# Status of Empirical Research in Software Engineering

Andreas Höfer, Walter F. Tichy

Fakultät für Informatik, Universität Karlsruhe,
Am Fasanengarten 5, 76131 Karlsruhe, Germany
{ahoefer|tichy}@ipd.uni-karlsruhe.de

**Abstract.** We provide an assessment of the status of empirical software research by analyzing all refereed articles that appeared in the Journal of Empirical Software Engineering from its first issue in January 1996 through June 2006. The journal publishes empirical software research exclusively and it is the only journal to do so. The main findings are: 1. The dominant empirical methods are experiments and case studies. Other methods (correlational studies, meta analysis, surveys, descriptive approaches, ex post facto studies) occur infrequently; long-term studies are missing. About a quarter of the experiments are replications. 2. Professionals are used somewhat more frequently than students as subjects. 3. The dominant topics studied are measurement/metrics and tools/methods/frameworks. Metrics research is dominated by correlational and case studies without any experiments. 4. Important topics are underrepresented or absent, for example: programming languages, model driven development, formal methods, and others. The narrow focus on a few empirically researched topics is in contrast to the broad scope of software research.

## 1 Introduction

During the 10½ years that have elapsed since the first issue of Empirical Software Engineering (ESE) appeared in January 1996, the journal has become the major venue for publishing empirical results in software research. It is the only journal exclusively dedicated to empirical studies in software. Thus, ESE can be seen as a good indicator for the status and health of empirical software research. We wanted to know what topics are addressed by empirical research, which research methods are used, and where the data comes from. Further, we were interested in the question whether there are important topics that are insufficiently covered by empirical research. To answer these questions, we performed an in-depth bibliographic study of all reviewed articles in ESE from volume 1, number 1 to volume 11, number 2.

## 2 Related Work

In 2005 Segal et al. [4] presented a study that investigated the nature of the empirical evidence reported in 119 papers which appeared in ESE between 1997 and 2003. The

classification scheme used in this paper is based on the one developed by Glass et al. [2]. Segal et al. [4] found among other things, that about half of the papers focused on measurement/metrics and inspections/reviews, that authors were almost as interested in formulating as in evaluating, and that other disciplines are referenced rarely.

The classification scheme introduced by Glass et al. [2] differentiates papers in the field of computing based on five characteristics: topic, research approach, research method, reference discipline, and level of analysis. Glass and his colleagues applied the scheme to 369 articles published in six leading software engineering journals over the period from 1995 to 1999. They conclude that software engineering research is diverse in topic but narrow in its research approach and method. Glass also found that 98 % of the papers examined do not reference another discipline.

Zelkowitz and Wallace [6] define a taxonomy for the classification of papers within the field of software engineering. They classified 612 articles published during the years 1985, 1990, and 1995 in the journals IEEE Transactions on Software Engineering and IEEE Software as well as in the proceedings of the International Conference on Software Engineering. One of their findings is that about 30 % of all classified papers lack experimental validation, but note that this situation is improving.

Sjøberg et al. [5] selected controlled experiments from 5,434 articles published in nine journals (including ESE) and proceedings of three conferences. The 103 papers describing controlled experiments were characterized according to topic, subjects, tasks, and environment of the experiment. One of the main results of Sjøberg and his co-authors is that controlled experiments constitute only a small fraction (1.9 %) of articles published.

Lukowicz et al. [3] compare the percentage of papers with experimental validation in several computer science journals and conference proceedings to the percentage of experimental work in the two journals Neural Computation and Optical Engineering. The findings of this study, which classified 403 articles, indicate that there is a lack of empirical validation in computer science.

The present paper concentrates on empirical work in software engineering in the journal dedicated to this type of work and attempts to get an indication of research quality and breadth. It is closest to the work of Segal et al. [4], but surveys a longer time span, classifies research method according to accepted categories in psychological research, and divides the largest of the categories in the work by Segal et al. [4], software life-cycle, into subcategories. We also identify gaps in the coverage of research topics.

# 3 Research Method

## 3.1 Selection of the Articles

We gathered all issues of ESE from January 1996 to June 2006 and selected all reviewed articles. Titles, authors, and keywords of those papers were entered into a

table for classification. We deliberately excluded 50 editorials, 30 viewpoints/position papers, 15 conference and workshop reports, and 6 comments/correspondence papers from the literature analysis. In total, we selected 133 reviewed articles.

**Table 1.** Topics.

| Topic |
|---|
| Design/Architecture |
| Diagrams/Notations |
| Empirical methods |
| Inspections/Reviews |
| Maintenance |
| Measurement/Metrics |
| Project planning/Estimation |
| Quality estimation/Fault prediction |
| Requirements |
| Software engineering process |
| Testing |
| Tools/Methods/Frameworks |
| Other |

### 3.2 Classification of the Articles

In order to develop a classification scheme for the articles, the authors jointly studied titles, keywords, and abstracts of all the articles that appeared in the first year of ESE. Out of this study, a first version of the classification scheme was developed. This scheme was refined during the classification process. Each paper was classified according to the three dimensions topic, method, and source of data.

- **Topic**: The subject area of the paper within software engineering. Table 1 provides the list of topics. The categories are self-explanatory, except for the following:
  - The category Empirical methods covers tools or approaches to conduct empirical work; such papers aim to improve research methods.
  - There are categories for all major phases in software development (Design/Architecture, Inspections/Reviews, Maintenance, etc.), except implementation (this class is empty). The class Software engineering process includes papers that address more than one phase (usually the overall software development process).
  - The class Tools/Methods/Frameworks covers papers that introduce a novel tool, method or framework for software development, coupled with an empirical study (typically a case study).
- **Method:** The empirical research method used for the study. We use categories from psychological research according to Christensen [1]. We only present

non-empty categories[1]: case study, correlational study, ethnography, experiment, ex post facto study, meta analysis, phenomenology, survey (see Table 2). Papers were classified according to the main method. For example, if a paper contains a survey as a preliminary step for an experiment, then it would be classified as experiment

- **Source of data:** This characteristic categorizes the origin of the data used for empirical research (see Table 3).

The following topics were sub-classified: Empirical method, Measurement/Metrics, and Tools/Methods/Frameworks. The reason is that papers in these classes typically address an additional topic. For instance, an empirical method might be specific to project planning, a metrics paper might apply to fault prediction, or a tool might be specific to the topic Requirements. Instead of double classification (which would be the alternative), we show the subcategories separately, in order to make the distribution of topics more transparent.

The classification process worked as follows. The first author initially classified all papers. If title, keywords, abstracts, and conclusions were not sufficient for classification, the whole article was studied. Doubtful assignments were tagged for the second author. After the first author had classified all articles, the second author checked the classification table for plausibility, spot-checked classifications in detail, and tagged additional doubtful classifications. The tagged classifications were then re-checked together and corrected if necessary.

**Table 2.** Research Method.

| Method | Definition |
|---|---|
| Case study | In-depth analysis of a particular project, event, organization, etc. |
| Correlational study | Measuring variables and determining the degree of relationship that exists between them. |
| Ethnography | Description and interpretation of the culture of a group of people. |
| Ex post facto study | Study in which the variables of interest are not subject to direct manipulation, but must be chosen after the fact (e.g., when analyzing software repositories). |
| Experiment | Quantitative study to test cause-and-effect relationships. |
| Meta analysis | Integrates and/or describes the results of several studies. |
| Phenomenology | Description of an individual's or a group's experience of a phenomenon. |
| Survey | Data is collected by interviewing a representative sample of some population. |

[1] Empty categories are: Longitudinal and cross-sectional study, naturalistic observation.

**Table 3.** Sources of Data.

| Source | Definition |
|---|---|
| Professionals | Data acquired from professionals directly by using them as subjects in an experiment or indirectly by collecting data from projects with professionals. |
| Students | Data acquired from students directly by using them as subjects in an experiment or indirectly by collecting data from a project with students. |
| Both | Data acquired from students and professionals. |
| Benchmarks | Benchmarks are artificially composed data designed to measure the performance of a tool, method, algorithm, etc. |
| Software | The source Software refers to data derived from operational software (such as reliability data) irrespective of the methods of development for such software. |
| Studies | Data acquired from other studies (meta analysis). |
| Unknown | Unstated source of data. Some articles do not state how the data was gathered or whether their subjects were students or professionals. |

# 4 Findings

## 4.1 Topic

Figure 1 depicts the distribution of topics. This dimension is dominated by the categories Measurement/Metrics and Tools/Methods/Frameworks followed by Inspections/Reviews and Software engineering process. The rest are all below 10 %. The categories Usability and Reliability were under 2 %, so we combined them with the papers that did not fit any category (class Other).

As mentioned, several categories have subtopics, which are not included in Figure 1. Of the 22 papers in the Measurement/Metrics category, half dealt with Project planning/Estimation and 27.3 % with Quality estimation/Fault prediction. Other topics are each under 5 %.

There are 20 Tools/Methods/Frameworks papers, but the topics are more spread out: 25.0 % Software engineering process, 20.0 % Quality estimation/Fault prediction, 15.0 % Project planning/Estimation, and 10.0 % Usability. The class Empirical methods contains 11 papers, with 36.4 % General (no particular topic) and 27.3 % dealing with project planning.

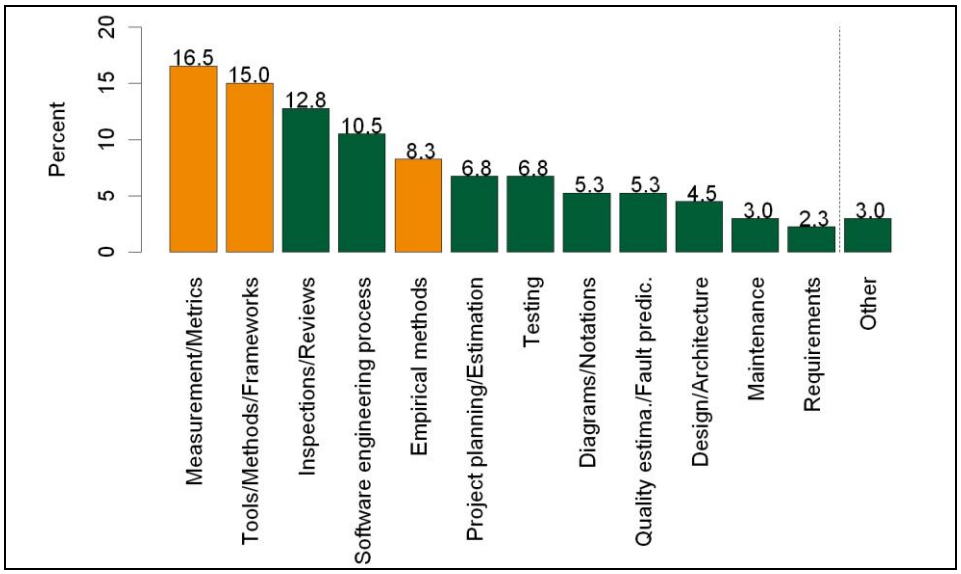


**Fig. 1.** Topic (Categories with subtopics are highlighted in orange.)

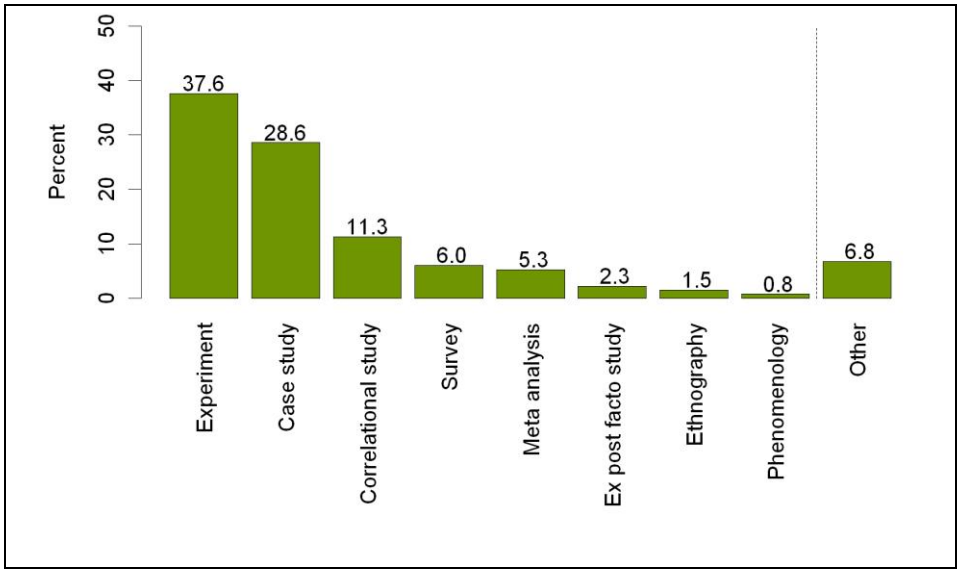


**Fig. 2.** Research Method

### 4.2 Research Method of the papers surveyed

The preferred research methods are Experiment and Case study (see Figure 2). Among the 50 papers describing an experiment 13 (26.0 %) were replications.

An interesting question is what methods were used in the top three topics. Among the 22 Measurement/Metrics papers, 36.4 % use correlational studies and 31.8 % case studies; there are no experiments and thus no systematic inquiries into cause and effect. For Tools/Methods/Frameworks, 55.0 % of 20 papers employ case studies, and 25.0 % experiments. Of the 17 articles with the topic Inspections/Reviews, 15 (88.2 %) use experiments, the remaining two papers contain case studies. Studies of Inspections/Reviews have the largest number of experiments. Diagrams/Notations is next with 7, followed by Design/Architecture with 6, and Project planning/Estimation as well as Tools/Methods/Frameworks each with 5. The high proportion of Inspections/Reviews combined with a high rate of experiments reflects the maturity of this area, as researchers are exploring causal relationships.

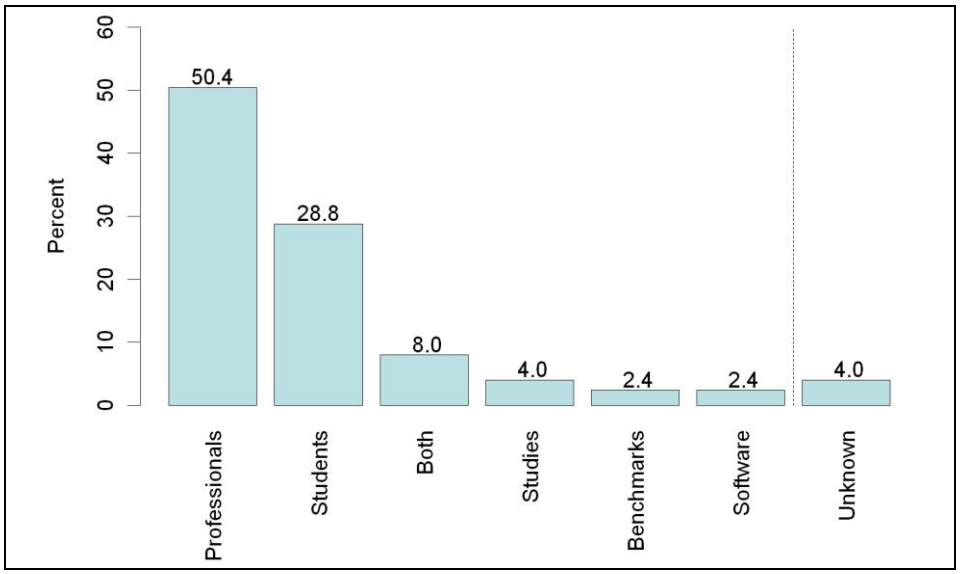


**Fig. 3.** Source of Data

### 4.3 Source of Data

Figure 3 shows the source of data. Papers employing professionals and students dominate. In 63 publications, professionals only were used, and solely students in 36. There were 10 papers using both, for example comparing professionals and students.

Figure 4 shows a cumulative graph of the distribution of papers with professional and student subjects. Though the proportion of papers with students and professional subjects varies from year to year, it can be seen that cumulatively, articles using professionals outnumber those using students over the years.

As empirical work is often criticized for relying on students, we looked at the data source with respect to research method. It turns out that 78.9 % of case studies used professionals, 5.3 % students, and 2.6 % both. The situation is nearly reversed for experiments: 60.0 % used students, 22.0 % professionals, and 14.0 % used both students and professionals (see also Table 4). These findings are in line with those of Sjøberg et al. [5]. On a much larger sample of experiments, Sjøberg and his co-authors report that 72.6 % of experiments employed students, 18.6 % professionals and 8.0 % both.

**Table 4.** Proportion of Professionals and Students in the Top Three Research Methods

| Type of Study | % (Number of Papers) | | |
|---|---|---|---|
| | **Professionals** | **Students** | **Both** |
| Experiment | 22.0 (11) | 60.0 (30) | 14.0 (7) |
| Case study | 78.9 (30) | 5.3 (2) | 2.6 (1) |
| Correlational study | 66.7 (10) | 13.3 (2) | 13.3 (2) |
| All types | 50.4 (63) | 28.8 (36) | 8.0 (10) |

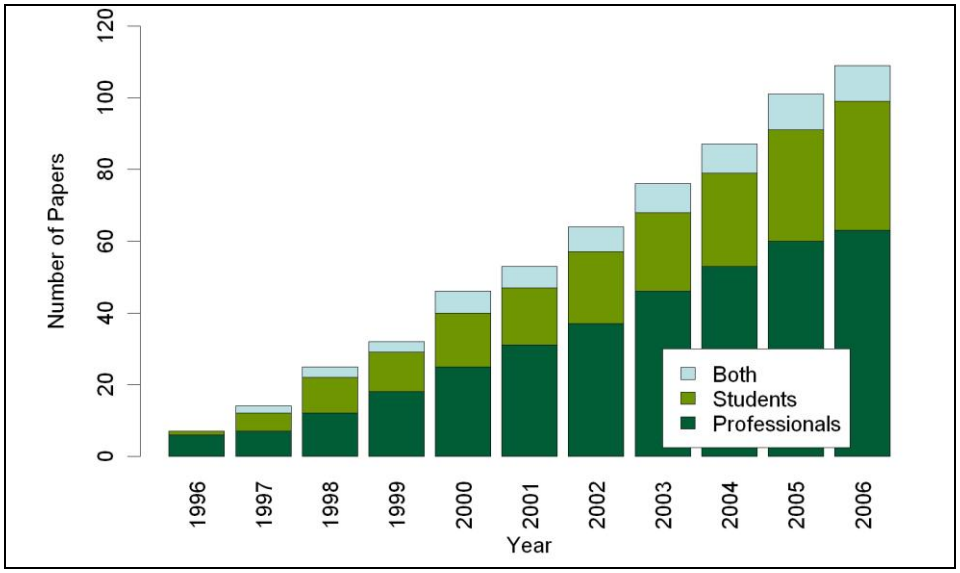


**Fig. 4.** Data from Students and Professionals

## 5 What is Missing?

Overall, it is a positive sign that studies with professionals outnumber those with students by a healthy margin. However, in experiments, student subjects dominate. This situation may reflect the difficulties of conducting controlled experiments outside a laboratory. More effort should be expended to repeat important experiments with professionals in order to improve generalizability of the results.

The Measurement/Metrics area is dominated by case studies and correlational studies, without any experiment. The lack of research into cause and effect seems to be a major weakness. It is well known that a correlation between two variables does not constitute a causal relationship; the values of both of these variables may be determined by other, hidden variables. There is strong evidence that causal relationships have not been identified: It is straight-forward for programmers to corrupt the indicator variables used today and thereby subvert any prediction based on them. By contrast, in the software inspections area, which is about as old as the metrics area, researchers have developed experimental techniques to successfully explore causal relationships.

Overall, the range of software topics studied empirically is rather narrow. Some important topics are missing completely. In particular, studies about programming languages and programming paradigms are conspicuously absent. As these topics are obviously important and subject to intense debate, studies comparing imperative vs. functional vs. scripting vs. object oriented languages are urgently needed, to inform further development of these languages and enable rational choices. Also missing are studies that compare programming approaches with standardized software that substitute customization for programming. Program verification is not represented, but if verification is a practical approach, even in a limited domain, empirical studies are needed to determine efficacy. Absent were articles covering recent areas such as model driven development or aspect orientation. Furthermore, we expected to find papers illuminating the relationship between developer's personal characteristics and their optimal mode of work. Such studies would require collaboration with other disciplines such as social sciences and psychology, but references from software engineering to these disciplines are rare, as observed by Glass et al. [2] and Segal et al. [4]. Long-term studies of programming methods, such as agile methods were missing, too. Large gaps as to topic are confirmed by Sjøberg et al. [5] and Segal et al [4].

A discussion with participants of the Dagstuhl seminar brought up additional topics that are missing. Unclear are the feasibility bounds of techniques—i.e., determining in what situation or in what context a particular method or approach is preferable to another. Closely related are cost/benefit tradeoffs covering the development cycle, for example models for determining the relative effort to be spent on requirements, design, quality assurance, and so on. In other words, what is needed is an answer to the question of what has to be done when, and how much of it. A unifying theory about defect causation and detection would help guide quality assurance efforts. Finally, the grand challenge for software research was seen as developing an

understanding of which software methods work and why. Such an understanding should provide a suitable foundation for predictable software processes and products.

## 6 Threats to Validity

The first threat to validity concerns the fact that articles reporting on empirical work are published in other venues as well. Thus, ESE might not provide a representative sample of all empirical research. But Sjøberg et al. [5] confirm some of our findings on a larger sample, restricted to controlled experiments.

We guarded against classification errors by a careful definition of classes and a cross check by a second person as described in section 3.2. Nevertheless, there are some borderline cases, and other raters might classify differently.

## 7 Conclusions

We conducted a literature review of all refereed articles published in ESE within the period form January 1996 to June 2006. We found that the use of professionals in 78.9 % of case studies is encouraging, while controlled experiments are predominantly conducted with students. The range of topics continues to be narrow and should be broadened considerably. The metrics area would benefit from emphasizing investigations into cause-and-effect relationships. The area of inspections and reviews appears to be methodologically mature with a high proportion of experiments.

## 8 References

1. L. B. Christensen, Experimental Methodology, 8th Edition, Allyn and Bacon (2001).
2. R. L. Glass, I. Vessey, V. Ramesh. Research in Software Engineering: An Analysis of the Literature. Journal of Information and Software Technology, vol. 44, no. 8, pp. 491-506 (2002).
3. P. Lukowicz, E. A. Heinz, L. Prechelt, W. F. Tichy. Experimental Evaluation in Computer Science: A Quantitative Study. Journal of Systems and Software, vol. 28, no. 1, pp. 9-18 (1995).
4. J. Segal, A. Grinyer, H. Sharp. The type of evidence produced by empirical software engineers. REBSE '05: Proceedings of the 2005 workshop on Realising evidence-based software engineering, pp. 1-4 (2005).
5. D. I. K. Sjøberg, J. E. Hannay, O. Hansen, V. B. Kampenes, A. Karahasanović, N-K Liborg, A. C. Rekdal. A Survey of Controlled Experiments in Software Engineering. IEEE Transactions on Software Engineering, vol. 31, no. 9, pp. 733-753 (2005).
6. M.V. Zelkowitz, D. Wallace. Experimental Validation in Software Engineering, Journal of Information and Software Technology, vol. 39, pp. 735-743 (1997).