# Rethinking Replication in Software Engineering: Can We See the Forest for the Trees?

Mika V. Mäntylä
Aalto University, SoberIT
P.O. Box 19210,
FI-00760 Aalto,
Finland
+358 50 577 1684
mika.mantyla@tkk.fi

Casper Lassenius
Aalto University, SoberIT & MIT Sloan
Center for Collective Intelligence
MIT Building NE25
Cambridge MA, 02142 USA
+1 (617) 253 4927
casperl@mit.edu

Jari Vanhanen
Aalto University, SoberIT
P.O. Box 19210,
FI-00760 Aalto,
Finland
-
jari.vanhanen@tkk.fi

## ABSTRACT
In this paper, we argue that the concept of replication of empirical studies in software engineering should be understood more broadly than it currently is. In particular, the replication of case studies and surveys as a way of validating and extending theories should be incorporated in the mainstream view of replication, which at present is mostly focused on controlled experiments. A small-sample study of papers in IEEE Transactions on Software Engineering shows that about 10% of studies published in 2009 can be considered replications. However, none of these was self-labeled as replication. We think that the authors believed that labeling their work as replication might decrease its value in the eyes of reviewers and editors. We conclude that there is no acute shortage of replication studies in software engineering if taking a broader viewpoint to replication, but the definition and valuation of 'replication studies' need to be re-evaluated in the software engineering community.

## Categories and Subject Descriptors
D.0 [General]

## General Terms
Experimentation

## Keywords
Replication, Case Study, Survey, Action Research, Experiment, Methodology

## 1. INTRODUCTION
The importance of replication as a means of creating and deepening scientific knowledge has been recognized throughout the natural, social and the engineering sciences. It should be difficult to find any serious scientist who does not consider replications an essential part of scientific work. Popper [27] states:

*We do not take even our own observations quite seriously, or accept them as scientific observations, until we have repeated and tested them. Only by such repetitions can we convince ourselves that we are not dealing with a mere isolated "coincidence," but with events which, on account of their regularity and reproducibility, are in principle intersubjectively testable.*

In the past, many methodological papers discussing empirical software engineering actually focus mainly on the use of controlled experiments, only mentioning other empirical approaches as side notes [17, 32, 42]. However, during recent years there has been a growing interest in other research methods as well [18, 26, 30, 35]. Nevertheless, the strong role of experiments in empirical software engineering undoubtedly explains why the importance of replication has been acknowledged mainly in relation to controlled experiments [4, 6, 14, 32, 41]. This currently dominant view is exemplified, e.g. by the following citation [7]:

*"We conclude that there exists only one route for empirical software engineering to follow: to make available laboratory packages of experimental materials to facilitate internal and external replications, especially the latter, which have greater confirming power."*

However, to advance software engineering as a science, it is necessary to take a broader view of replication, including its use for other empirical methods, such as case studies and surveys. This question has received considerably less attention, although there should be no doubt of the value of using a replication logic also for these kinds of studies. Indeed, we think that many studies that actually can be considered replications are not labeled as such, perhaps due to a bias against publishing replication studies in major software engineering journals.

The lower emphasis of other empirical approaches in comparison to experiments might be due to the seemingly less rigorous research methodology in surveys and case studies, which from a positivistic viewpoint and guided by the natural sciences, make them look "less scientific". However, important theories can and should also be created by investigating a phenomenon within its real world context. For this, case studies are the preferred approach [43]. Indeed, many studies validating their constructs used case studies as their main empirical approach. Recently published guidelines for conducting case studies in the software engineering context might help make case studies more accepted and better performed in our community in the future [30]. Of particular interest from the point of view of creating scientific knowledge are case studies using a replication logic [43], as well as case studies investigating similar phenomena as earlier ones, building on their results.

With the given background, we think that it is important to consider replication of all kinds of empirical software engineering studies. It is clear that the replication of surveys and case studies must be treated differently than the replication of controlled experiments. However, the replication of case studies is particularly important as they

often lack quantitative methods, and thus cannot back up their results with statistical power.

This paper has two main goals: first, to argue for a wider sense of what should be accounted as replication in the context of empirical software engineering, and second: to study what types of replication studies have been successful in getting published in top-level software engineering journals. At this point, we attempt mainly to provide a basis for discussion and some preliminary validation of our ideas. In the future, we hope to extend the discussion and the empirical part of this work in a subsequent paper.

This paper is structured as follows: the next section discusses previous work on replication in software engineering, as well as provides a short discussion of replication in the social sciences. Section 3 presents our research methodology, and Section 4 our results. This is followed by a discussion in Section 5, and Conlusions in Section 6.

# 2. RELATED WORK

## 2.1 Replication in software engineering

A survey of controlled software engineering experiments by Sjoberg et al. [36] found that of the 5435 scientific articles published in 12 leading software engineering journals or conferences from 1993 to 2002 only 20 were replication studies. According to [22] replications can either be *close* meaning that one aims to replicate as closely as possible, or *differentiated* meaning variations in vital experiment variables. Of the 20 replication studies, five were close replications, and 15 were differentiated. All close replications confirmed the findings of the original studies, while of the 15 differentiated replications, six reported results differing from the original experiment. Eleven of the 20 replication studies were conducted by the original authors and nine by other scientists.

The value of replication in software engineering has been recognized, as there are several papers published in the field that explicitly addresses the issue and attempts to provide strategies and guidelines for its use. Shull et al. [32] identify two types of replications *exact* and *conceptual* replications. Exact replications focus on following the original study procedures as closely as possible, whereas conceptual replications are studying merely the same research question. The researchers point out that conceptual replications of realistic experiments are much more expensive than exact replications, which makes the latter more feasible. Furthermore, they argue that conceptual replications offer little possibility for comparison when there are different results due to the different procedures used. The positive note on conceptual replication is that when the results hold it offers stronger evidence than exact replication as the phenomena is then more likely to hold in more fluctuating context.

Kitchenham [16] disagrees with Shull et al. and argues that there is a danger in focusing solely on identical replications as the industrial impact is weaker and due to the risk of reusing and reinforcing possible flaws in the original experiment design (that in turn would be present in the replications as well). Therefore, it might be wiser not to utilize exact replications

Juristo and Vegas [14], note that almost all experiment replications between researchers at different sites have been unsatisfactory due to variations in experimental conditions. They conclude that identical replications are almost impossible to achieve and thus senseless as non-identical replications also can generate new knowledge.

## 2.2 Replication in the social sciences

In comparison to software engineering the discussion on replications seems more mature in the area of social sciences. E.g. in the research of marketing replication is discussed as early as 1976 [8]. In the social sciences, Tsang and Kwan [38] present perhaps the most notable classification of replications in their discipline. They note that replications can refer to studies of different nature. Throughout their paper of replications there is no special focus given to experiments – a theme that has taken most of the attention in discussing replication in empirical software engineering research. Their view of replications cover the whole spectra, from reanalysis of existing data in a previous study, often done in economics, to replication in which only the research method and question is the same as in the original study. In this case, if the replication confirms the results, it provides great benefit from the point of view of external validation of the original results. However, if the replication does not support the original study, it only serves as a basis for yet another replication that addresses the reason for the discrepancy between the original and the replicated study. However, this is not to be viewed as a failure, since it is unrealistic to think that a single replication or only a very small number of replications could provide enough scientific evidence for any theory.

# 3. METHODOLOGY

## 3.1 Research question

It has been claimed across disciplines that replications are not published frequently enough and that they are not as highly valued as original publications although replications are essential in creating scientific knowledge [1, 14, 22, 32, 38]. One way of mitigating this issue would be to improve the replication studies. Learning from successful and highly valued replications can be seen as a key element in improving the quality of replication studies. Thus, the research question of this work is as follows:

*What elements exist in successful software engineering replication studies?*

Our research question has a set of terms that need to be defined. First, what exactly is a *replication study*? Second, what *criteria* are used for success? Third, what are the *elements* we wish to extract from the studies?

## 3.2 What is a replication study

In previous works Shull et al. [32] suggest there exists two types of replications: *exact* and *conceptual*. Lindsay and Ehrenberg [22] make distinction between *close* and *differentiated* replicated experiments. Tsang and Kwan [38] propose a two dimensional taxonomy for replicated studies in the context of management sciences. The dimensions of their taxonomy are *measurement and analysis* and *data source*.

In this work, we view replication studies as ones that can be placed on a bipolar scale where the ends are *'same research question and method'* and *'identical replication'*, see Figure 1. The end 'same research question and method' represents the minimum requirement for the study to be considered a replication, an approach also applied in [38]. Naturally, it is required that the research questions are well defined in both the original and the replication study. For example, the research question *"Is technique X better than technique Y"* is not well-defined unless 'better' is well defined in the context of the studies. The requirement to use the same research method exists, as there is little possibility to compare two studies if the collected data is completely different in nature, e.g. qualitative data from an industrial case study versus quantitative data from a controlled student experiment. However, as this paper is exploratory in nature, we loosely apply both principles, i.e., it is enough that the research questions and methods are 'close enough' as subjectively determined by the authors of this paper, for a study to be called a replication study.

At the other end *'identical replication'* represents the ideal of replicating everything in the study although this not practically possible. We think that all replication studies can be placed somewhere on this bipolar scale.

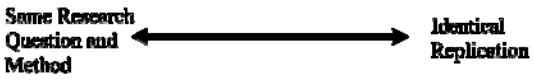


**Figure 1 Bipolar scale for replication studies**

Another requirement for replication studies is that a previous study of the topic exists. It means that the results of the previous study must have been available to the replication study authors. Simultaneous replication is not considered a replication study, but a specific research method, a viewpoint also taken in [38]. We also require that the previous study of the topic is published in a peer reviewed scientific forum and that it has data to back up its claims, i.e. studies started to verify or dispute claims without data are not considered replications. If previous studies fulfilling this criterion do not exist, a study cannot be considered a replication study.

## 3.3 Replication and success

Success can be defined in various ways even within the context of empirical software engineering research. In this work we define a replication study as successful if it is published in a prestigious software engineering journal. The two scientific journals with the highest impact factor and prestige in software engineering are ACM Transactions on Software Engineering Methodology (TOSEM), and IEEE Transaction on Software Engineering (TSE). For this pilot study, we selected only TSE as a means of limiting our search, and since we believe that it contains more empirical papers than TOSEM. Furthermore, as this is small-scale pilot study trying to explore the nature of successful replication studies we only analyzed articles published in TSE during the most recent year (2009). In the future, we hope to extend this scope both temporally as well as to other fora.

## 3.4 Study selection

Next, we describe our study selection process, an overview of which is shown in Table 1. First, the primary author read all the titles and abstracts of all TSE papers published in 2009. The criteria for study inclusion were that the study has an empirical focus and that it is possible that the study is a replication, even when the authors do not explicitly call it as such. At this stage, all papers that clearly promoted a new construction (a tool, process, etc.) were removed, even when they had an empirical component present, as studies presenting new constructions cannot be replication studies. After the first step, the number of papers was reduced from 55 to 14.

In the second step, we analyzed the 14 papers in detail to determine whether a study is a replication according to the criteria presented in Section 2.4. We did this by reading the sections presenting prior works. We had to rely on these sections in determining whether the work is a replication of a previously studied research question or not, because we do not claim to master the prior works of all topics covered in the studies. Our approach was feasible, because in a top quality journal, we can assume that the literature reviews are sufficiently thorough. At this stage, we removed studies that had previously unstudied research questions, or studies that were in fact validating a new construction. A study was included if 50% of its research questions were semantically identical to the original works, and the data was collected using the same research method. For example, one study [11] was excluded because it had three research questions, but only one of them was identical to prior studies. Furthermore, studies with only small but semantically meaningful modifications to research questions were excluded [44], e.g. studying the confounding effect of class size on *change-proneness* versus the confounding effect of class size on *error-proneness*. Making the inclusion/exclusion decisions was by no means easy and we need to develop more clear criteria to ensure repeatability. Finally, we were left with six papers, which we analyzed in detail using the framework presented in the next section.

**Table 1. Study selection and data extraction process**

| | **Study selection step 1** | **Study selection step 2** | **Data extraction** |
|---|---|---|---|
| **Articles analyzed** | 55 | 14 | 6 |
| **Action** | Read the title and abstract. | Read sections describing prior work. | Read the entire article. |
| **Goal** | Exclude studies that have no empirical focus or that clearly cannot be replications. | Exclude studies that are not replications. | Extract data according to the analysis framework. |

## 3.5 Analysis framework

We analyzed the included papers using a two-dimensional framework. First, we characterized the replication *study type*. This helped us understand what elements are typically present in replication studies. Secondly, we analyzed the *type of replication* performed in the study. This helped understand

the nature of successful replication as well as to find shortcomings in the current state of replication reporting. Next, we describe each dimension in more detail.

We characterize the study type using the following classifications. *Research method* is action research, case study, experiment, or survey as defined in [29]. However, it should be noted that our interpretation of survey extends the traditional view that sees surveys only as questionnaires or interviews. For example, a study linking code metrics and defects of two open source systems would be considered a survey. This follows directly from the definitions of Robson [29] who characterizes surveys as "collection of standardized information from a specific population". The example cannot be classified as a case study since Robson stresses that they include "multiple sources of evidence". The example only has two standardized sources of data, the code where the measures are calculated, and the defect database. *Data type* describes the types of data researchers utilized, such as work reports, defect databases, or rigorous observations. *Context* refers to the environment where the data was created, such as students in the classroom, an open source project or industrial software development. For context and data type, we purposefully did not use any preformed schema, such as one promoted in [21], as we wanted to provide rich descriptions of the studies.

We describe the type of replication using the following characteristics. *Data source* comes from [38] and describes whether the replication used the same data, the same population, or a different population. Using the same data means that a replication study had access to the data of the original study. The same population means that the new data is collected from another but similar population. Different population means that the new data is collected from a different population, e.g. students versus professionals. *Operationalization and Analysis* is a modification from [38] which suggests using the term Measurement and Analysis. However, as we think that using the word measurement would be unfair to qualitative research [31], which we consider being of great value in empirical software engineering, we have relabeled the term to be more generic. It describes the type of analysis performed and the way studied constructs are operationalized for the data collection. In practice, this is either the same operationalization and analysis used or different operationalization and analysis used than the original study. *Replication label* tells whether the authors of the replication study use the word replication to describe their study. *Replication support* means whether the replication study made available support for further replications. This support can be anything such as the original data, extensive lab packages, compared in detail in [37], or just a link to a web-page providing further details not available in the paper.

# 4. RESULTS

This section presents our analysis of the six papers [9, 12, 15, 19, 24, 25] that we classified as replication studies. We utilize the framework described in the previous section. We describe each study individually and provide a summary for comparison in Table 2.

## 4.1 Description of the Included Studies

In [24] the authors of this work studied the type of defects detected in manual ad hoc code reviews of C and Java code that had passed smoke or unit testing. Prior work had suggested that 75% of the defects detected affect evolvability rather than functionality [34]. The replication study confirmed this result by using 759 defects from both industrial developers (same population as the original study) and students (different population). The replication study used the same operationalization and analysis in deciding between functional and evolvability defects as the original work. However, a different analysis was performed for the detailed types of evolvability defects than in the original study. We have labeled the work as a survey although the replication study makes no such reference. The paper does not label the work as replication although the contribution of the original study is clearly stated. Nor does the original work or the replication provide support for replication other than the descriptions and measurement results given in the articles.

Hatton [12] studied power-law distributions of an object-oriented open source software. The original work [28] had suggested that power-law relationship exists in the component sizes of object-oriented open source software. The replication confirmed the results of the original study with a different population (none-OO open source software), using the same operationalization and analysis. The paper does not label the work as a survey, but we have done so as size of 21 software systems were analyzed. The paper does not label itself as replication, but support for replication is provided as they analyzed open source systems, available to other researchers without request.

In [15] the authors used PSP data to find the optimal review rate in relation to defect detection efficiency. The studied language was C/C++. The replication study confirmed suggestions of prior work that individual review rate should be 200LOC/h [40]. The population in both studies was the same (industrial developers). However, the context was different as the replication study used data from PSP training sessions taken by professional software developers rather than data from an industrial setting. Although both studies in principle had the same measures, the replication study provided far more sophisticated and analysis. The paper does not label the work as a survey, but we have done so as 617 programs made PSP training were analyzed. The replication study does not label itself as a replication and no replication support is given in the study.

Nan and Harter [25] studied the effect of work pressure, measured by the difference of team-estimated and customer negotiated budget and time, on software development performance at the team level. The original work [33] in organization research had established that work pressure that is not too high or too low would result in optimal work performance, known as the Yerkes-Dodson law. The replication study confirms that the Yerkes-Dodson law is applicable to software development teams as well. The replication utilized a different population than the original study, and the operationalization and analysis was performed in different ways. The authors of the replication had not

indicated the type of their study and we have labeled this as a survey as the researchers selected 66 suitable projects from the company to perform the analysis. The authors make no indication that the study would be replication although they give extensive description of the prior work and admit that they extended the applicability of the Yerkes-Dodson law to a new domain. The authors provide links to appendices that may be helpful in replicating the study.

In [9] the authors studied how software dependencies and work dependencies affects software failures. Software dependencies refer to code level coupling between modules and whether certain modules are modified together as a part of modification request. Work dependencies refer to the set of people who have worked to fix a particular modification request. The impact of both dependencies on software failures have been extensively studied in prior works, but not with the rigor or quality as done in the replication study. The authors analyze defects of over ten thousand modification requests. We have labeled this as a survey although the analyses only consist of two software systems. The replication study utilizes the same population as prior work, i.e. industrial software systems and developers. However, as the authors were not satisfied with the approaches of the original studies, they performed a somewhat different operationalization and analysis. Prior works are extensively presented in the replication study, but the authors do not call their work replication nor do they provide information for future replications.

Koru et al [19] studied the relationship between module size and software defects, one that has been extensively studied in the past. Like many prior works, the study found that small classes are proportionally more defect-prone than large ones. The study used the same population as many prior works, i.e., open-source software systems. However, the study performed different analysis utilizing Cox modeling that, according to the replication study authors, is a more solid approach than the ones used in the past. We have labeled they study as survey since they studied over ten thousand source code modifications of four open source systems. The study does not label itself as a replication, but provides support for future replications as the open source systems can be analyzed by other researchers as well.

## 4.2 Comparison of studies

Next, we use the framework to compare the replication studies with each other, i.e., we go through the columns of Table 2 to make inferences about the replications studies.

Following our definitions of research methods in Section 2.7, we can see that all the studies are surveys. However, as noted in [30], there is fine line between research methods, and other scholars may have classified many of them as case studies. However, it should be noted that none of the studies could be classified as action research or experiments.

Four studies used multiple data types while two had only a single data type. The two most popular data types utilized were defect data (5), and software metrics and size (5), which is most likely due to their good availability.

Three studies utilized industrial developers or their work products as the context of the study. Two studies used open-source, and one used industrial developers in classroom settings related to PSP training. Only one study, utilized students as subjects, but that work also had industrial data as well. There was only one work that drew data from multiple contexts and data sources.

With respect to the replication data, we can see that four studies used different populations than the original works, while three studies used the same population as the original study. When looking at the analysis and type of data used we can see that five studies used different analysis and two used the same analysis. The first study used two populations and both different and identical analysis methods, which explains the fact that the numbers do not add up to the number of studies.

It is interesting to see that no studies labeled their work as replications although in all works the related work was extensively covered. Perhaps the authors are not willing to label their studies as replications, as that might make them seem less novel and/or creative, and thus being perceived as of lower value than without the replication label.

Support for further replication existed in the open-source studies, as such systems are accessible to anyone. In addition to the open-source studies, only one paper provided links to appendixes.

## 4.3 Use of the word 'Replication'

To further look at labeling the studies as a replication, we performed additional search to the ones described in Section 3. We searched using the keyword 'replication' (in the IEEE Explore database, TSE, years 1987-2008, title abstract, keywords), to see whether we could find any other TSE publications in the area of empirical software engineering presenting replication study results. Most of the 25 hits dealt with data replication in computer systems, but we were able to find three relevant hits. Two of them [2, 10] were similar to works presented in sections, i.e. surveys or case studies from industry or open source with high volume of data. One paper was methodological work about replicating experiments [3]. The searches also resulted in two empirical experiments, of which neither was a replication of an existing study. One simply recognized the need for replications [5], and the other had used internal study replication to increase the validity of the results [20].

We also contacted the first two authors of the five studied papers[1] and asked them why they had not used the word replication in their papers and whether they considered their work as a replication. We got two responses. Koru [19] considered that: "Many earlier studies followed a fundamentally flawed approach … plotted size versus defect density … we judged that there was no healthy research done". Thus, Koru recognizes the prior works, but due to their methodological flaws does not consider his work as replication. Kemerer and Paulk [15] saw that: "A replication should seek to mirror as much of the original study as possible. A mere validation (finding similar results through different means) would not be a replication in our view." When it comes to our replication study [24], it is difficult to

[1] One paper was our own work

**Table 2 Comparison of replication studies published in TSE 2009**

| | Study type | | | Replication type | | | |
|---|---|---|---|---|---|---|---|
| **Study Topic** | **Research method** | **Data type** | **Context** | **Data source** | **Operationalization and Analysis** | **Replication label** | **Support for replication** |
| Type of defects detected in code reviews [24] | Survey | 759 defects detected in code reviews | Industrial company and students | Same population and different population | Same analysis and Different analysis | No | No |
| Power-law and component size [12] | Survey | Component size measures 21 systems | Open source | Different population | Same analysis | No | Yes |
| Review rate and defect detection [15] | Survey | Review defects of 617 programs, and effort | PSP training sessions for industry professionals | Same population (different context) | Different analysis | No | No |
| Job pressure and sw-development performance [25] | Survey | Project data (effort, defects) of 66 projects | Industrial company | Different Population | Different analysis | No | Yes |
| Dependencies and defects [9] | Survey | <10000 defects of two system, code metrics | Industrial company | Same population | Different analysis | No | No |
| Size and defects [19] | Survey | >10000 source code changes labeled as defects, module size | Open source | Same population | Different analysis | No | Yes |

judge why the 'replication' word was not used as the thought process of writing this paper has irreversibly changed our view of replications. Looking at our paper from distance, we can state that the word replication should have been used to describe at least part of the work that was carried out in [24].

# 5. DISCUSSION

## 5.1 Answering the research question

*What elements exist in successful replication studies?*

Based on our limited review of articles published in TSE in 2009, it seems that successful replication studies focus on basic software engineering data that is widely available, such as defects, effort, and size or other code measures. To our surprise, students or student projects were not used in any of the replication studies analyzed. Instead, data was either from industrial companies or open source projects. The research methods were case studies or surveys, depending on how one makes the distinction between the two. However, the data utilized in the studies was quantitative, with the defect types being the only qualitative data we were able to distinct [24]. However, one could hardly call defect types a rich set of qualitative data. None of the studies utilized the same data as the original studies, but several used the same population.

None of the authors labeled their work as a replication. We think that this is due to two reasons. Firstly, in software engineering, novelty is highly appreciated. In all the studies, the authors try to make a good case for why their work is novel and distinct from the original studies, and thus needs to be published in a top-level journal. Secondly, the software engineering community has currently taken a narrow view of replication, where replication mainly relates to experimental replication through the development and use of lab-packages. If a wider definition of replication from the social sciences would be adopted by the software engineering community, which we think it should, all the identified studies would have labeled themselves as replications.

Thus, it seems that software engineering is not lacking replications, but that our research community does not value them, and thus the authors are unwilling to present their works as replications. Based on our small sample we found that 11% (6/55) of the publications of a top-level journal were replications if adopting the definition from social sciences. In Business disciplines, the rate of reported replications from 1970 to 1995 varies between 2.2-10.1% [38]. The numbers of software engineering are not very different from those. We are aware that the question whether something is a replication is partly philosophical, and we do not expect everyone to agree with our viewpoint.

## 5.2 Limitations and future work

The most striking limitations to our work is the limited number of studies reviewed and the poor sampling of those studies, i.e. one volume of a journal. However, the purpose of this study was to test our initial ideas of the utility of adopting a wider definition of replication than currently used in the software engineering community. While we think even this small study helped prove our point, we plan to extend the number of studies and fora in a future publication.

It is also likely that researcher bias exists in this work as interpreting the reviewed studies is always affected by individual perceptions and values whether one admits it or not. However, there is little that the authors could do to this if they happen to share a common set of values. The authors have participated in creating and running experiments

regarding code smells [23], pair programming [39] and exploratory testing [13], but our current research outlook is focused more towards industrial surveys and case studies rather than experiments. This fact has undoubtedly affected the results of this study.

Another problem is related to the data available. We take as a starting point that journals accept well-performed replication studies using reasonable criteria. Verifying this is difficult, but could be done by surveying editors of top journals, something we did not do for this study, but hope to do later. It is possible that replication studies are more often rejected than other studies, e.g. due to wrong criteria for assessing replications, or because they are not considered novel enough to be published in a top journal. If this is the case, there might be a need for journal editors to reconsider the criteria applied towards publication of replication studies. As no official statistics of unpublished replication studies are available, it is difficult to analyze the importance of this without discussing the matter with the editors.

## 6. Conclusions

Based on this small-scale study, we draw three main conclusions:

*The software engineering community should re-evaluate the concept of replication and adopt a broader definition.* The broader view used in more mature sciences, such as social sciences, should be used as a basis when discussing replications in the software engineering community. The discussion should be extended from only replication of experiments to replication of all types of empirical works, such as case studies and surveys. We think that the experimental context that has dominated the publication fora of replications has done good job in promoting replications, but think that for software engineering to progress more quickly as a science, this broader view of replication is necessary.

*There is no dramatic shortage of replication in software engineering.* Based on our limited review, over ten percent of publications are actually replications, but authors are probably unwilling to admit it, as it may decrease the likelihood of getting their work published.

High quality replications of high quality original works should be highly valued. We see this as the only way to improve the current state of software engineering research. This would make it more tempting to label studies as replications, which would improve the comparison to prior works in the replicated studies, which would lead to more solid conclusions and in finally to better science.

## 7. References


[1] Anda,B.C.D., Sjøberg,D.I.K. and Mockus,A., 2009."Variability and reproducibility in software engineering: A study of four companies that developed the same system," *IEEE Trans.Software Eng.*, vol. 35, no. 3, 407-429.

[2] Andersson,C. and Runeson,P., 2007."A Replicated Quantitative Analysis of Fault Distributions in Complex Software Systems," *Software Engineering, IEEE Transactions on*, vol. 33, no. 5, 273-286.

[3] Basili,V.R., Shull,F. and Lanubile,F., 1999."Building knowledge through families of experiments," *Software Engineering, IEEE Transactions on*, vol. 25, no. 4, 456-473.

[4] Basili,V.R., 1996, "The role of experimentation in software engineering: past, current, and future," In ICSE '96: Proceedings of the 18th international conference on Software engineering.442-449.

[5] Briand,L.C., Bunse,C. and Daly,J.W., 2001."A Controlled Experiment for Evaluating Quality Guidelines on the Maintainability of Object Oriented Designs," *IEEE Trans. Software Eng.*, vol. 27, no. 6, 513-530.

[6] Brooks,A., Daly,J., Miller,J., Roper,M. and Wood,M., 1996."Replication of experimental results in software engineering," *International Software Engineering Research Network (ISERN) Technical Report ISERN-96-10, University of Strathclyde*,

[7] Brooks,A., Roper,M., Wood,M., Daly,J. and Miller,J., 2007."Replication's role in software engineering," *Guide to Advanced Empirical Software Engineering*, 365-379.

[8] Brown,S.W. and Coney,K.A., 1976, "Building a replication tradition in marketing," In Marketing 1776-1976 and Beyond, Chicago.622-625.

[9] Cataldo,M., Mockus,A., Roberts,J.A. and Herbsleb,J.D., 2009."Software dependencies, work dependencies, and their impact on failures," *IEEE Trans.Software Eng.*, vol. 35, no. 6, 864-878.

[10] Dinh-Trong,T.T. and Bieman,J.M., 2005."The FreeBSD project: a replication case study of open source development," *Software Engineering, IEEE Transactions on*, vol. 31, no. 6, 481-494.

[11] Hamill,M. and Goseva-Popstojanova,K., 2009."Common trends in software fault and failure data," *IEEE Trans.Software Eng.*, vol. 35, no. 4, 484-496.

[12] Hatton,L., 2009."Power-Law Distributions of Component Size in General Software Systems," *Software Engineering, IEEE Transactions on*, vol. 35, no. 4, 566-572.

[13] Itkonen,J., Mäntylä,M.V. and Lassenius,C., 2007, "Defect Detection Efficiency: Test Case Based vs. Exploratory Testing," In Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on.61-70.

[14] Juristo,N. and Vegas,S., 2009, "Using differences among replications of software engineering experiments to gain knowledge," In Empirical Software Engineering and Measurement, 2009. ESEM 2009. 3rd International Symposium on.356-366.

[15] Kemerer,C.F. and Paulk,M.C., 2009."The impact of design and code reviews on software quality: An empirical study based on PSP data," *IEEE Trans.Software Eng.*, vol. 35, no. 4, 534-550.

[16] Kitchenham,B., 2008."The role of replications in empirical software engineering—a word of warning,"

*Empirical Software Engineering*, vol. 13, no. 2, 219-221.

[17] Kitchenham,B.A., Pfleeger,S.L., Pickard,L.M., Jones,P.W., Hoaglin,D.C., El Emam,K. and Rosenberg,J., 2002."Preliminary guidelines for empirical research in software engineering," *Software Engineering, IEEE Transactions on*, vol. 28, no. 8, 721-734.

[18] Kitchenham,B.A. and Pfleeger,S.L., 2002."Principles of survey research part 4: questionnaire evaluation," *ACM SIGSOFT Software Engineering Notes*, vol. 27, no. 3, 20-23.

[19] Koru,A.G., Zhang,D., El Emam,K. and Liu,H., 2009."An investigation into the functional form of the size-defect relationship for software modules," *IEEE Trans.Software Eng.*, vol. 35, no. 2 SPEC. ISS., 293-304.

[20] Laitenberger,O., El Emam,K. and Harbich,T.G., 2001."An internally replicated quasi-experimental comparison of checklist and perspective based reading of code documents," *Software Engineering, IEEE Transactions on*, vol. 27, no. 5, 387-421.

[21] Lethbridge,T.C., Sim,S.E. and Singer,J., 2005."Studying software engineers: Data collection techniques for software field studies," *Empirical Software Engineering*, vol. 10, no. 3, 311-341.

[22] Lindsay,R.M. and Ehrenberg,A., 1993."The Design of Replicated Studies." *The American Statistician*, vol. 47, no. 3,

[23] Mäntylä,M.V., 2005, "An experiment on subjective evolvability evaluation of object-oriented software: explaining factors and interrater agreement," In International Symposium on Empirical Software Engineering.277-286.

[24] Mäntylä,M.V. and Lassenius,C., 2009."What Types of Defects Are Really Discovered in Code Reviews?" *Software Engineering, IEEE Transactions on*, vol. 35, no. 3, 430-448.

[25] Nan,N. and Harter,D.E., 2009."Impact of budget and schedule pressure on software development cycle time and effort," *IEEE Trans.Software Eng.*, vol. 35, no. 5, 624-637.

[26] Petersen,K. and Wohlin,C., 2009, "Context in industrial software engineering research," In Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement.401-404.

[27] Popper,K.R., 1959.*The logic of scientific discovery*, Hutchinson,

[28] Potanin,A., Noble,J., Frean,M. and Biddle,R., 2005."Scale-free geometry in OO programs,"

[29] Robson,C., 2002.*Real world research: A resource for social scientists and practitioner-researchers*, Wiley-Blackwell,

[30] Runeson,P. and Höst,M., 2009."Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, 131-164.

[31] Seaman,C.B., 1999."Qualitative methods in empirical studies of software engineering," *Software Engineering, IEEE Transactions on*, vol. 25, no. 4, 557-572.

[32] Shull,F.J., Carver,J.C., Vegas,S. and Juristo,N., 2008."The role of replications in Empirical Software Engineering," *Empirical Software Engineering*, vol. 13, no. 2, 211-218.

[33] Singh,J., 1998."Striking a balance in boundary-spanning positions: An investigation of some unconventional influences of role stressors and job characteristics on job outcomes of salespeople," *The Journal of Marketing*, vol. 62, no. 3, 69-86.

[34] Siy,H. and Votta,L., 2001, "Does the modern code inspection have value?" In International Conference on Software Maintenance.281-289.

[35] Sjoberg,D.I.K., Dyba,T. and Jorgensen,M., 2007."The future of empirical methods in software engineering research," *Future of Software Engineering, 2007.FOSE'07*, 358-378.

[36] Sjøberg,D., Hannay,J., Hansen,O., Kampenes,V., Karahasanovic,A., Liborg,N.K. and Rekdal,A., 2005."A survey of controlled experiments in software engineering," *IEEE Trans.Software Eng.*, vol. 31, no. 9, 733-753.

[37] Solari,M. and Vegas,S., 2006, "Classifying and analysing replication packages for software engineering experimentation," In Workshop Empirical Software Engineering (WSESE06) of the 7th International Conference on Product Focused Software Process Improvement, Amsterdam, Holland.

[38] Tsang,E.W.K. and Kwan,K., 1999."Replication and Theory Development in Organizational Science: A Critical Realist Perspective," *The Academy of Management Review*, vol. 24, no. 4, 759-780.

[39] Vanhanen,J. and Lassenius,C., 2005, "Effects of pair programming at the development team level: an experiment," In Empirical Software Engineering, 2005. 2005 International Symposium on.10.

[40] Weller,E.F., 1993."Lessons from three years of inspection data [software development]," *Software, IEEE*, vol. 10, no. 5, 38-45.

[41] Wohlin,C., Höst,M., Runeson,P., Ohlsson,M.C., Regnell,B. and Wesslén,A., 2000.*Experimentation in software engineering: an introduction*, Kluwer Academic Pub,

[42] Wohlin,C., 2004."Are Individual Differences in Software Development Performance Possible to Capture Using a Quantitative Survey?" *Empirical Software Engineering*, vol. 9, no. 3, 211-228.

[43] Yin,R.K., 2002.*Applications of case study research*, Sage Pubns,

[44] Zhou,Y., Leung,H. and Xu,B., 2009."Examining the potentially confounding effect of class size on the associations between object-oriented metrics and change-proneness," *IEEE Trans.Software Eng.*, vol. 35, no. 5, 607-623.