Preface

Have you ever had a need to evaluate software engineering methods or techniques against each other? This book presents experimentation as one way of evaluating new methods and techniques in software engineering. Experiments are valuable tools for all software engineers who are involved in evaluating and choosing between different methods, techniques, languages and tools.

It may be that you are a software practitioner, who wants to evaluate methods and techniques before introducing them into your organization. You may also be a researcher, who wants to evaluate new research results against something existing, in order to get a scientific foundation for your new ideas. You may be a teacher, who believes that knowledge of empirical studies in software engineering is essential to your students. Finally, you may be a student in software engineering who wants to learn some methods to turn software engineering into a scientific discipline and to obtain quantitative data when comparing different methods and techniques. This book provides guidelines and examples of how you should proceed to succeed in your mission.

Software Engineering and Science

The term "software engineering" was coined in 1968, and the area is still maturing. Software engineering has over the years been driven by technology development and advocacy research. The latter referring to that we have invented and introduced new methods and techniques over the years based on marketing and conviction rather than scientific results. To some extent, it is understandable with the pace the information society has established itself during the last couple of decades. It is, however, not acceptable in the long run if we want to have control of the software we develop. Control comes from being able to evaluate new methods, techniques, languages and tools before using them. Moreover, this would help us turn software engineering into a scientific discipline. Before looking at the issues we must address to turn software engineering into science, let us look at the way science is viewed in other areas.

In “Fermat’s Last Theorem” by Dr. Simon Singh, [160], science is discussed. The essence of the discussion can be summarized as follows. In science, physical phenomena are addressed by putting forward hypotheses. The phenomenon is observed and if the observations are in line with the hypothesis, this becomes evidence for the hypothesis. The intention is also that the hypothesis should enable prediction of other phenomena. Experiments are important to test the hypothesis and in particular the predictive ability of the hypothesis. If the new experiments support the hypothesis, then we have more evidence in favor of the hypothesis. As the evidence grows and becomes strong, the hypothesis can be accepted as a scientific theory.

The summary is basically aiming at hypothesis testing through empirical research. This may not be the way most research is conducted in software engineering today. However, the need to evaluate and validate new research proposals by conducting empirical studies is acknowledged to a higher degree today than 10 years ago. Empirical studies include surveys, experiments and case studies. Thus, the objective of this book is to introduce and promote the use of empirical studies in software engineering with a particular emphasis on experimentation.

Purpose

The purpose of the book is to introduce students, teachers, researchers, and practitioners to experimentation and empirical evaluation with a focus on software engineering. The objective is in particular to provide guidelines of how to perform experiments to evaluate methods, techniques and tools in software engineering, although short introductions are provided also for other empirical approaches. The introduction into experimentation is provided through a process perspective. The focus is on the steps that we have to go through to perform an experiment. The process can be generalized to other types of empirical studies, but the main focus here is on experiments and quasi-experiments.

The motivation for the book comes from the need of support we experienced when turning our software engineering research more experimental. Several books are available which either treat the subject in very general terms or focus on some specific part of experimentation; most of them focusing on the statistical methods in experimentation. These are important, but there is a lack of books elaborating on experimentation from a process perspective. Moreover, there are few books addressing experimentation in software engineering in particular, and actually no book at all when the original edition of this book was published.

Scope

The scope of the book is primarily experiments in software engineering as a means for evaluating methods, techniques etc. The book provides some information

regarding empirical studies in general, including case studies, systematic literature reviews and surveys. The intention is to provide a brief understanding of these strategies and in particular to relate them to experimentation.

The chapters of the book cover different steps to go through to perform experiments in software engineering. Moreover, examples of empirical studies related to software engineering are provided throughout the book. It is of particular importance to illustrate for software engineers that empirical studies and experimentation can be practiced successfully in software engineering. Two examples of experiments are included in the book. These are introduced to illustrate the experiment process and to exemplify how software engineering experiments can be reported. The intention is that these studies should work as good examples and sources of inspiration for further empirical work in software engineering. The book is mainly focused on experiments, but it should be remembered that other strategies are also available, for example, case studies and surveys. In other words, we do not have to resort to advocacy research and marketing without quantitative data when research strategies as, for example, experiments are available.

Target Audience

The target audience of the book can be divided into four categories.

Students may use the book as an introduction to experimentation in software engineering with a particular focus on evaluation. The book is suitable as a course book in undergraduate or graduate studies where the need for empirical studies in software engineering is stressed. Exercises and project assignments are included in the book to combine the more theoretical material with some practical aspects.

Teachers may use the book in their classes if they believe in the need of making software engineering more empirical. The book is suitable as an introduction to the area. It should be fairly self-contained, although an introductory course in statistics is recommended.

Researchers may use the book to learn more about how to conduct empirical studies and use them as one important ingredient in their research. Moreover, the objective is that it should be fruitful to come back to the book and use it as a checklist when performing empirical research.

Practitioners may use the book as a "cookbook" when evaluating some new methods or techniques before introducing them into their organization. Practitioners are expected to learn how to use empirical studies in their daily work when changing, for example, the development process in the organization they are working.

Outline

The book is divided into three main parts. The outline of the book is summarized in Table 1, which also shows a mapping to the original edition of this book. The first part provides a general introduction to the area of empirical studies in Chap. 1. It puts empirical studies in general and experiments in particular into a software engineering context. In Chap. 2, empirical strategies (surveys, case studies and experiments) are discussed in general and the context of empirical studies is elaborated, in particular from a software engineering perspective. Chapter 3 provides a brief introduction to measurement theory and practice. In Chap. 4 we provide an overview of how to conduct systematic literature reviews, to synthesize findings from several empirical studies. Chapter 5 gives an overview of the case studies as a related type of empirical studies. In Chap. 6, the focus is set on experimentation by introducing general experiment process.

Part II has one chapter for each experiment step. Chapter 7 discusses how set the scope for an experiment, and Chap. 8 focuses on the planning phase. Operation of the experiment is discussed in Chaps. 9 and 10 presents some methods for analyzing and interpreting the results. Chapter 11 discusses presentation and packaging of the experiment.

Part III contains two example experiments. In Chap. 12, an example is presented where the main objective is to illustrate the experiment process, and the example in Chap. 13 is used to illustrate how an experiment in software engineering may be reported in a paper.

Some exercises and data are presented in Appendix A. Finally, the book displays some statistical tables in Appendix B. The tables are primarily included to provide support for some of the examples in the book. More comprehensive tables are available in most statistics books.

Exercises

The exercises are divided into four categories, the first presented at the end of each chapter in Parts I and II of the book (Chaps. 1–11), and the other three in Appendix A:

Understanding. Five questions capturing the most important points are provided at the end of each chapter. The objective is to ensure that the reader has understood the most important concepts.

Training. These exercises provide an opportunity to practice experimentation. The exercises are particularly targeted towards analyzing data and answering questions in relation to an experiment.

Reviewing. This exercise is aimed at the examples of experiments presented in Chaps. 12–13. The objective is to give an opportunity to review some presented experiments. After having read several experiments presented in the literature, it is

Table 1 Structure of the book

Subject	Revised version	Original edition	Major updates
Part I. Background			
Introduction	1	1	
Empirical Strategies	2	2	New sections on replication, synthesis, technology transfer and ethics
Measurement	3	3	New section on measurement in practice
Systematic Literature Reviews	4	10[a]	New chapter
Case Studies	5		New chapter
Experiment Process	6	4	
Part II. Steps in the Experiment Process			New running example
Scoping	7	5[b]	Adapted terminology
Planning	8	6	
Operation	9	7	
Analysis and Interpretation	10	8	
Presentation and Package	11	9	Major revision
Part III. Example Experiments			
Experiment Process Illustration	12	11	
Are the Perspectives Really Different?	13		New chapter
Appendices			
Exercises	A	13	Understanding exercises moved to each chapter
Statistical Tables	B	A	

[a] Entitled Survey, and with a different scope
[b] Entitled Definition

clear that most experiments suffer from some problems. This is mostly due to the inherit problems of performing experimentation in software engineering. Instead of promoting criticism of work by others, we have provided some examples of studies that we have conducted ourselves. They are, in our opinion, representative of the type of experiments that are published in the literature. This includes that they have their strengths and weaknesses.

Assignments. The objective of these exercises is to illustrate how experiments can be used in evaluation. These assignments are examples of studies that can be carried out within a course, either at a university or in industry. They are deliberately aimed at problems that can be addressed by fairly simple experiments. The assignments can either be done after reading the book or one of the assignments can be carried out as the book is read. The latter provides an opportunity to practice while reading the chapters. As an alternative, we would like to recommend teachers to formulate an assignment, within their area of expertise, that can be used throughout the book to exemplify the concepts presented in each chapter.

http://www.springer.com/978-3-642-29043-5

Experimentation in Software Engineering

Wohlin, C.; Runeson, P.; Höst, M.; Ohlsson, M.C.;
Regnell, B.; Wesslén, A.

2012, XXIV, 236 p., Hardcover
ISBN: 978-3-642-29043-5