Foreword

Prof. Claes Wohlin, Blekinge Institute of Technology, Sweden

As the name of the field indicates, software engineering is expected to be an engineering discipline. However, it is not governed, to the same extent, by underlying mathematical models as many other engineering disciplines, in particular, those addressing physical artifacts as in electrical engineering or mechanical engineering. Thus, mathematics is insufficient to conduct research and improve in software engineering, although it is vital for some sub-areas within software engineering. There are several reasons for this insufficiency.

First of all, the software is invisible[1]. We can read the code, but we cannot see it in use. We can only observe the effect of the software being executed. Furthermore, software engineering is intrinsically complex since it is, to a considerable extent, dependent on the knowledge and capability of humans developing the software. Moreover, the ability of the individuals to work in a team contributing to the same software system is essential. The development is supported by different processes, methods, techniques, languages, and tools, which, in one way or another, are used by the organization developing the software. Thus, software engineering is an interplay between human knowledge, social networks of the individuals, and available assets in the organization developing the software[2].

To be able to study and improve the way software is engineered, many researchers have embraced and promoted software engineering as an empirical engineering discipline. Empirical studies were conducted early in the discipline, but they were quite rare. In 1986, an article describing experimentation in software engineering was published[3] outlining software engineering as an experimental science. The establishment of empirical software engineering was done to a large extent in the 1990ies. At the beginning of the 21st century, two books on experimentation in software engineering were published[4,5]. The former book came in a second edition in 2012[4], and it was published in Chinese in 2015.

In 2004, the concept of evidence-based software engineering was established in software engineering[6]. The evidence is most often generated from empirical studies, and hence, it was a natural continuation of the previous work on empirical software engineering. As the area of empirical software engineering became well-established, the need for advances in our

[1] F. P. Brooks, Jr, “No Silver Bullet – Essence and Accidents of Software Engineering,” IEEE Computer, Vol. 20, Issue 4, pp. 10-19, 1987.
[2] C. Wohlin, D. Šmite, and N. B. Moe, “A General Theory of Software Engineering: Balancing Human, Social and Organizational Capitals,” Journal of Systems and Software, Volume 109, pp. 229-242, 2015.
[3] V. R. Basili, R. W. Selby, and D. H. Hutchens, “Experimentation in Software Engineering,” IEEE Transactions on Software Engineering, Vol. SE-12, Issue 7, pp. 733-743, 1986.
[4] C. Wohlin, P. Runeson, M. Höst, B. Regnell, M. C. Ohlsson and A. Wesslén, “Experimentation in Software Engineering”, Springer-Verlag Berlin Heidelberg, 2012.
[5] N. Juristo and A. M. Moreno, “Basics of Software Engineering Experimentation,” Springer US, 2001.
[6] B. A. Kitchenham, T. Dybå, and M. Jørgensen, “Evidence-based Software Engineering,” Proceedings. 26th International Conference on Software Engineering, Edinburgh, UK, pp. 273-281, 2004.

conduct of empirical studies grew[7]. Given the applied nature of software engineering, the need to conduct empirical studies in a real-life context was strengthened by the publication of guidelines for conducting case studies[8].

As a continuation concerning the focus on evidence in software engineering, a book on evidence-based software engineering was published in 2015[9]. Furthermore, empirical software engineering has gone from being a sub-area of software engineering to be an integral part of software engineering. Nowadays, it is expected that research is evaluated and assessed using empirical methods. Thus, it is, in most cases, insufficient to present an idea or a solution without empirical evidence. In summary, software engineering has moved into truly being an engineering discipline.

The current book "Contemporary Empirical Methods in Software Engineering," edited by Prof. Michael Felderer and Prof. Guilherme Horta Travassos takes the next step by including chapters on essential and timely topics in empirical software engineering. The chapters are written by some of the world-leading experts on empirical methods in software engineering. The editors have done an excellent job of attracting experts in the field who contribute with essential topics concerning the empirical software engineering of today.

The book follows up on the previous books and articles on empirical and evidence-based software engineering. As the title of the book indicates, the book takes a timely step in including a set of chapters addressing emerging areas in empirical software engineering. It provides an excellent combination of chapters addressing contemporary areas of interest for anyone conducting research in software engineering and in particular, for those with a strong focus on empirical software engineering. The book is highly recommended to read for, in particular, Ph.D. students and researchers interested in conducting high-quality software engineering research aspiring to apply empirical research methods for today and the future.

[7] F. Shull, J, Singer and D. I. K. Sjøberg (editors), "Guide to Advanced Empirical Software Engineering," Springer-Verlag London, 2008.

[8] P. Runeson, M. Höst, A. Rainer and B. Regnell, "Case Study Research in Software Engineering – Guidelines and Examples," John Wiley & Sons, Inc., 2012.

[9] B. A. Kitchenham, D. Budgen and P. Brereton, "Evidence-based Software Engineering and Systematic Reviews," Chapman and Hall/CRC, 2015.