

Experiences on Using Software Experiments in the Validation of Industrial Research Questions

Dominik Gessenharter¹, Alexander-Marc Merten², Alexander Raschke¹ and Nicolas Fernando Porta²

¹ Ulm University, Institute for Software Engineering and Compiler Construction, D-89069 Ulm, Germany

Dominik.Gessenharter@uni-ulm.de, Alexander.Raschke@uni-ulm.de

² DaimlerChrysler, Group Research & Advanced Engineering, Ulm
alexander-marc.merten@daimlerchrysler.com, nicolas.porta@daimlerchrysler.com

Abstract. Experimentation in software engineering is difficult. One reason is the large number of context variables [1] and the impracticality of experiments in an industrial setting. Considering the budgets of comprehensive projects, it is apparent that a company cannot double its effort executing a project twice, in order to compare two different approaches concerning process or method improvement. Performing experiments on the basis of small projects seldom offers solutions valid for industrial settings. Our commendation is a cooperation between industry and academic education. This approach offers multiple advantages. In this paper, we outline our experiences in experimental software engineering gained in about 20 experiments over the past 10 years by the Ulm University cooperating with DaimlerChrysler Group Research & Advanced Engineering, Ulm. Additionally we provide an insight into a current experiment and present our approach to experimental software engineering in further detail.

1 Introduction of an cooperation of industrial and academic educational stakeholders

The cooperation of the Ulm University and DaimlerChrysler(DC) implements the idea of proving new approaches concerning software or process engineering and improvement. Whereas DC describes the way of looking at a problem and proposes a solution, the University proves the proposal by conducting an applicable experiment. Though, the crux of the matter is, that academic and industrial environments have little deals in common. In addition, the real world and the surroundings of an experiment are in principle two different things. In spite of these differences, the described cooperation is a win-win situation for both partners as we will show.

1.1 Ulm University and DaimlerChrysler research - Stakeholders of two different application areas

The main interest of an industrial stakeholder is to have a prove of concept for ideas in improving software engineering or software development processes. In

order to benefit from innovations in the software engineering processes or the rise of product quality by using new tools, innovations or tools are used in a test run at the university. This task is accomplished by offering a practical course for students who will sample the modified process or the tool that might - after further research - be introduced. Whereas companies take stock in the results of such a field test, the university is interested in current industrial practises.

1.2 The idea of merging industrial and academic issues

There is a win-win situation for both, the industrial and the academic part of the cooperation. On the one hand, the industrial partner is outsourcing the planning, conduction and evaluation of the experiments and the measured data. That leads to considerable reduce of costs on the industrial side. This advantage is not paid by less quality rather than by having impartial participant for the experiment, i.e. the students. They do not have any knowledge about customs or preferences in the business practice of the industry in general or the actual cooperation partner in detail. However, the lack of experience of the student in contrast to the staff of a company must be considered. On that account, the subject matter of an experiment is not influenced by any habits of the assumed domain.

On the other hand, the university obtains the possibility to teach their students nowadays business practise and problems.

1.3 Empirical Software Engineering vs. Experimental Software Engineering

Empirical Software Engineering makes use of evaluating data collected from any project that is related to the matter of interest. The problem is, that collecting data in some projects lacks the comparability if there are no data to be compared to. When gathering data from experiments, it is possible to change the surrounding variables to figure out a benefit or a drawback on any modification of the software engineering process, methodologies or the used tools. Furthermore projects in industrial environment can not take the risk of implementing methods without having any idea of their practicability as the financial impact in case of a failure would be unacceptable.

We consider our efforts to be rather experimental software engineering than empirical software engineering.

1.4 The set-up of a typical experiment

An experiment needs a planning, conduction and evaluation phase. Every phase is essential since every deficiency in any phase might cause following deficiencies in subsequent phases. Bad planning as for example bad scheduling, will likely result in pressure of time during some parts of the experiment and therefore the collected data may be not feasible. In the following three chapters, we describe our proceeding and our experiences of each phase in experimental software engineering.

2 Planning, scheduling and designing an experiment

With this chapter, we describe an experiment as it is typical in our cooperation with DC. This includes details of the experiments object, its scheduling and some aspects of how teamwork can be organized.

2.1 Planning the product, development-phases, development-process and deliverables

Experiments as we did for the past 10 years are always used to support the evaluation of hypotheses which deal in most cases with process or method improvements. The experiment therefore must be based on convenient artefacts and use adequate processes. Cooperating with DC, we most often use examples of the field of embedded systems, e. g. different kinds of controllers in a car whereas processes are often varied.

When being committed to a product, the next step in planning an experiment is to decide, which phases of the development process are necessary for the analysis of the experiments hypothesis. In some cases, we concentrate on the analysis phase and typical documents like a specification of a product, in other cases design phase or exclusively the implementation are the focus of interest.

Depending on the hypothesis, the observed phases and the underlying process as well as all deliverables must be specified and documented. The reason is, that the experiment must be comprehensible and controllable. Therefore, every detail of the planning of an experiment must be available in written form in order to be able to iterate the experiment. A documentation of every single step in planning and conduction is a crucial point whenever the product or document quality is used as an indicator of the compliance of a hypothesis.

2.2 Scheduling of an experiment

The main focus when scheduling an experiment is to keep in mind, that the overall time must be limited. If not, the experiment is to distant from reality, where time is an economic factor. Due to the fact that students are not experienced developers, a training phase should be arranged before an experiment starts. The remaining overall time must be divided into fixed periods assigned to special activities. The partition of the development process is the basis of a division of work that allows to simulate a client-contractor relationship. Therefore, the deadlines for deliverables of each phase must be fixed at the very beginning of the experiment. During training, all examples should not antedate problems or even solutions of the experiments object.

2.3 Designing an experiment

We consider the design of an experiment as a means of controlling how students work together. On the one hand, we support team work all over the experiments

