Empirical Evaluation in Software Engineering: Role, Strategy, and Limitations

Lionel C. Briand

Though there is a wide agreement that software technologies should be empirically investigated and assessed, software engineering faces a number of specific challenges and we have reached a point where it is time to step back and reflect on them. Technologies evolve fast, there is a wide variety of conditions (including human factors) under which they can possibly be used, and their assessment can be made with respect to a large number of criteria. Furthermore, only limited resources can be dedicated to the evaluation of software technologies as compared to their development. If we take an example, the development and evaluation of the Unified Modeling Language (UML) as an analysis and design representation, major revisions of the standard are proposed every few years, many specialized “profiles” of UML are being developed (e.g., for performance and real-time) and evolved, it can be used within the context of a variety of development methodologies which use different subsets of the standard in various ways, and it can be assessed with respect to its impact on system comprehension, the design decision process, but also code generation, test automation, and many other criteria. Given the above statement and example, important questions logically follow: (1) What can be a realistic role for empirical investigation in software engineering? (2) What strategies should be adopted to get the most out of available resources for empirical research? (3) What does constitute a useful body of empirical evidence?

It is evident that we cannot possibly assess and validate every single software technology being used or adopted under every possible relevant set of conditions with respect to every possible criterion. Empirical studies should therefore (a) target specific technologies which are of economic importance, (b) for which there is significant uncertainty in terms of cost-effectiveness, and (c) which must be investigated under the most representative or plausible conditions. Nevertheless, such assessments will always involve a significant amount of judgment and interpolation. Instead of focusing on unquestionable scientific evidence, our objective is rather to buy information to support decision making. Furthermore, because of the impact of human factors on the cost-effectiveness of many technologies (e.g., education, training, management structure), to be fully understood, the quantitative results of studies must be complemented with qualitative analysis and an investigation of subjective, human perceptions. There are many strategies to do so, ranging from simple questionnaire surveys to think aloud protocols.

An empirical body of evidence in software engineering can therefore be described as a set of studies, each performed under certain explicit conditions, for which both quantitative and qualitative, subjective and objective data have been collected, and based on which certain conclusions and interpretations have been provided. This may be completed by some form of meta-analysis attempting to find an emerging pattern across studies. However, how to make such information reusable in practice?

V. Basili et al. (Eds.): Empirical Software Engineering Issues, LNCS 4336, p. 21, 2007.