisfying level. In some cases, such intervention resulted in the entire team to withdraw the course. Only in very few cases, continued encouragement and intervention was needed for the respective team to maintain acceptable performance and attendance.

These findings are in line with experiences from previous years, where regardless of the mode of instruction, a considerable effort must be exerted by the instructors to maintain involvement of some students. We experienced this to be quite taxing on both the instructors as well as the students. In one case, a well-performing highly engaged team complained that in comparison, the less engaged team would receive more attention and "got away with worse solutions." In this case, we responded by enforcing that, as mentioned above, our teaching approach focuses on individual knowledge discovery rather than optimal solutions. We emphasized that the better teams would engage with milestones, the more likely it would be for them to score highly in the final exam. This was accepted by most students, and in fact held true: Team members of low engaging teams on average performed at best in the 50th percentile in the final exam, while highly engaged team members were more likely to receive top scores.

Summing up, we consider the course setup to be largely self-regulating, but recognized a higher amount of organizational challenges the instructor was forced to deal with than in the graduate course.

Issue 2.3: Stronger focus on knowledge discovery. Considering that undergraduate education to some degree requires more instructor guidance than graduate education, we were surprised to see that the majority of actively involved students seemed to constantly inquire about other, related topics. While in the previous years, the undergraduate requirements engineering course was akin to a school curriculum, where students were given a topic, an accompanying assignment sheet, and grades for the assignment sheet solutions, the case example based instruction sparked interest for many students. Many students were able to draw conclusions and understand dependencies on related topics, which was not common in the years before. Albeit in contrast the graduate course, the focus was not on alternative solutions, the focus in the undergraduate course seemed to be on how topics fit together.

For example, a recurring theme was how natural language requirements elicited during early milestones needed to be prioritized for modeling in later milestones. In the first milestone, students were asked, among other things, to come up with a list of at least 20 natural language requirements (some of which functional requirements, some quality requirements, and some constraints), but students were free to add as many more requirements as necessary to describe their case example system as completely as possible. While at first, student submissions typically entailed a rather unstructured collection of exactly 20 requirements, in subsequent milestones, teams decided to focus on some aspects of their system more than on other aspects. For example, albeit a subsequent milestone asked for a complete behavioral model of the system, many teams decided to only focus on a single function, but then described the behavior of that function in greater detail than their requirements from the first milestone depicted. In these cases, teams elected to revise their first milestones in order to ensure that the system's requirements specification remains consistent. Considering the additional effort, instructors accepted such deviations from the strict milestone fulfillment. In fact, we encouraged such explorations of the problem domain as we felt that it would allow students to gain a more detailed understanding of how requirements' quality properties are impacted by their elicitation and documentation.

In general, students seemed to gain a broader understanding of the requirements engineering techniques, as compared to previous years. Students seemed in general motivated by the project milestones and often mentioned their appreciation of being able to explore certain topics in a structured, but self-guided fashion.

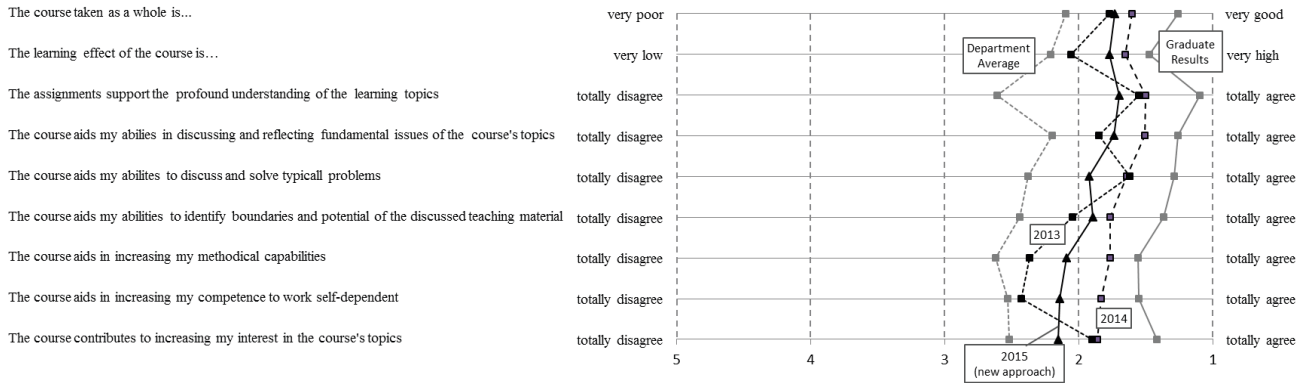

Fig. 2. Development of the Results from the Annual Student Evaluation of the Undergraduate Course in Comparison with Base Lines from the Graduate Course and the Department's Average

C. *Quantitative Experiences and Results from the Undergraduate Course*

Like for the graduate course, we investigated the results of the annual student evaluation regarding the students' self-reported learning experience and their exam scores. Figure 2 shows the results from students' self-reported learning experience of the course when the new teaching approach was adopted in 2015 (solid black line) and compares these results with the student evaluations from the years prior to adoption in 2013 and 2014 (i.e. the dashed black lines), the department average (dashed grey line), and the graduate results from [14] (solid grey line). As can be seen, students' evaluations of the undergraduate course in 2015 cannot be considered better or worse than the results of the previous years. In 2013 already, there was some fluctuation in how students evaluate the course across different evaluation aspects, which evened out to some degree in 2014. In 2015, the students' perceived learning benefit is essentially in between the self-reported benefit from the previous years. This shows that in contrast to the impact in the graduate course (see solid grey line as well as [14]), undergraduate students do not recognize the new teaching approach as dramatically more beneficial for them. On the other hand, the self-reported benefit seems to be on par with the way students perceived the course in the previous years. Nevertheless, it can be seen that in comparison, the course benefit has consistently been rated as more beneficial in 2013 and 2014 than the department average, which was also the case for 2015, when the new teaching approach was implemented.

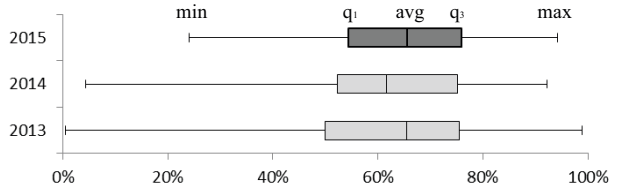

Fig. 3. Development of the Student Exam Results in the Undergraduate Course

In Figure 3, the results of the final exam of the two years prior to adoption of the new teaching approach are depicted (light grey boxes) and compared to the final exam scores from the 2015 course, where the new teaching approach was implemented. As can be seen, in contrast to the graduate exam results depicted in Figure 1, undergraduate exam results did not improve considerably after implementing the new teaching approach. Albeit a slight increase in average results from 2014 to 2015 can be observed, the average results in 2015 was on par with the average results from 2013. This suggests that the new teaching approach does not improve exam results in an undergraduate setting (albeit in a graduate setting, such an improvement became quite noticeable). However, it can be seen from Figure 3, in contrast to 2014 and 2013 in particular, the range of scores is between 25% and 90% in 2015 and therefore narrower than in the previous years, where the whole spectrum was covered. However, this is a negligible effect which we are hesitant to attribute to the new teaching approach, as other factors beyond the scope of investigation could have been at play.

V. Discussion

We applied the teaching approach accommodating the strong focus on case examples in order to increase students' readiness to solve industrial challenges upon graduation and to increase student motivation. After having applied the case example-centric approach in the graduate requirements engineering course over several years now, we are satisfied that placing such a strong focus on individual knowledge discovery in case study milestones improves student communication and leads to lively discussions beyond the milestone assignments. This impact was also pertinent in the undergraduate setting.

Moreover, in the graduate course, we consistently recorded a strong positive impact on the students' self-reported learning experience and increasing exam scores. In the undergraduate course, the same positive change in classroom discussions was noticeable, although quantitatively, the course was not evaluated better in terms of learning experience, nor was there an impact on grades. Nevertheless, considering there was also

no negative impact on self-reported learning experience nor a negative impact on grades, we are confident that our teaching approach is a well-suited means to instruct requirements engineering on both the graduate and undergraduate level and positively influences students' ability to study in a self-directed fashion by means of real-world examples highlighting industry challenges.

However, a direct comparison of graduate and undergraduate performance in a setting that merely differed in the instructed topics, our experience shows two key findings: (1) students are increasingly preoccupied with exams and passing grades than with knowledge discovery and (2) especially undergraduates show a low level of resilience and discipline needed to succeed in self-directed learning environments. To some degree, differences in self-directedness, procrastination, and motivation between undergraduate and graduate students are to be expected, and widely documented in various studies (e.g., [29]). Nevertheless, as a side-effect of applying our teaching approach, we observed an increased tendency for students to be more concerned with what is expected of them, rather than what they can learn. We find these developments alarming. Higher education is placing emphasis on quantifying students' achievements and graduating students that show aptitude to take exams, but that are not necessarily capable of satisfying industry needs. Moreover, low student resilience and low student discipline as experienced in our team-oriented case studies might be indicative of a larger issue beyond the initial scope of investigation of this research: A declining level of student resilience at large. In popular science, [30], it has been recently recognized that student resilience is on the decline. We believe to have witnessed such development as well. Over the years of applying our teaching approach in the graduate course, students continuously grew more concerned with grades and quantifiable goals and less concerned with the process of knowledge discovery and critical thinking. On the one hand, this might be due to school-like curricula, which prime students towards completing assignments and achieve degree requirements, rather than fostering a well-rounded education. Although graduate students were able to adapt to the new, unfamiliar teaching approach (and, consequently, how they would be evaluated), the increased trend of students to be more concerned with quantifiable scores remains more pertinent from year to year. This effect was also particularly noticeable in the undergraduate course. Albeit a certain level of concern for quantifiable scores was expected, we were surprised by how much students were almost exclusively preoccupied with satisfying the evaluative criteria. In most but not all cases, this preoccupation vanished over the course of the semester, leading to a subjectively inverted Gaussian distribution of those students who seemingly did extraordinarily well in their case study and those that struggled to cope with the casual milestone requirements.

Nevertheless, for almost every undergraduate as well as graduate team, the milestones were completed in a timely fashion and with a high degree of detail and professional application of the instructed theoretical concepts. We therefore believe that our experiences are compelling reasons for broad adoption of project-based learning approaches using real industrial case examples. Especially in graduate education, students appreciate the knowledge discovery process and their ability to solve industrial challenges. For undergraduate education, our teaching approach could foster student discipline and resilience. All of these benefits are important aspects which could increase student's ability to deal with real-world challenges. However, it must be pointed out that the impact of our teaching approach on real-world readiness nor on resilience was not investigated in this study and can only be inferred.

VI. Conclusion and Future Work

In this paper, we have discussed our problem-based and industry-oriented teaching approach to instruct requirements engineering courses. The teaching approach focuses on individual knowledge discovery through realistic industrial case examples in order to rehearse theoretical concepts. We have summarized previous and additional findings from continued application in a graduate level requirements engineering course. We furthermore reported in detail on qualitative experiences and quantitative findings from a replication in an undergraduate setting. Our results show increased student motivation, indications of a more self-directed knowledge discovery process, and higher levels of engagement in students. However, we also found increased student concerns regarding their evaluations, and a more taxing effort on the part of the instructors to supervise and guide student progress in the undergraduate setting. In summary, we regard our teaching approach as successful in instructing highly theoretical concepts in a fashion that allows students to gain insights into real industrial challenges while at the same time fostering motivation and personal growth.

It must be noted, however, that the findings reported in this paper are subjective and may be impacted by researcher bias. Of course, we had an interest in seeing our approach succeed in both the graduate as well as the undergraduate requirements engineering course. To alleviate the issue of researcher bias, we reported on quantitative data in Section IV.C. Furthermore, our results might not be generalizable to all instances of software engineering education at the university level, as curricula and course structures may be vastly different. Especially within the German university system, a lot of emphasis is traditionally placed on assessing student performance through monolithic final exams and classroom times are traditionally limited to one or two lectures a week and occasionally lecture-accompanying seminars. Since our teaching approach was developed in such an organizational setting, it remains an open question, how the teaching approach transfers to different organizational settings. Therefore, we have detailed the complete course design and invite others to replicate our teaching approach. We furthermore plan to implement our teaching approach in a northeastern US university to investigate the teaching approach in more detail. In addition, we believe that our experiences will also hold in other areas of software engineering, since the case examples had no particular focus on requirements engineering. Hence, if the milestones and student assignments were changed according to the respective needs of other software engineering courses, this approach may hold for those courses as well. We

therefore also plan to expand our teaching approach to other courses as well.

Our findings indicate that our teaching approach is more successful in a graduate setting than in an undergraduate setting. This suggests that a certain level of maturity, background knowledge, and resilience is necessary in order to attain all benefits of the case example based approach. However, our findings also show that undergraduate students benefit from the self-guided knowledge discovery process in that they seemed to draw similarities and dependencies on related but orthogonal topics more easily than in a strict school-like curriculum. We have furthermore observed the alarming fact that students seem to be preoccupied with quantifiable grading metrics rather than what they can learn in a course and that some students rely on school-like prompting rather than self-motivated engagement. Popular science has recognized such trends at large (e.g., [30]). Since university education must lately not only ensure that sound theory is taught in a manner that makes graduates ready for the industry, but also increase student's ability to cope with challenges in the real world (see e.g., [31]), we believe that our teaching approach can help in this manner.

ACKNOWLEDGEMENTS

This research was partly funded by the *German Federal Ministry of Education and Research* under grant no. 01IS12005C and grant no. 01IS15058C. Thanks to our industrial partners for their support. In particular, we thank Jens Höfflinger and John MacGregor (Bosch), Frank Houdek (Daimler), and Stefan Beck as well as Arnaud Boyer (Airbus).

REFERENCES

[1] K. Garg and V. Varma, "Case Studies: The Potential Teaching Instruments for Software Engineering Education," in *Proc. 5th Int. Conf. Soft. Quality*, 2005, pp. 279-284.

[2] C. Wohlin and B. Regnell, "Achieving Industrial Relevance in Software Engineering Education," in *Proc. 12th IEEE Conf. Soft. Eng. Educ. & Training*, 1999, pp. 16-25.

[3] K. Beckman, N. Coulter, S. Khajenoori, and N. Mead, "Collaborations: Closing the Industry-Academic Gap," in *IEEE Software*, vol. 14, no. 6, 1997, pp. 49-57.

[4] M. Shaw, "Software Engineering for the 21st Century: A basis for rethinking the curriculum," Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-ISRI-05-108, 2005.

[5] E. Oh and A. v. d. Hoek, "Towards Game-Based Simulation as a Method of Teaching Software Engineering," in *Proc. 32nd Ann. Frontiers in Educ. Conf.*, 2002, pp. 2-13.

[6] A. LaSalle. "An inverted computing curriculum: preparing graduates to build quality systems," in *Proc. 27th Ann. Frontiers in Educ. Conf.*, 1997, pp. 280-284.

[7] K. Garg and V. Varma, "A Study to the Effectiveness of Case Study approach in Software Engineering Education," in *Proc. 20th IEEE Conf. Soft. Eng. Educ. & Training*, 2007, pp. 309-316.

[8] P. Senge, "The Fifth Discipline: The Art and Practice of the Learning Organization," in *Performance+Instruction*, vol. 30, no. 5, 1991, p.37.

[9] P. Rillero, "The enlightenment revolution: A historical study of positive change through science teacher education," in *J. Science Teacher Educ.*, vol. 4, no. 2, 1993, pp. 37-43.

[10] J. Burge, "Application and Appreciation: Changing Course Structure to Change Student Attitudes," in *Proc. 22nd IEEE Conf. Soft. Eng. Educ. & Training,* 2009, pp. 45-52.

[11] Q. Li and B. Boehm, "Making Winner for Both Education and Research: Verification and Validation Process Improvement Practice in a Software Engineering Course," in *Proc. 24th IEEE Conf. Soft. Eng. Educ. & Training*, 2011, pp. 304-313.

[12] F. Meawad, "The Virtual Agile Enterprise: Making the Most of a Software Engineering Course," in *Proc. 24th IEEE Conf. Soft. Eng. Educ. & Training*, 2011, pp. 324-332.

[13] A. Sivan, R. Wong Leung, L. Gow, and D. Kember, "Towards more active forms of teaching and learning in hospitality studies", in *Int. J. of Hospitality Management*, vol. 10, no. 4, 1991, pp. 369-379.

[14] M. Daun, A. Salmon, B. Tenbergen, T. Weyer, and K. Pohl, "Industrial Case Studies in Graduate Requirements Engineering Courses: The Impact on Student Motivation," in *Proc. 27th IEEE Conf. Soft. Eng. Educ. & Training*, 2014, pp. 3-12.

[15] M. Alfonso and A. Botía, "An Iterative and Agile Process Model for Teaching Software Engineering," in *Proc. 18th IEEE Conf. Soft. Eng. Educ. & Training,* 2005, pp. 9-16.

[16] C.E. Hmelo-Silver, "Problem-Based Learning: What and How Do Students Learn?," in *Educational Psychology Review*, vol. 16, no. 3, Springer US 2004, pp. 235-266.

[17] J. Sweller, "The worked example effect and human cognition," in *Learning and Instruction*, vol. 16, no. 2, 2006, pp. 165-169.

[18] S. A. Azer, "Introducing a problem-based learning program: 12 tips for success," in *Medical Teacher*, vol. 33, no. 10, 2011, pp 808-813.

[19] C. C. Bonwell and J. A. Eison, "Active Learning: Creating Excitement in the Classroom," The George Washington University, Washington, DC, ASHE-ERIC Higher Educ. Rep. No. 1, 1991.

[20] P. C. Blumenfeld *et al.*, "Motivating project-based learning: sustaining the doing, supporting the learning," in *Educational Psychologist*, vol. 26, 1991, pp. 369-398.

[21] S. Boss and J. Krauss, *Reinventing project-based learning: Your field guide to real-world projects in the digital age*, 1st ed. Eugene, OR, ISTE, 2008.

[22] I. Richardson and Y. Delaney, "Problem Based Learning in the Software Engineering Classroom," in *Proc. 22nd IEEE Conf. Soft. Eng. Educ. & Training*, 2009, pp. 174-181.

[23] B. Penzenstadler, M. Mahaux, and P. Heymans, "University meets industry: Calling in real stakeholders," in *Proc. 26th IEEE Conf. Soft. Eng. Educ. & Training*, 2013, pp. 1-10.

[24] B. Brügge and M. Gluchow, "Towards production ready software in project courses with real clients," in *Proc 1st Int. Workshop Soft. Eng. Education Based on Real-World Experiences*, 2012, pp. 5-8.

[25] G. Gabrysiak, H. Giese, and A. Seibel, "Why Should I help You Teach Requirements Engineering?," in *6th Int. Workshop Requirements Eng. Educ. and Training*, 2011, pp. 9-13.

[26] D. Zowghi and S. Paryani, "Teaching requirements engineering through role playing: Lessons Learnt," in *11th IEEE Int. Requirements Eng. Conf.*, 2003, pp. 233-241.

[27] L. Abeysekera and P. Dawson (2015). "Motivation and cognitive load in the flipped classroom: definition, rationale and a call for research," in *Higher Educ. Research & Develop.*, vol. 34, no.1, 2015, pp. 1-14.

[28] M. Daun, B. Tenbergen. and T. Weyer, "Requirements Viewpoint," in *Model-Based Eng. of Embedded Systems*, Heidelberg, Germany, Springer, 2012, pp. 51-68.

[29] A. R. Artino Jr. and J. M. Stephens, "Academic motivation and self-regulation: A comparative analysis of undergraduate and graduate students learning online," in *The Internet and Higher Educ.*, vol. 12, no. 3-4, 2009, pp. 143-151.

[30] P. Gray. (2015, Sep. 22). Declining Student Resilience: A Serious Problem for Colleges [Online]. Available: https://goo.gl/fcnMNn

[31] E. Piper. (2015). This is not a Day Care. It's a University! [Online]. Available: https://goo.gl/QY9VSO