Combining education, industry, and empirical studies in Software Engineering – an experience report

Josip Maras, Ljiljana Šerić, Maja Štula
University of Split
Rudera Boškovića 32
Split, Croatia
{josip.maras}, {ljiljana}, {kiki}@fesb.hr

Nenad Ukić
Ericsson Nikola Tesla
Poljička cesta 39
Split, Croatia
nenad.ukic@ericsson.com

ABSTRACT

Software industry is one of the most pervasive industries today and has a great impact on our day-to-day lives. At the same time, the quality of software systems is directly related to the quality of software engineers – it is the responsibility of software engineering educators to provide students with relevant skills needed for the development of high-quality software systems. Amongst the cornerstones of developing high-quality software systems are industry-relevant experience and the ability to quantify certain aspects of the software development process. In this paper, we describe our experience of performing an empirical study on students, during a software engineering course, on an industry-relevant topic taught by an industry expert – the understandability of models in model-driven engineering.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics

General Terms

Experimentation

Keywords

Software engineering education, empirical studies, model-driven engineering.

1. INTRODUCTION

Software industry is one of the most knowledge-intensive industries, where the quality of the workforce has a determining influence on the quality of the products. Since the quality of the workforce directly depends on the quality of the software engineering education, it is the responsibility of software engineering educators to provide students with relevant skills needed in the software industry. One way to achieve this is to incorporate industry experience by inviting industrial professionals as guest educators to present their complementary view to the issues of developing real-world, high-quality software systems [8].

IEEE defines software engineering as an application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software [1]. As can be seen from the definition, an important aspect of software engineering is the ability to quantify certain aspects of software development. This is done through software metrics [5]. Usually, in software engineering education students gain theoretical knowledge about the usefulness of software metrics, as well as practical experience with some low-level metrics (such as lines of code, cyclomatic complexity, depth of inheritance, etc.). Quantifying higher-level metrics, such as maintainability or understandability, is usually discussed only in theory.

One way of increasing our understanding of the software development process is by using empirical studies [9]. This is usually done by obtaining evidence-based data with which we can make informed assessments that can be applied to software engineering processes, the choice of tools, techniques, etc. There are different types of empirical studies that can be performed in software engineering: surveys, case-studies, quasi-experiments, experiments, etc. Ideally, these studies should be performed in an industrial setting. Unfortunately, performing industrial experiments is expensive, so it is often useful to carry out pilot studies on students in an academic setting [2].

In this paper, we give a report about an empirical study on the understandability of software models. The study was performed as a part of a software engineering course organized in cooperation with industry. In this way, in a single course, we have provided three different aspects: *i)* educational – students have gained experience in model-driven engineering (a relatively novel software engineering discipline, used by the industry, but still not wide spread in software engineering education) and software metrics; *ii)* research – even though there are some issues with using students as subjects in empirical research, the obtained results are still useful as guidelines, even from the research perspective; and *iii)* industrial: the students were thought by an industry expert in model-driven engineering. A survey, which student participants have answered, has shown that students consider that using industry experts provides additional insights in software engineering education and that more courses

ECSAW '15, September 07 - 11, 2015, Dubrovnik/Cavtat, Croatia
ACM 978-1-4503-3393-1/15/09 ...$15.00.
http://dx.doi.org/10.1145/2797433.2797472.

should include some form of industry cooperation. In addition, students believe that participating in an empirical research experiment has deepen their knowledge of model-driven engineering, and are willing to participate in other, similar experiments.

The paper is organized as follows: Section 2 presents the background on model-driven engineering, empirical studies, and related work. Section 3 presents the experiment that we have performed, while Section 4 gives an overview of different lessons learned. Finally, Section 5 concludes the paper.

2. BACKGROUND

In this section, we give a short introduction to model-driven engineering, empirical studies, and related work.

2.1 Model-Driven Engineering

A core feature of software is its intangibility, which often leads to difficulties with achieving system understanding. In order to tackle this problem, software engineers often visualize the internal organization of software with models, most often UML models. Traditionally, UML models are only used during design-time, and in certain cases they are used to generate the skeleton source-code of the application, which is a starting point for the implementation phase. In later phases of development, all modifications are usually done directly in source code. This means that after a certain time, the initial model is usually deprecated and used only for documentation purposes. However, model-driven engineering (MDE) and executable models changed this paradigm by making models the main software artifact. The term "executable models" denotes that models contain all semantic information necessary for program execution, and can thus be executed. Since executable, models can be used for specification of complete application functionality which can be verified by executing the model-based tests in the model interpreter. Such work-flow considers models as the main intellectual property of the software development process. While MDE is starting to be adopted by industry, it has not yet widely spread to software engineering education.

2.2 Empirical Studies

Often in software engineering, we have to make choices about the methodology, the process, or the tools that will be used for the development of software systems. In order to make informed decisions we have to find a way to quantitatively or qualitatively measure certain aspects of software development. This is often done by performing empirical studies such as: surveys, case-studies, quasi-experiments, experiments, etc. Empirical studies, when done correctly, can provide significant observational data that can help us understand the reality of software development. In order to provide useful data, empirical studies have to be performed in realistic, representative environment, ideally in an industrial setting. However, preparing and conducting empirical studies requires significant time and effort, especially from industry representatives participating in the study. The alternative is to use students as study participants in an academic setting. This situation is also not without its problems, as students are often significantly less experienced than industry professionals. However, some research has found that there are only minor differences between the performance of students and professionals [7], while others [10] have found that the difference depends on the student level – there are significant differences when comparing undergraduate and graduate students, but they are small when comparing graduate students and industrial professionals. Regardless of these differences, it is often advisable that pilot studies are prepared and carried out on students in an academic setting [2]. These pilot studies can provide interesting data that can be used as a basis for the improvement of a full empirical study in an industrial setting.

2.3 Related Work

There are numerous empirical studies in software engineering that use students as study participants, e.g. [4, 6, 3], but in terms of similarity, ours is most closely related to [2], in which Carver et al. discuss different perspectives and provide insight about using students as experiment participants. Compared to them, in this paper, we present a case study experiment in which we analyze the effects of combining education, research, and industry in software engineering education.

3. EXPERIMENT SETUP

The primary goal of the experiment was to perform a pilot study on the understandability of models in model-driven engineering, more specifically, to study the influence of complexity distribution on the overall system understanding.

Subjects – The subjects of the experiment were third-year, undergraduate computer science students enrolled in the "Systems Analysis and Design" course at the University of Split, Croatia, during the spring semester of 2015. This was their first exposure to MDE, but it is important to note that they already had courses – "Software Engineering" and "Object-Oriented programming", in which they gained experience with the general ideas and techniques behind software modelling.

Objects – The objects of the study were three versions of a medium-sized Calculator application, all implemented with MDE (Figure 1), by a MDE expert. In all three cases, the functionality of the application is the same – they expose the same external interfaces that pass the same test suite. The only difference between these applications is in their internal implementation. The internal implementation of application A was based on simple C-style functional/structural decomposition. The internal implementation of application B was based on well-known object-oriented abstractions, while the internal implementation of application C has additionally introduced state-machines that model object life-cycles.

3.1 Subject Preparations

The experiment was performed as an integral part of the "Systems Analysis and Design" course. The students were given one three-hour lecture on the theoretical foundations of MDE, by an industry expert from the Ericsson Corporation. The lecture was followed by three two-hour lab assignments, performed in the following three weeks. The lab assignments were designed to reinforce the theoretical concepts, as well as to provide the students with practical experience of creating and understanding models and the act of MDE. They also gained practical experience working with an open-source MDE tool – Bridgepoint[1].

[1]https://xtuml.org/download/

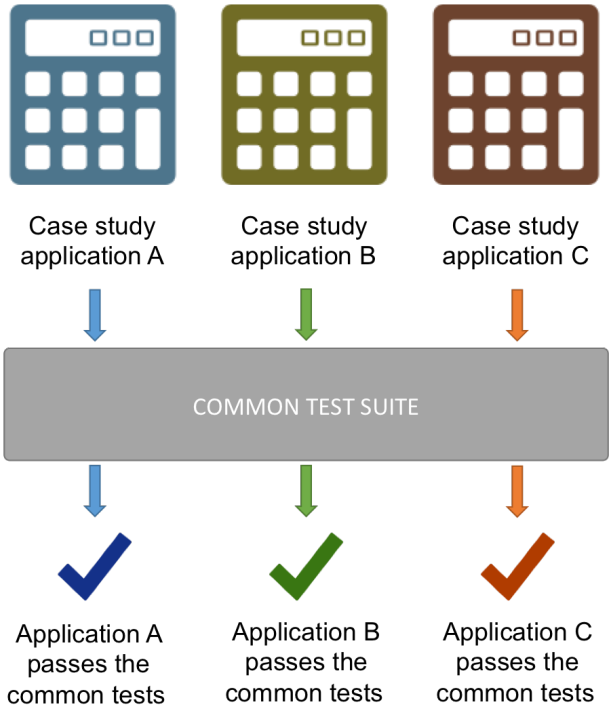

Figure 1. Three study applications. All applications have the same interface and functionality (they all pass a common test suite), but each has a different internal implementation.

Since we had three case study applications, the students were divided into three groups based on the results of two test quizzes: one testing the knowledge of MDE theory and the other testing their knowledge of the Calculator domain and tests (which are the same for all three test applications). To each group of students we have explained the overall organization of their study application, and they were left to freely explore it.

The students were informed that they are participating in an empirical experiment regarding the understandability of models in MDE. Also, they were notified that their work will not affect their final grade in the course.

3.2 Data Collection

In order to test the understandability of each application, we have designed three web questionnaires[2] – one questionnaire per study system and group. The questions in a questionnaire are in the form: "choose one" or "choose many" correct answers. The web questionnaire displays one question at a time, and measures both the time spent on the whole quiz, as well as the time spent on each individual question. All three questionnaires have the same questions, but different answers, which depend on the specifics of the implementation. In addition, a question has the same number of correct answers at the same positions in all three questionnaires (e.g. in all questionnaires, the correct answers to the third question are answers b and c). In that way, since we are measuring time spent on each question, all questions are on equal basis (e.g. it would be unfair, in terms of time spent, if there is only one correct answer that that answer is a in one questionnaire and d in another).

We determine the understandability of the system, by measuring the average number of correct answers and the time it takes to solve them. While answering the questionnaire, the students can freely explore their case study application. The students were given 45 minutes to answer 20 questions, and they were informed that we measure the time it takes to solve each question.

[2] `http://pzi.fesb.hr/Josip.Maras/MDE/index.php`

3.3 Student Opinions

During the same course, the students were given a separate questionnaire through which they could express their opinions on the performed study. Specifically, we were interested in their attitudes to combining education, research, and industry experience. The list of questions and answers can be found in Table 1, while the results are shown in Figures 2 and 3.

3.3.1 *Combining Education and Industrial Experience*

The first two questions, **Q1** and **Q2** were designed to find out student attitudes towards combining education and industrial experience (Figure 2). Out of 59 participants, 32 students have said that they *Agree*, and 26 students that they *Strongly Agree* with the statement: "*Including an industrial expert in a software engineering course has given me a deeper insight into software development (**Q1**)*", while one student thinks it did not have an influence on his/her insight. In addition to that, a great majority of students (44 *Strongly Agree*, 14 *Agree*, and 1 *Disagree*), thinks that "*It would be good if most courses would strive to include a similar form of industry cooperation (**Q2**)*".

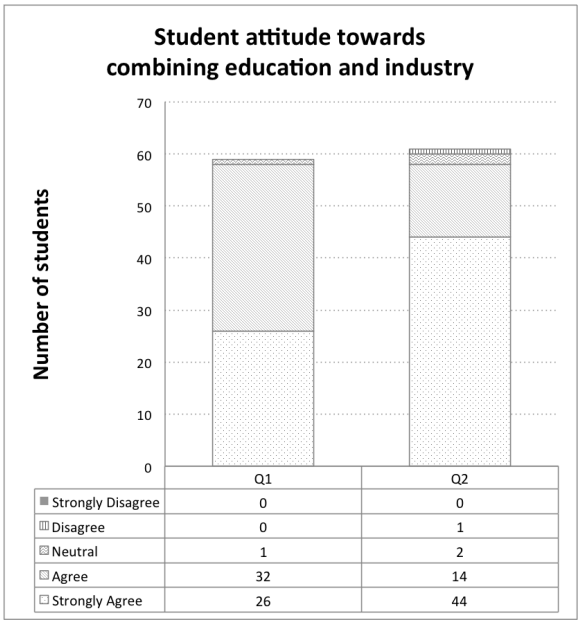

Figure 2. Student attitudes towards combining education and industry. **Q1.** "*Including an industrial expert in a software engineering course has given me a deeper insight into software development*"; **Q2.** "*It would be good if most courses would strive to include a similar form of industry cooperation*"

Table 1. Survey questions designed to explore the attitudes of students towards combining education, research, and industry.

Question	Possible answers				
Q1. Including an industrial expert in a software engineering course has given me a deeper insight into software development	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Q2. It would be good if most courses would strive to include a similar form of industry cooperation	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Q3. Participating in this research has had a positive influence on my understanding of MDE	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
Q4. I would again participate in a similar research	No			Yes	

These results show that students highly value combining education with industry experience, and that effort should be invested in achieving education - industry cooperation.

3.3.2 Combining Education and Research

The last two questions, **Q3** and **Q4** were designed to find out student attitudes towards combining education and research (Figure 3). Again, that attitude is highly positive: 36 students *Agree*, 16 *Strongly Agree*, 4 think that it did not have an impact (*Neutral*), and 2 *Disagree* with the statement that "*Participating in this research has had a positive influence on my understanding of MDE(**Q3**)*". In addition, if a similar study would be conducted, 54 students would again like to participate in the study, while 5 would not.

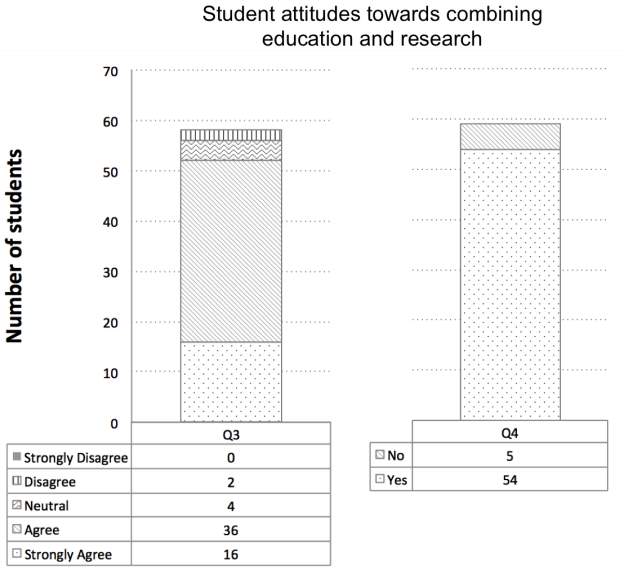

Figure 3. Student attitudes towards combining education and research. **Q3.** "*Participating in this research has had a positive influence on my understanding of MDE*"; **Q4.** "*I would again participate in a similar research.*"

The results show that the students think that participating in this research was beneficial for their understanding of the topic. We are especially encouraged by their willingness to participate in other, similar studies.

4. LESSONS LEARNED

The empirical study described in this paper, was a result of cooperation between education and industry, designed with the goal of being a pilot study for an industry relevant topic - the understandability of models in MDE. In the study there are three different points of view that had to be taken into account: *i) Educator's* perspective, *ii) Industrial representative's perspective*, and *iii) Researcher's perspective.*

4.1 Educator's Perspective

The goal of the software engineering educator is to provide students with skills relevant for the development of high-quality software systems. Certain software engineering classes place more emphasis on theoretical concepts (that is the case with our class "System analysis and design"), in which students learn about best practices in a somewhat idealized fashion. Bringing in an industry expert, with real, practical experience in day to day use of technologies and software development processes as an educator, comes with the benefit of introducing a more balanced approach to a software engineering course. As the results of our survey show, the students were very satisfied with this inclusion, and in their opinion other courses would also benefit from industry cooperation. However, including an industry expert is not always possible, since the organization from which the expert comes has to be willing to invest time into supporting education.

4.2 Industrial Representative's Perspective

The industry has two different benefits in participating in an empirical study that combines education, industry, and research. First, by participating in designing the curriculum, they ensure that these students, which will be future software developers, whose education they have influenced, have a particular set of skills that are required by the company. In that way, they lower the cost of including newly employed novice software developers into the normal company work flow.

Secondly, when a cooperation between software engineering researchers from industry and universities is established, the industry can influence the topic of research, ensuring that a researched topic is of their interest.

Performing empirical studies on professional developers is expensive, and in general a number of unforeseen problems can occur when constructing and carrying out empirical research. For this reason, students can be viewed as a more

cost-effective alternative. While carrying out empirical research with students as subjects might not be necessarily generalizable to a wider developer population, the results are still indicative. In addition, students can be used to carry out pilot studies, whose results can be used for narrowing down the research hypothesis or improving the empirical research protocol. However, it is important to emphasize that student interests should always remain in special focus.

4.3 Researcher's Perspective

Carrying out empirical research within the context of a normal software engineering class comes both with certain advantages and disadvantages.

The obvious advantage is that students are generally required to attend all lab assignments and that it is possible to, over a course of a semester, carry out longer subject preparations that will ensure that the students have a particular level of skill required for the experiment. At the same time, a single course usually covers several different topics, and we have to be very careful that subject preparation do not end up taking time that would otherwise be spent on other class topics that are not necessarily important from the perspective of current research, but that are important from an educational perspective. In addition, the lab schedule is usually predefined at the beginning of a semester, and it can be difficult to arrange additional labs, when the situation requires it.

There's also the problem of student attendance. If an experiment requires longer preparation, over the course of several weeks, there is a good chance that, due to various reasons, some students will not be able to attend all labs. This can cause a conflict between the research perspective and the educational perspective. From the research perspective, the inclusion of subjects that have not received the full preparation treatment is questionable, since they are not on equal terms with other subjects. But from the educational perspective, we should not withhold the learning experience that comes with the participation in an experiment, simply because the student was not able to achieve full attendance. One option is to allow all students to participate in the experiment, but not to take their results into account when testing the hypothesis.

Even though this was not the case with our research, certain empirical studies have to have control groups. This might be problematic from the educational perspective since all students should receive the same educational opportunities.

5. CONCLUSION

In this paper we have reported on a pilot empirical study with student participants that was designed to explore the understandability of models in model-driven engineering. The study was organized within the context of a "System's and analysis" course in cooperation with an industry representative. In this way, we have combined three different perspectives: *educational perspective* – the students were learning about a relatively novel software engineering approach, model-driven engineering; *research perspective* – the students were participants in a study that explores an industry relevant topic, the understandability of models; and *industrial perspective* – the students were prepared for the experiment by an industrial representative.

According to student surveys, the students were positive towards the whole experience – the majority of them highly value combining education and industrial experience and think that other courses should incorporate this mode of teaching. In addition, the students were also highly positive towards participating in research. They feel that the participation in the study has helped their understanding of the subject at hand, and they would again participate in similar research.

6. REFERENCES

[1] A. Abran, P. Bourque, R. Dupuis, and J. W. Moore. *Guide to the software engineering body of knowledge-SWEBOK*. IEEE Press, 2001.

[2] J. Carver, L. Jaccheri, S. Morasca, and F. Shull. Issues in using students in empirical studies in software engineering education. In *Software Metrics Symposium, 2003. Proceedings. Ninth International*, pages 239–249, Sept 2003.

[3] J. Carver, L. Jaccheri, S. Morasca, and F. Shull. Using empirical studies during software courses. In *Empirical Methods and Studies in Software Engineering*, volume 2765 of *Lecture Notes in Computer Science*, pages 81–103. Springer Berlin Heidelberg, 2003.

[4] A. Causevic, D. Sundmark, and S. Punnekkat. Test case quality in test driven development: A study design and a pilot experiment. In *Evaluation Assessment in Software Engineering (EASE 2012), 16th International Conference on*, pages 223–227, May 2012.

[5] N. E. Fenton and S. L. Pfleeger. *Software Metrics: A Rigorous and Practical Approach*. PWS Publishing Co., Boston, MA, USA, 2nd edition, 1998.

[6] O. S. Gómez, J. L. Batún, and R. A. Aguilar. Pair versus solo programming - an experience report from a course on design of experiments in software engineering. *CoRR*, abs/1306.4245, 2013.

[7] M. Höst, B. Regnell, and C. Wohlin. Using students as subjects—a comparative study ofstudents and professionals in lead-time impact assessment. *Empirical Softw. Engg.*, 5(3):201–214, Nov. 2000.

[8] L. Jaccheri and S. Morasca. On the importance of dialogue with industry about software engineering education. In *Proceedings of the 2006 International Workshop on Summit on Software Engineering Education*, SSEE '06, pages 5–8, New York, NY, USA, 2006. ACM.

[9] D. E. Perry, A. A. Porter, and L. G. Votta. Empirical studies of software engineering: A roadmap. In *Proceedings of the Conference on The Future of Software Engineering*, ICSE '00, pages 345–355, New York, NY, USA, 2000. ACM.

[10] P. Runeson. Using students as experiment subjects–an analysis on graduate and freshmen student data. In *Proceedings of the 7th International Conference on Empirical Assessment in Software Engineering.–Keele University, UK*, pages 95–102. Citeseer, 2003.