*4th International Workshop on Adoption-Centric Software Engineering*
(ACSE 2004: May 25, 2004; Edinburgh, Scotland, UK)

# On the Challenges in Fostering Adoption via Empirical Studies

**Shihong Huang**
Department of Computer Science
University of California, Riverside
shihong@cs.ucr.edu

**Scott Tilley**
Department of Computer Sciences
Florida Institute of Technology
stilley@cs.fit.edu

**Abstract**

*This paper outlines some of the issues in fostering adoption via empirical studies. There has been a welcomed shift in recent years towards the use of evidence-based techniques to support arguments of efficacy of proposed software engineering tools and techniques. When available, such objective indicators can provide a strong argument for (or against) the adoption of a particular approach. Unfortunately, gathering such evidence is not a trivial matter. Several of the key issues encountered in conducting a series of limited-scope qualitative experiments are used to illustrate the hurdles that must be overcome before empirical studies can enjoy more widespread use.*

**Keywords:** *adoption, empirical studies, evidence-based software engineering, education*

## 1. Introduction

One of the central tenets of software engineering is to adopt a more disciplined approach to the production of software artifacts [4]. There are many stated benefits to such an approach, such as increased quality, improved effort estimation, and lower maintenance costs. However, even with such impressive benefits, getting practitioners to adopt the methods is problematical without indisputable proof that the cost of adoption is outweighed by the return on their investment.

In other words, there is a need to employ evidence-based arguments to support the adoption of best practices in software engineering, rather than arguments based upon advocacy [8]. Objective evidence would facilitate adoption by providing an independent predictor of the likely benefits of the proposed approach. Unfortunately, many researchers do not provide such objective evidence. The result is that there is rarely any solid audit trail that can provide a validation of the ideas, and which links theory and concepts to observed practices [1].

One reason for this lamentable situation may be a lack of understanding on the part of many researchers in the academic community as to how to produce such evidence. For example, recent work in the reengineering of transactions for Web-based systems shows promise [2], but without a comprehensive case study the research is not as convincing as it might otherwise be. One of the most powerful techniques that could be used in this situation is an empirical study, but successfully realizing such a study is fraught with difficulties.

Our own personal experience in conducting two limited-scope qualitative empirical studies assessing the efficacy of UML as one type of graphical documentation in aiding program understanding supports this observation [6][7]. We are novice empiricists who are interested in using empirical studies as a measurement instrument; we are not particularly interested in becoming experts concerning empirical methods per se. As novices, we struggled with learning how to properly execute an empirical study, while at the same time remaining focused on the underlying problem that we were interested in exploring.

Of the numerous challenges in fostering adoption by means of empirical studies, one of the most vexing is dealing with threats to validity. All empirical studies can have possible threats to the validity of the experiment and the conclusions drawn from the analysis of the results. For qualitative experiments, these threats can be managed somewhat more easily than with quantitative experiments. Nevertheless, proactively addressing any possible doubts that others might have concerning the credibility of reported results is essential for using the experiemnt as a means of fostering adoption of the tool or technique that was the focus of the study.

The next three sections of the paper discuss specific challenges related to dealing with threats to validity for empirical studies: designing the experiment, selecting the participants, and choosing the sample system. These challenges are described from a personal point of view, and are certainly not meant to be exhaustive. The paper concludes by outlining possible avenues for future work to addresses these challenges.

## 2. Designing the Experiment

The first threat to validity is the nature of the experiment itself. The threat begins with questioning the basic premise for conducting the experiment: the hypothesis. In essence, this threat is concerned with verifying that the right question is being asked. It is fruitless to expend the energy needed for an empirical study on a question that has a patently obvious answer. At the same time, knowing which question to ask is not easy. There is a constant tug between asking a very specific question (whose answer may be accurate but not broadly applicable), and asking a more general question (whose answer may be subjective and hence would not hold up to third-party scrutiny).

In other disciplines, experiments are often tightly controlled, with a single variable the subject of investigation; all other variables are held constant. This approach is also possible in a software engineering experiment, but it may not be the most desirable choice. An experiment that is so narrowly focused may indeed produce objective results using a repeatable procedure, but those results may not support the broader goal of adoption. In contrast, an experiment that more closely recreates the conditions of an actual development project will necessarily have many variables, only one of which may be of interest to the researchers. Any results arising from the analysis of such an experiment can be questioned due to the uncertain traceability between cause and effect.

## 3. Selecting the Participants

The second threat to validity is the selection of participants in the experiment. A hard reality for most academic researchers is that the people available to participate in experiments are either students or colleagues. The advantage of having such participants is that one might be more knowledgeable about the skills and experience of each person, which could be valuable information when it comes to evaluating personal biases in the results.

The disadvantage of using friends and/or students in empirical studies is that such people are not necessarily representative of the society at large. If the goal of the experiment is to provide evidence to support widespread adoption of a particular tool or technique, it is important to have participants who reflect the likely characteristics of the larger body of users. Otherwise, arguments for (or against) efficacy will be tainted by the undue influence of the participants' skills (or lack thereof).

There is also the issue of scale: results from an experiment with only a handful of participants can rightly be called into question. In other fields, such as medicine, empirical studies often involve hundreds or thousands of participants. In most software engineering studies, researchers are often pleased if they can attract a dozen students to participate in their experiment. The statistical significance of such a small number of participants is questionable. But sometimes the results are still of value.

For small-scale experiments, the use of students as the primary experimental participants may be acceptable. However, in the context of fostering adoption, it is more desirable to involve people from the other two pillars of the community: government and industry. Unfortunately, it is notoriously difficult to get representatives from these two groups to participate in academic experiments. The reasons for this are many and varied, from lack of interest to lack of time. One technique that has proven successful is to change the image of the experiment from an exercise purely of interest to the academic, to a collaborative activity between equal partners. Carnegie Mellon's Software Engineering Institute adopted this approach during the development of the Software Capability Maturity Model [5] with great success.

## 4. Choosing the Sample System

The third threat to validity is choosing the sample system. Most academic empirical studies in software engineering are of limited scope (that is, performed during a short time frame with a relatively small number of participants). This limitation usually forces the researchers to make a choice between a sample system that is representative of a real-world problem, and a much smaller application that is (hopefully) still representative but much simpler.

The problem with choosing a sample system that is as close as possible to a real-world application is that the complexity inherent in such systems may be too much for the participants to master within the context of the experiment. In contrast, choosing a smaller and simpler sample system may make it easier for the participants to work with the material, but at the cost of potentially exposing the results to prejudice based on "toy" experiments – easy to run, but with results that are difficult to generalize.

The choice of the sample system is also complicated by the nature of the experiment's design. It is quite difficult to control "variables" in a scientific sense for most software engineering experiments. For example, if one is interested in evaluating the usefulness of a particular form of graphical documentation in support of program understand, the sample system should be complex enough that a typical software engineer would have some difficulty in understanding the application without the aid of such documentation. However, there is no such thing as a "typical" software engineer; previous

experience, knowledge of the application and implementation domain, and personal preferences for cognitive support can all influence analysis of the results.

The ability to reproduce the results of the same experiment by a different experimenter is one of the trademarks of a scientific study. One way to encourage this in software engineering may be to use an open source project as the sample system. This would seem to offer a commendable level of visibility into the experiment. However, using a commercial application also has advantages, particularly in terms of adoption: industry may be more receptive to results that derive from analysis of one of their own applications, as opposed to one that lacks the very important "ownership" attribute. Ownership leads to buy-in, and buy-in leads to adoption.

## 5. Summary

This paper briefly outlined three challenges related to threats to validity of empirical studies: designing the experiment, selecting the participants, and choosing the sample system. These challenges were identified from our own experience in organizing limited-scope empirical studies in software engineering.

There are of course many other challenges related to empirical studies outside of those concerning threats to validity, particularly in using the results of the experiments to foster adoption. For example, addressing ethical issues pertaining to treatment of the participants, running the experiment itself, and analyzing the results. Addressing these challenges requires education of both the researchers and the consumers of the results.

During the preparation for our own empirical studies we felt the need for a source of guidance on performing such studies, perhaps in the form of a handbook. There is a clear need for pedagogical material specifically targeted towards empirical studies in software engineering. The tutorial "Case Studies for Software Engineers" that is part of the ICSE 2004 conference program [3] is an excellent step towards satisfying this goal.

## References


[1] Budgen, D.; Hoffnagle, G.; Müller, M.; Robert, F.; Sellami, A.; and Tilley, S. "Empirical Software Engineering: A Roadmap." *Proceedings of the 10th International Conference on Software Technology and Engineering Practice* (STEP 2002: Oct. 6-8, 2002; Montréal, Canada), pp. 180-184. Los Alamitos, CA: IEEE Computer Society Press, 2003.

[2] Distante, D.; Parveen, T.; and Tilley, S. "Towards a Technique for Reverse Engineering Web Transactions from a User's Perspective." *Proceedings of the 12th IEEE International Workshop on Program Comprehension* (IWPC 2004: June 24-26, 2004; Bari, Italy). Los Alamitos, CA: IEEE CS Press, June 2004.

[3] Perry, D.; Sim, S; and Easterbrook, S. "Case Studies for Software Engineers." Tutorial presented at *The 26th International Conference on Software Engineering* (ICSE 2004: May 23-28, 2004; Edinburgh, Scotland, UK).

[4] Pfleeger, S. *Software Engineering Theory and Practice* (2nd Edition). Upper Saddle River, NJ: Prentice-Hall, 2001.

[5] Software Engineering Institute, Carnegie Mellon University. "CMMI Adoption." Online at http://www.sei.cmu.edu/cmmi/adoption/.

[6] Tilley, S. and Huang, S. "Assessing the Efficacy of Software Architecture Visualization Techniques for Recovered Artifacts." *Dagstuhl Seminar 03061: Software Architecture Recovery and Modeling* (Feb. 2 – 7, 2003; Schlöss Dagstuhl, Germany).

[7] Tilley, S. and Huang, S. "Workshop on Graphical Documentation for Programmers: Assessing the Efficacy of UML Diagrams for Program Understanding." Held in conjunction with *The 11th International Workshop on Program Comprehension* (IWPC 2003: May 10, 2003; Portland, OR).

[8] Tilley, S.**;** Huang, S.; and Payne, T. "On the Challenges of Adopting ROTS Software." *Proceedings of the 3rd International Workshop on Adoption-Centric Software Engineering* (ACSE 2003: May 9, 2003; Portland, OR), pp. 3-6. Published as CMU/SEI-2003-SR-004. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, June 2003.