

Empirical Evaluation of Software Maintenance Technologies

Filippo Lanubile¹

Department of Computer Science

University of Maryland

College Park, MD 20742

email: lanubile@cs.umd.edu

Abstract

Technology evaluation is part of the decision-making process of any software organization. Unlike conventional wisdom, empirical evaluation strives to avoid biased conclusions by relying on observation and looking for pitfalls in the evaluation process.

In this paper, we provide a summary of the maintenance studies presented in the session «Study and assessment of (new) technologies» of the International Workshop on Empirical Studies of Software Maintenance (WESS '96), and also report on the working group discussion which focused on common problems and open issues in the field of technology evaluation. These empirical studies are then classified according to a multi-dimensional framework to synthesize the state of the research in technology evaluation and ultimately discover interesting patterns.

1. Introduction

The evaluation of technologies has as its primary goal to answer questions about the effects of some existing software tool, technique, or method (here, generally referred to as software technologies).

In recent years, many new technologies have been proposed for developing and maintaining better software, on time and within budget. Some of these technologies have been introduced into production environments, but only a few have been widely accepted in the software industry. The technologies that have been applied, and then rejected because they were inappropriate, have wasted time and resources, and even caused trouble when applied on critical projects. Software practitioners do not want to miss

¹ Filippo Lanubile is on sabbatical from the University of Bari, Italy.

the competitive advantages derived from successful improvements. However, they are often forced to choose the technology to adopt, based on naive judgments.

Much of that we want to know about software technology leads to comparisons between alternative ways of performing a software engineering activity within a particular organization and software project. The question is: «Which technology is better than the others under specific circumstances?» However, since the application of a technology varies and is affected by reality, we are also interested in understanding if a technology that was found successful in another project or even in another organization, can be as successful in our environment. Here, the question is: «Which circumstances are better than others when using a specific technology?» This second question is even more difficult to answer than the former because technical, psychological, social, and managerial factors can be deeply intertwined and influence each other. Although it is widely recognized that a specific technology cannot be the best in all environments, we know little about the limits of applicability of technologies.

Much of what we believe about the benefits and effectiveness of software technologies appears to be «common sense» because it comes from our daily, ordinary observations of which technology is better, or if a technology is appropriate. Unlike conventional wisdom, empirical research always relies on observation to evaluate a hypothesis, and systematically tries to avoid biased conclusions. First, empirical research assumes that all constructs of interest must have observable features that we can measure, although imperfectly. Ordinary observers are unlikely to spend much time thinking about which data must be gathered and then collecting enough data to achieve confidence in the findings. Second, regardless of the field, empirical research always tries to exclude biases and pitfalls in the process of validating a hypothesis by using well-defined criteria to judge the quality of scientific research: construct validity, internal validity, and external validity. Scientists are not better than ordinary observers, but science provides a mechanism of critical review from peers before a finding can be accepted as knowledge. Without a scientific approach to software engineering we can only make conjectures about the competing benefits of different technologies.

During the International Workshop on Empirical Studies of Software Maintenance (WESS '96), a group of people with an interest in the evaluation of software maintenance technology were brought together in the session #1 «Study and assessment of (new) technologies». Participants who participated in the presentations and discussions were:

- Charles Ames, California Institute of Technology
- Brent Auernheimer, California State University
- Erich Buss, IBM Canada

- David Eichmann, University of Houston
- Keith Gallagher, Loyola College
- James Kiper, Miami University
- Filippo Lanubile (session chair), University of Maryland
- Mikael Lindvall, Linkoping University
- David Rosenblum, University of California
- Gregg Rothermel, Oregon State University
- Forrest Shull, University of Maryland
- Adrian Smith, University of Southampton

The objectives of the working group were three-fold:

1. share practical experience on the empirical evaluation of maintenance technologies. Participants were asked to present their current work focusing on the evaluation part rather than the technology itself;
2. identify and discuss open issues in technology evaluation. Discussion was triggered by problems encountered by participants in their experiences of technology evaluation;
3. arrive at a common framework for classifying empirical studies of technology evaluation. The application of the framework to the evaluation studies presented at the workshop would provide a first picture of the empirical research in software maintenance technologies.

The rest of the paper is organized according to these three objectives.

2. Summary of Position Papers

The position papers discuss the authors' experience with evaluating some technology for software maintenance. The following gives a short summary of each position paper in the workshop proceedings.

An assessment of object-oriented design in the maintenance of a large-scale system (Harrison, 1996), presented by Adrian Smith.

The goal of the study is to assess an OO application system with respect to the impact of changes.

The authors analyzed the changes made to a commercial retailing application, with the restriction of having access only to changed lines of code and initial design documentation. The case study covered the work of 14 programmers over more than 20 phases of development. The impact analysis was performed automatically, using PERL scripts, to compute: the impact of each phase of development, differences in impact between more and less abstract classes, differences in impact between business and

technology-changes, and the impact of each type of requirement. Results show that phase impacts were either very high or very low, impact and class abstraction were independent, the impact of business-changes was greater than the impact of technology-changes, and different type of requirements produced different impacts.

Improving the maintenance process: making software more accessible (Kiper, 1996), presented by James Kiper.

Two experiments were presented which compare alternative visual representations of code with respect to comprehension.

The first experiment tested the hypothesis that the graphical representation of decision statements has a more positive effect on the comprehension of decision trees for technical, non-programmers than for programmers. The analysis of the experimental data show that for both programmer and non-programmer subject groups, the average response time for text was less than for graphic.

The second experiment tested the hypothesis that the appropriateness of a graphical form to a particular task can affect response times for comprehension of decision statements. The graphical notation of the previous experiment was improved and compared to the original notation. Results show that there was a significant difference between the two graphical notations, and thus small modifications to graphical forms can produce measurable improvements to comprehension times.

Lessons learned from a regression testing case study (Rosenblum, 1996), presented by David Rosenblum.

The goal of the study is to evaluate a selective regression testing technique with respect to cost-effectiveness.

The study was performed on a sequence of 31 versions of the KornShell (KSH88), a popular UNIX command processor (30 KLOC). For each version, the authors used a test suite of 13-16 shell scripts. The automated regression technique, called TestTube, was applied during the simulation of the development history of KSH88, to select the test cases. Analysis showed that TestTube was not cost-effective for the selected system: 100% of test cases were selected in 80% of versions, and the analysis cost was two orders of magnitude greater than test execution.

The authors conclude that cost-effectiveness is not guaranteed but it depends on the test selection method, the system being tested, the test suite, and test coverage relation.

Experience with regression test selection (Rothermel, 1996), presented by Greg Rothermel.

Three studies were presented with the goal to evaluate another selective regression testing technique, again with respect to cost-effectiveness.

The authors implemented their regression technique as a tool called DejaVu. The first study used a set of 7 small programs with seeded faults. Only a small percentage of the test suite revealed the faults. An average of 44% of tests were selected with a wide range variance (from 43% to 93%) on individual programs and modified versions. Reduction in test size resulted in a reduction of time to run tests. The second study was performed on 5 versions of the main program (50 KLOC) of an internet-based game. The test suite contained 1035 functional tests. DejaVu selected 5% of test cases with a 82% savings in regression testing time. The third study used 9 versions of a small commercial calculator program (2 KLOC) with a real test suite. An average of 33% of test cases were selected. For two versions, no tests were selected meaning that no available test actually executed the modified code.

The authors conclude that regression testing techniques can yield even greater savings when applied to large programs than when applied to small ones. They also recognize that results depend on the structure of programs, the nature of modifications, and test suites.

Greg Rothermel also presented a cluster diagram which models the generalizability of empirical studies without human subjects.

Evaluating impact analysis - A case study (Lindvall, 1996), presented by Mikael Lindvall.

The goal of the study is to evaluate an impact analysis method with respect to the discrepancies between predicted and actual impact of changes.

A long term case study (4 years) at Ericsson Radio Systems AB was presented at the workshop. Evaluation was conducted on a multi-release OO project at several levels of detail. Changes to the source code were analyzed on the C++ class level.

The author showed an example which compared predicted and actual impact of changes at the system release level. The results from the analysis show that the impact analysis method underestimated the number of classes to be changed, although maintainers were very confident in their predictions. Qualitative information was used to correctly distinguish between changed and unchanged classes. This is an example of how qualitative data can be used to avoid mistakes during the evaluation.

Investigating focused techniques for understanding frameworks (Basili, 1996a), presented by Filippo Lanubile.

The goal of the study is to compare a system-wide reading technique with a task-oriented reading technique with respect to ease of OO framework learning and usage.

To compare these two techniques, the authors were undertaking a controlled experiment at the University of Maryland. Graduate students and upper-level undergraduates, working in teams of three people, were required to develop an application task (an OMT object diagram editor) using the GUI application framework ET++. One half of the class was taught the system-wide reading technique and the other half the task-oriented reading technique.

At the date of the presentation, the study had just initiated the operation phase. At the end of the course, the authors would be able to measure team productivity, amount of functionality delivered and accepted, degree of framework reuse, quality of delivered application, and individual comprehension of the framework. Quantitative analysis would be complemented by the qualitative analysis of information collected at different times of the projects.

3. Discussion

The discussion following the presentations of the position papers touched on many issues but centered around the following themes: (1) use of qualitative data, (2) distinction between human-based and non human-based empirical studies, and (3) how to allow for replicability and generalizability.

3.1 Use of Qualitative Data

Everybody in the discussion group agreed that qualitative data can be used to validate the collection of the quantitative data and support the interpretation of the quantitative analysis. However, there was a concern that a study that provides evidence exclusively based on qualitative data could not be considered objective. This concern is encouraged by Kitchenham (1996) who uses quantitative evaluation as a synonym for objective evaluation and qualitative evaluation as a synonym for subjective evaluation. However, this position is not supported by the literature in the other scientific fields (Judd, 1991), (Lee, 1989), (Yin, 1994). Qualitative data can be collected with an equally rigorous process as quantitative data. The ability of expressing a concept in numbers or in verbal propositions influences the way controlled deductions are made: statistics for quantitative analysis and logic for qualitative analysis. The objectivity of empirical research comes from a review process that assures that the analysis relies on

all the relevant evidence and takes into account all the rival interpretations. This can be done (or not done) in both a quantitative analysis and qualitative analysis.

3.2 Human-based vs. Non Human-based Empirical Studies.

The second theme discussed was the distinction between human-based and non human-based empirical studies. There was a debate over whether the differences are essential or incidental. We consider this distinction as essential because of some specific human factors problems that have to be considered: high individual variability, carry-over effects, novelty effects, and expectation effects. These problems influence the validity of human-based experiments but do not apply to non-human-based experiments. A discussion of the role of human factors in software empirical studies can be found in (Brooks, 1980), (Sadler, 1996), (Sheil, 1981).

3.3 Replication

Everybody agreed that isolated, single studies are not as credible as sets of related studies which focus on the same research questions. Another point of general agreement was that the investigators are responsible for making their studies replicatable. A necessary requirement is that the study is carefully documented so that at least its validity can be checked. But investigators can enhance the replicability of studies by also making public their artifacts and their analysis procedures so that the cost of replication for other investigators is held at a minimum. This is not just a wishful thought because laboratory packages are available for some recent experiments (Porter, 1995, Basili, 1996b, Briand, 1996). However, there was a concern about to what degree a study could vary to be considered a replication of a previous one.

The distinction between particularistic and universalistic studies (Judd, 1991) is concerned with the nature of the desired generalization and may be useful to better understand the role of replication in empirical research.

The goal of a *particularistic* study is restricted to a specific target population, and thus there is no interest in replicating such a study in different contexts. The external validity of a particularistic study is achieved by ensuring that the sample is representative of the target population, for example conducting the study in a real life setting or using procedures of random sampling for selecting a survey's subjects. If the analysis is based on quantitative data, investigators can internally replicate the original study to collect more observations and thus increase the statistical power of tests.

On the other hand, a *universalistic* study is conducted to test hypotheses derived from theories (for technology evaluation, the theory describes and predicts the effect of a technology). Looking at the results of a universalistic study, the question is «Does the study support or deny the initial hypothesis?» The ability to generalize is a property of the theory not of the results, i.e., we want to know if the predictions apply to contexts that the theory does not specifically address and that have not been tested yet. Given a universalistic study, any other study that tests the same hypotheses, can be considered a replication, whatever its method, design, or setting. If the hypotheses are denied, then the theory has failed to account for the different variables of the new study, and then it must be updated in an iterative learning process. On the contrary, if the hypotheses are confirmed, then the theory has survived another probe, and thus we have more confidence in its predictions (Campbell, 1963).

4. Classification of the Evaluation Studies

In this section we classify the six position papers according to the multidimensional framework shown in Table 1.

The first dimension, the *object* of study, is the thing being investigated (e.g., for a study of technology evaluation, the object is the technology itself). However, we may distinguish between a product and a process technology. By *product technology* we mean any software application, including applications that perform a specific function directly for the user, infrastructure components that provide services to an application, and tools that automate some software related activity. On the other hand, a *process technology* is any practice, technique, or method that guides some software development or maintenance task.

CLASSIFICATION DIMENSIONS	CLASSIFICATION ATTRIBUTES
<i>object</i>	<i>product technology vs. process technology</i>
<i>purpose</i>	<i>outcome evaluation vs. process evaluation</i>
<i>focus</i>	<i>single/specific vs. multiple/generic</i>
<i>empirical method</i>	<i>experiment, quasi-experiment, case study, survey</i>
<i>study setting</i>	<i>laboratory vs. real life</i>
<i>sources of evidence</i>	<i>human participation vs. no human participation</i>

<i>type of empirical evidence</i>	<i>quantitative analysis, qualitative analysis, combination</i>
<i>extrapolation</i>	<i>particularistic vs. universalistic</i>

Table 1. The classification framework

The second dimension, *purpose*, is the reason for which the evaluation is being performed. Social scientists distinguish between two kinds of evaluation purposes: outcome and process evaluation (Judd, 1991). An *outcome evaluation* investigates the effect of a technology by asking «Does it work?» after it has been used long enough to have produced some measurable effects. The results are used to decide if it is worthwhile to keep or change a technology. On the other hand, a *process evaluation* studies the effects of a technology by asking «How does it work?». Evaluation is conducted since the technology has begun to be used. The results are used to provide feedback about how the technology and its use can be improved.

The third dimension is the *focus* of the evaluation study, i.e., the effect of the technology that is going to be observed such as cost, time, errors, or changes. An evaluation study might have only a *single/specific* focus, e.g., effort, for which the decomposition onto variables is straightforward. On the other hand, when the preexisting knowledge is poor, an evaluation study might have a *multiple/generic* focus, such as effectiveness or a list of criteria. This is typical of exploratory studies that look to many directions because there is a weak background theory or no other studies to support any expectation on the outcome of the evaluation.

The fourth dimension, the *empirical method*, spans a spectrum of different ways to collect and analyze empirical evidence, each with its own logic. At a high level, an empirical study of technology evaluation can almost always be classified as an *experiment* (also called controlled experiment or randomized experiment), a *quasi-experiment*, a *survey* (also called a correlational study), or a *case study* (also called field study).

The fifth dimension, *study setting*, is the context in which people or artifacts participate as subjects in the investigation. We distinguish between laboratory and real life settings (Judd, 1991). A *laboratory* setting allows the researcher to attain control over the extraneous variables, manipulate the independent variables, and adapt the setting to the specific study goal. On the contrary, *real life* settings have to be considered when the evaluation has a particularistic purpose or the time frames to observe the effects of the technology are too long for a laboratory.

The sixth dimension, *sources of evidence*, describes the sources of information from which data are gathered and evidence is built. Here, we distinguish between sources of evidence that require *human participation*, and sources of evidence with *no human participation* (other than the investigators, of course). The distinction is relevant for the classification because when human subjects are involved in an evaluation study, there may be human factors issues (high individual variability, carry-over effects, novelty effects, and expectation effects) that can bias the evaluation.

The seventh dimension, *type of empirical evidence*, includes evidence based on quantitative analysis, evidence based on qualitative analysis, or evidence based on a combination of both quantitative and qualitative analysis. A *quantitative analysis* is based on information represented in numerical form. A *qualitative analysis* is based on verbal or textual information usually derived from interviews and direct observation. Qualitative analysis when used in *combination* with quantitative analysis helps to ensure conformance to the study procedures and to interpret the results, especially unexpected results.

The last dimension, *extrapolation*, is concerned with the nature of the desired generalization. Social scientists (Judd, 1991) distinguish between a particularistic and a universalistic research. Here, we apply this distinction to technology evaluation. A *particularistic* evaluation investigates the effects of a technology for a specific target population, which is specified in the goal. Since the context of the evaluation is well defined, the key concern is that the study is as close as possible to the real conditions in which the technology will be used. A research question such as «What is the effect of this technology in organization XYZ?» is typically addressed with a particularistic evaluation. Because of this unique interest in a particular context, extrapolation and replications of the results across multiple settings and populations are of minor interest. On the other hand, a *universalistic* evaluation investigates a theoretical proposition about the relationship between specified variables (here, involving a technology). The description of the conditions under which the predictions hold is part of the theory. For those conditions which are not specified by the theory, it is assumed that the study sample is representative of all the world, and thus there is not a specific environment as the focus of interest. It is reasonable to extrapolate the results from the specific setting where the experiment has been performed to others, unless they are specifically excluded by the theory. Nonetheless, we can increase our confidence that the hypothesized relationship actually exists by replication, i.e., by trying to reproduce the findings using different settings and populations. A research question such as «what are the best circumstances for a given technology?» must be answered with a universalistic evaluation.

Tables 2, 3, and 4 classify the maintenance studies with respect to the dimensions of the framework. Although the sample is small and not all the evaluation studies were completed, the classification can be used as an example of framework application. The classification also provides a synthesis, albeit very partial, of the state of the research in the empirical evaluation of software maintenance technologies. Readers can review the classification by comparing the papers in the workshop proceedings with the definitions given in the previous section.

We provide the rationale for some choices.

The papers with multiple studies (Kiper, 1996), (Rothermel, 1996), have been abstracted as if they were only one, to be fair with respect to the other workshop's participants. The classification captures the common aspects of these aggregated studies.

Study	Object	Purpose	Focus
(Harrison, 1996)	product technology	outcome evaluation	single/specific
(Kiper, 1996)	product technology	outcome evaluation	single/specific
(Rosenblum, 1996)	product technology	outcome evaluation	single/specific
(Rothermel, 1996)	product technology	outcome evaluation	single/specific
(Lindvall, 1996)	process technology	outcome evaluation	single/specific
(Basili, 1996a)	process technology	outcome evaluation	multiple/generic

Table 2. Classification with respect to object, purpose, and focus

Study	Empirical Method	Study Setting	Sources of Evidence
(Harrison, 1996)	case study	real life	no human participation
(Kiper, 1996)	experiment	laboratory	human participation
(Rosenblum, 1996)	experiment	laboratory	no human participation
(Rothermel, 1996)	experiment	laboratory	no human participation
(Lindvall, 1996)	case study	real life	human participation
(Basili, 1996a)	experiment	laboratory	human participation

Table 3. Classification wrt empirical method, study setting, and sources of evidence

Study	Type of Empirical Evidence	Extrapolation
(Harrison, 1996)	quantitative analysis	particularistic
(Kiper, 1996)	quantitative analysis	universalistic
(Rosenblum, 1996)	quantitative analysis	universalistic
(Rothermel, 1996)	quantitative analysis	universalistic
(Lindvall, 1996)	combination of quantitative and qualitative analysis	particularistic
(Basili, 1996a)	combination of quantitative and qualitative analysis	universalistic

Table 4. Classification with respect to type of empirical evidence and extrapolation

We have classified the empirical method of (Rosenblum, 1996) as an experiment although the authors uses the term «case study» in the paper. Although the authors use 31 real versions of the popular (in the Unix world) KornShell command processor, the test suite is artificial and thus the study is a simulation of the real evolution. The artificiality of the study does not fit with the definition of case study, which focuses on real events. The full control of the independent variables (to use or not the selective regression testing technique) allows us to classify the study as an experiment. In this case, as well as for (Rothermel, 1996), the control group is represented by not applying the regression technique, i.e., the retest-all technique.

Both these experiments on regression testing have treatments that vary within subjects (the programs or versions tested), i.e., each subject is measured under different treatment conditions. It is noteworthy that for a within-subject experiment, randomization should be applied by randomly determining the order in which any subject is exposed to the two treatments (Judd, 1991). However, it seems in this case that randomization is not necessary for ensuring full internal validity. The two studies would be classified as quasi-experiments only if there were reasonable rival hypotheses as a consequence of the lack of randomization.

Since the sample is very small and some studies were still in progress, we can only note the absence of studies with the purpose of process evaluation, i.e., studies which address the question «How does it work?». This lack seems in contrast with the exploratory nature of some of the studies.

5. Conclusions

The working group revealed a significant community of interest among the participants. A good remark from one participant was that evaluation studies are a great occasion for researchers to work in synergy with practitioners who cannot conduct rigorous studies but need real facts.

However, in software engineering and more generally in computer science, the balance between evaluation of results and development of new theories or technologies is still skewed in favor of the unverified proposals (Glass, 1995), (Tichy, 1995). As a result empirical research may appear in the software engineering field as a side topic instead of being a standard way to validate claims, as in other scientific disciplines. In part, this happens because empirical research in software engineering borrows methods from both the social and physical sciences (depending on whether humans are involved or not), thus adding more complexity and effort to a type of work that does not give easy rewards.

The multiple dimensions of the framework reveal how complex it is to conduct an empirical study for evaluation purposes. Investigators are always challenged to design the best study which the circumstances make possible, trying to rule out all the alternative explanations of the results. Although the investigators may not get the «perfect» study (assuming there is a perfect one), they are aware of the biases which can make the conclusions equivocal. It is this kind of awareness that makes the difference between an empirical and a non-empirical evaluation.

Acknowledgments

Thanks to all the participants in the WESS '96 working group «Study and assessment of (new) technologies.» Thanks also to Forrest Shull for having taken notes during the workshop, and Carolyn Seaman for having improved a draft version of this paper.

References

- (Basili, 1996a) Basili, V. R., Caldiera, G. , Lanubile, F., and Shull, F., 1996. Investigating focused techniques for understanding frameworks. WESS '96, *Proc. Int. Workshop on Empirical Studies of Software Maintenance*.
- (Basili, 1996b) Basili, V. R., Green, S., Laitenberger, O., Lanubile, F., Shull, F., Sorumgard, S., and Zelkowitz, M., 1996. Packaging researcher experience to assist replication of experiments. *Proc. ISERN Meeting*, 3-6.

- (Briand, 1996) Briand, L., Bunse, C., Daly, J., and Differding, C., 1996. An experimental comparison of the maintainability of object-oriented and structured design documents. *Technical Report ISERN-96-13*, International Software Engineering Research Network.
- (Brooks, 1980) Brooks, R. E., 1980. Studying programmer behavior experimentally: the problems of proper methodology. *Communications of the ACM*, 23:4, 207-213.
- (Campbell, 1963) Campbell, D. T., and Stanley, J. C., 1963. *Experimental and Quasi-Experimental Designs for Research*. Boston: Houghton Mifflin Co.
- (Glass, 1995) Glass, R. L., 1995. A structure-based critique of contemporary computing research. *The Journal of Systems and Software*, 28, 3-7.
- (Harrison, 1996) Harrison, R., and Smith, A., 1996. An assessment of object-oriented design in the maintenance of a large-scale system, WESS '96, *Proc. Int. Workshop on Empirical Studies of Software Maintenance*.
- (Kitchenham, 1996) Kitchenham, B. A., 1996. Evaluating software engineering methods and tool - Part 1: The evaluation context and evaluation methods. *ACM SIGSOFT Software Engineering Notes*, 21:1, 11-15.
- (Judd, 1991) Judd, C. M., Smith, E. R., and Kidder, L. H., 1991. *Research Methods in Social Relations*, 6th edition. Orlando: Holt Rinehart and Winston, Inc.
- (Kiper, 1996) Kiper, J., Auernheimer, B., and Ames, C., 1996. Improving the maintenance process: making software more accessible, , WESS '96, *Proc. Int. Workshop on Empirical Studies of Software Maintenance*.
- (Lee, 1989) Lee, A. S., 1989. A scientific methodology for MIS case studies. *MIS Quarterly*, March, 33-49.
- (Lindvall, 1996) Lindvall, M., 1996. Evaluating impact analysis - A case study, WESS '96, *Proc. Int. Workshop on Empirical Studies of Software Maintenance*.
- (Porter, 1995) Porter, A. A., Votta, L. G., and Basili, V. R., 1995. Comparing detection methods for software requirements inspections: a replicated experiment. *IEEE Transactions on Software Engineering*, 21:6, 563-575.
- (Rosenblum, 1996) Rosenblum, D., and Weyuker, E., 1996. Lessons learned from a regression testing case study, WESS '96, *Proc. Int. Workshop on Empirical Studies of Software Maintenance*.
- (Rothermel, 1996) Rothermel, G., and Harrold, M. J., 1996. Experience with regression test selection, WESS '96, *Proc. Int. Workshop on Empirical Studies of Software Maintenance*.
- (Sadler, 1996) Sadler, C., and Kitchenham, B. A., 1996. Evaluating software engineering methods and tool - Part 4: The influence of human factors. *ACM SIGSOFT Software Engineering Notes*, 21:5, 11-13.
- (Sheil, 1981) Sheil, B. A., 1981. The psychological study of programming. *ACM Computing Surveys*, 13:1, 101-120.

- (Tichy, 1981) Tichy, W. F., Lukowicz, P., Prechelt, L., and Heinz, E. A., 1995. Experimental evaluation in computer science: a quantitative study. *The Journal of Systems and Software*, 28, 9-18.
- (Yin, 1994) Yin, R. K., 1994. *Case Study Research: Design and Methods*, 2nd ed., Thousand Oaks: SAGE Publications.