Guidelines for Using Empirical Studies in Software Engineering Education

Fabian Fagerholm[1] , Marco Kuhrmann[2] , Jürgen Münch[3]

Abstract: Software engineering education is supposed to provide students with industry-relevant knowledge and skills. Educators must address issues beyond exercises and theories that can be directly rehearsed in small settings. A way to experience such effects and to increase the relevance of software engineering education is to apply empirical studies in teaching. In our article, we show how different types of empirical studies can be used for educational purposes in software engineering. We give examples illustrating how to utilize empirical studies, discuss challenges, and derive an initial guideline that supports teachers to include empirical studies in software engineering courses.

This summary refers to the paper *Guidelines for Using Empirical Studies in Software Engineering Education* [FKM17]. This paper was published in the PeerJ Computer Science journal.

Keywords: Software Engineering Education, Computer Science Curricula, Teaching Methods, Empirical Studies, Experimentation, Education, Guideline

1 Introduction

Providing relevant knowledge and skills is a continuous concern in software engineering education. Students must be exposed to realistic settings to understand why applying fundamental software engineering principles is necessary, why decisions should be grounded in evidence, and to learn to foresee long-term and delayed effects of certain behavior or decisions in software projects. Using empirical instruments is one approach to teach relevant software engineering knowledge and skills.

Problem Statement & Objective Since real-life software development routinely deals with large, software-intensive systems and is influenced by the manifold and complex effects of teamwork and distributed software development, software engineering education must enable students to understand such environments and to apply knowledge properly and effectively. However, restrictions in the academic curriculum and the complexity and criticality of real software products limit the level of realism that can be achieved in education. There is a

[1] University of Helsinki, Helsinki, Finland & Blekinge Institute of Technology, Karlskrona, Blekinge, Sweden, fabian.fagerholm@helsinki.fi

[2] Clausthal University of Technology, Institute for Applied Software Systems Engineering, 38640 Goslar, Germany, kuhrmann@acm.org

[3] Reutlingen University, Böblingen, Germany, Juergen.Muench@Reutlingen-University.de

lack of guidance on how to use empirical studies in software engineering education. In order to address this gap, our article [FKM17] provides an overview of different types of empirical studies, their suitability for use in education, as well as challenges with respect to their execution.

Contribution Based on our experience (e.g., [KM16, Fa17]) we show how different instruments, ranging from controlled experiments to qualitative studies, can be used for teaching purposes. We also consider overarching approaches that situate empirical studies in a larger context. We systematize purposes and challenges of the different study types, discuss the validity of the results that can be obtained in a teaching context, and create a link between teaching and research. The guidelines developed in this paper provide a systematic collection of purposes, learning goals, challenges, and validity constraints, and aims to support teachers in selecting proper study types for inclusion into their courses.

2 Results

We provide an initial assignment to the four major study types experiment, case study, simulation, and continuous experimentation. We derived a set of purposes and challenges relevant for selecting a particular study type. We identified a total of ten purposes, describing major learning goals associated with empirical studies in software engineering teaching that we consider important. Complementing the purposes, we identified eight challenges that should be taken into account when designing empirical studies for educational purposes. Taking the close relation to industry and relevance of the topics into account, we analyzed the different study types for validity constraints. Therefore, we derived four validity considerations associated with empirical studies.

3 Conclusion

The guideline presented in this paper represents a starting point based on reflection grounded in teaching practice. It can be seen as an initial systematization of empirical instruments from the educational perspective. Although the experiences from our cases show positive impacts, the learning outcomes of the application of empirical instruments should be further explored: beyond what is currently known, what do students learn by conducting empirical studies, and how do their learning outcomes differ from other approaches to software engineering education?

References

[Fa17] Fagerholm, Fabian; Guinea, Alejandro Sanchez; Mäenpää, Hanna; Münch, Jürgen: The RIGHT model for Continuous Experimentation. Journal of Systems and Software, 123:292–305, 2017.

[FKM17] Fagerholm, Fabian; Kuhrmann, Marco; Münch, Jürgen: Guidelines for Using Empirical Studies in Software Engineering Education. PeerJ Computer Science, 3:e131, 2017.

[KM16] Kuhrmann, Marco; Münch, Jürgen: When Teams Go Crazy: An Environment to Experience Group Dynamics in Software Project Management Courses. In: Proceedings of the 38th International Conference on Software Engineering Companion. ICSE '16, ACM, New York, NY, USA, pp. 412–421, 2016.