Aspects of Large-Scale Process Studies in Industrial Settings
Position Paper

Andreas Birk
Fraunhofer Institute for Experimental Software Engineering
Sauerwiesen 6
D-67661 Kaiserslautern, Germany
+ 49 6301 707 256
Andreas.Birk@iese.fhg.de

ABSTRACT
Empirical studies are about to become widely established in software engineering. The spectrum of study types ranges from highly focused, (quasi-)experimental investigations in laboratory settings to long-term studies of industrial software development practices that span different business domains. The majority of studies belongs to the first type. Much less guidance is available for the planning and execution of more complex types of software engineering studies.

This paper presents aspects of large-scale process studies in industrial settings and describes an example of such a study. It also outlines possible sources of further technical and methodological advice from other disciplines of empirical research.

KEYWORDS
Software Process, Empirical Research, Research Strategies.

1 INTRODUCTION
Empirical software engineering research becomes more and more common. Many studies focus either on quasi-experimental designs in laboratory settings (cf. [3]) or on the investigation of well-focused phenomena in a single software project of interest. While these kinds of studies are very appropriate for some research questions, they are not suitable, if the object or context of a study exceed a certain degree of complexity.

This paper addresses aspects of empirical studies that can be characterized as *large-scale process studies*. These studies have attributes like the following:

- Object of study is a process (usually difficult to grasp)
- Study is conducted over a long period of time
- Industrial, non-laboratory environments
- Several very diverse environments of study
- A complex set of hypotheses (empirical theories)
- Very limited control over variables
- Large number of contextual factors

These studies do not necessarily require that the studied process is large (e.g., long projects or large, distributed teams). The focus is put on the complexity of the study, which is determined by factors like the duration of the study, the number of different instances studied, or the complexity of the research question investigated.

Examples of large-scale process studies are those that investigate the effectiveness of improvement methodologies, the critical success factors of software engineering technologies of interest, the long-term maintainability of complex software architectures, and the characteristics of development methodologies in selected business domains.

The following sections briefly introduce an example of a large-scale process study, discuss difficulties involved in such studies, and outline possible support by concepts from other empirical disciplines.

2 EXAMPLE OF A LARGE-SCALE PROCESS STUDY

An example of a large-scale process study has been conducted in the European applied research and technology transfer project PROFES[1][8]. PROFES has developed an improvement methodology that identifies improvement actions based on the definition of organization-specific product quality goals and integrates approaches like process assessments, process modeling, and software measurement.

The major evaluation criterion of the *PROFES Improvement Methodology* was whether it effectively guides the attainment of initially defined, organization-specific product quality goals. Other evaluation criteria addressed process improvements (e.g., standardization of work practices), the development of the quality awareness in the organization, as well as team building and organizational culture.

The investigations were developed in close collaboration with three industrial software organizations from the embedded systems domain. The methodology was applied in several software projects at these organizations over a period of three years. These applications of the methodology were evaluated and the evaluation results were fed back to the methodology developers. They served as an input for enhancing the methodology further and for illustrating its effectiveness (cf. [2]).

In addition, detailed effort data about the activities of the improvement programs have been collected. From this data, effort models about process assessments and software measurement programs have been developed that contain effort numbers per involved role (e.g., measurement engineer or manager) and activity (e.g., measurement planning, data collection, or data analysis). There have been derived different instances of such models for different kinds of organizations (e.g., organizations that conduct measurement for the first time and those that have routine in software measurement). Results from these analyses are presented in [6] [7].

3 DIFFICULTIES INVOLVED IN LARGE-SCALE PROCESS STUDIES

The experiences from the empirical investigations in *PROFES* pin-point several difficulties that can be regarded typical for large-scale process studies. In the following, three such difficulties are outlined and discussed briefly.

Gaining the buy-in of industrial software organizations

The first challenge that empirical investigations in industrial settings face is to find organizational units or projects that are willing to participate in the study. Usually, the difficulty is nor to create interest into the research question nor the funding of the organizations involvement. The most important obstacles are the overhead effort and possible disturbances of the project work that might be caused by the study and that distract the team from the software development tasks.

In *PROFES*, the motivation of the project teams and their willingness to participate in the study was very high. However, there were phases in which individual projects had very critical schedule constraints. During these periods, it had to be tolerated that the availability of project personnel was very limited. The needed data could almost always be gained later when the schedule pressure had decreased.

A study must be carefully designed in order to avoid too much overhead effort or disturbance of the software projects in which the study is performed. This calls for focused and well-planned studies and raises the need for observation and data collection techniques that meet the needs of software projects.

1.ESPRIT Project No. 23239, *PROFES* (*PROduct-Focused improvement of Embedded Software processes*). Members of the *PROFES* Consortium were Dräger Medical Technology (NL), Ericsson Finland (FIN), Etnoteam (I), Fraunhofer IESE (D), Tokheim (NL), University of Oulu (FIN), and VTT Electronics (FIN).

Study planning in yet unexplored fields
The focused planning of a study requires a considerable amount of concepts and baseline knowledge about the subject of study. For instance, there must be a basic knowledge and understanding of empirical phenomena before detailed hypotheses or theories can be formulated. Many areas of software engineering are not yet understood sufficiently well. So advanced hypotheses can not always be formulated or operationalized.

A helpful guidance for the formulation of hypotheses and related tasks of study planning is the common distinction of exploratory, descriptive, and causal types of studies. Depending on the available baseline knowledge about a field of endeavor, a study should be classified explicitly into one of these categories. Then, the study objectives should be clarified and the appropriate empirical techniques be selected.

In *PROFES*, the effort measurement, for instance, could benefit from related investigations in a previous project. A detailed effort measurement procedure could be set up in a straight-forward manner. The already existing data baseline facilitated the analysis of the newly collected data and the formulation of different variants of detailed effort models.

In contrast to this, it was at first quite challenging to find a way for tracing product quality improvements back to the improvement actions that have caused the quality improvements. This was needed for the validation of the effectiveness of the *PROFES Improvement Methodology*. In this case, considerable baseline work had to be done first (including *exploratory* empirical research) before appropriate analysis concepts could be developed (cf. [2]).

Presentation, dissemination, and exploitation of results
The main aim of every empirical study is to advance the understanding of the given field of endeavor. In order to make as many people benefit from a study as possible, the study results should be presented in a way that allows for identifying and reusing these results easily.

Particularly in applied research, scientific publications are just one means of exploiting the results of investigations. Other equally important means of exploitation are updates and enhancements of the studied concepts (such as new versions of methods, tools, or models) or some kinds of lessons learned documentation that transfers empirical findings to its potential users (cf. [5]).

A framework for the continuous learning and reuse of software engineering experience is the Quality Improvement Paradigm (QIP) / Experience Factory (EF) approach by Basili et al. [1]. It includes the concept of a knowledge repository called *Experience Base*, which stores and organizes reusable experience in the form of so-called *Experience Packages*.

In *PROFES*, the Experience Factory concept has been adopted for organizing the empirical findings. The various study results are made available for reuse through a web site [7]. The experience from this work indicates that the internet–besides other means like technical handbooks, training, and research reports–is an appropriate means for presenting and disseminating the results from empirical studies. Future investigations can take up the presented results and feed back their findings. Thus, a community-wide body of empirical knowledge can be developed and advanced gradually.

However, although the Experience Factory and related concepts of organizational learning provide a good basis for the presentation and reuse of empirical findings, further support for the implementation of these concepts is needed. Specialized representation schemas, tailored models, and customized strategies for informed decision making should be developed for the various different kinds of empirical knowledge that can possibly be stored in such knowledge repositories.

4 POSSIBLE SUPPORT BY OTHER EMPIRICAL DISCIPLINES
Several disciplines of empirical research have faced similar challenges as empirical software engineering does today. It is worthwhile to find out what software engineering can learn from these other disciplines. In the following, three possibilities of such knowledge transfer are outlined.

Marketing research
Marketing research has developed specific instruments for the rapid investigation of new products and market trends. These can support the development of empirical theories about newly emerging phenomena. They thus facilitate the planning of studies in yet unexplored fields of software engineering.

The following aspects of marketing research can be particularly useful for software engineering:

- The use of survey research, which is not very established in software engineering, yet.
- The integrated use of laboratory and survey research for different phases of the same research program.
- The intensive use of explicitly defined empirical research projects that typically are structured into the three phases: exploratory, descriptive, and causal research.

Pharmaceutical research
In pharmaceutical research, one concern of empirical studies is to ensure drug safety. Every drug can cause some adverse events that may be more or less serious. The pharmaceutical industry maintains a large system for monitoring the occurrence of adverse events. So-called *Drug Safety Groups* receive reports from hospitals that document possible adverse events from drug treatments. Based on these reports, the Drug Safety Groups trigger in-depth analyses of drug effect and maintain a large body of knowledge about drugs and their adverse events. There exist standardized forms for the reporting of adverse events, legal regulations prescribe how the information is to be handled and archived, and feedback mechanisms to the hospitals are defined for the case that new adverse events occur.

Empirical research in software engineering can use this system as a highly developed prototype of a feedback system between the developers of software engineering technology (e.g., tool vendors or research institutes as the analogy for the pharmaceutical industry) and the technology user community (i.e., the industrial software organizations as the analogy for the hospitals). This system can also illustrate the efficient collection of standardized data about technology usage across a variety of different organizations. In addition, it can provide strategies for the continuous evolution of a body of community-wide empirical knowledge. This can foster the participation of industrial software organizations in empirical research and facilitate the exploitation of research results.

Social sciences
Social sciences are often cited as a model of many strategies and instruments for empirical software engineering research. For large-scale process studies, social science methods might provide solutions in particular for the following aspects:

- Observational methods that have little impact on the studied environment.
- Methods for the analysis of qualitative data in non-laboratory settings (e.g., action research [9]).
- Methods for the aggregation of different lines of empirical evidence such as Grounded Theory [4].

These methods can be the foundation for developing a set of specialized empirical methods for software engineering that avoid unnecessary disturbance to software development projects. This can help to increase the participation of industrial software organizations in empirical studies.

5 CONCLUSION
The role of empirical studies in software engineering has been becoming increasingly important since the early nineties. However, additional efforts are still needed for making empirical research as established and mature as it is in many other disciplines. Experiences from these disciplines can be useful for accelerating the progress in empirical software engineering.

It can be argued that empirical research in software engineering is particularly challenging for reasons like the following: (1) Software engineering is still a young discipline that is undergoing frequent and rapid changes. This constantly invalidates part of the empirical knowledge and calls for the development of new hypotheses and theories. (2) The importance of empirical software engineering research is not yet shared throughout the entire community. This can make it hard to find industrial partner organizations at which empirical studies can be performed. (3) Many of the potential objects of study in software engineering are particularly difficult to grasp (e.g., abstract concepts like processes, organizations, architectures, or quality). This raises the need for advanced and specialized empirical research instruments.

In particular for large-scale process studies, additional appropriate research instruments are needed: (1) Data collection techniques are needed that have very little impact on the environment of the study, cause little overhead effort for the studied software organization, and can cope with periods in which the organization can not be approached due to schedule pressure. (2) Research methodologies are needed that effectively guide the rapid development of (sets of alternative) empirical theories through phases of exploratory, descriptive, and causal research. (3) Knowledge repositories should be developed that collect the results from empirical research and that foster the accumulation and exchange of the findings.

REFERENCES

1. Victor R. Basili, Gianluigi Caldiera, and H. Dieter Rombach. Experience Factory. In John J. Marciniak, editor, *Encyclopedia of Software Engineering*, volume 1, pages 469–476. John Wiley & Sons, 1994.
2. Andreas Birk, Janne Järvinen, and Rini van Solingen. A validation approach for product-focused process improvement. *In Proc. of the Int. Conf. on Product Focused Software Process Improvement (PROFES'99)*, Oulu, Finland, 1999.
3. Donald T. Campbell and Julian C. Stanley. *Experimental and quasi-experimental designs for research.* Houghton Mifflin, 1963.
4. Barney G. Glaser and Anselm L. Strauss. *The Discovery of Grounded Theory: Strategies for Qualitative Research*. Aldine de Gruyter, 1967.
5. Art Kleiner and George Roth. How to make experience your company's best teacher. *Harvard Business Review*, 75(5):172–177, September/October 1997.
6. PROFES Consortium. *The PROFES user manual*. Fraunhofer IRB, Stuttgart, Germany, 2000.
7. PROFES Consortium. *The PROFES cost/benefit repository.* ESPRIT Project no. 23239 PROFES. http://www.iese.fhg.de/projects/profes/CBRepository/CBRepository.html.
8. PROFES Consortium. *The PROFES web pages.* http://www.profes.org.
9. Rapoport, R.N. Three Dilemmas in Action Research. *Human Relations*, (23:4), 1970, pp. 499-513.