Towards Multi-Method Research Approach in Empirical Software Engineering

Vladimir Mandić, Jouni Markkula, and Markku Oivo

University of Oulu, Department of Information Processing Science, Rakentajantie 3,
90014 University of Oulu, Finland
vladimir.mandic@tol.oulu.fi, Jouni.Markkula@oulu.fi, Markku.Oivo@oulu.fi

Abstract. This paper presents results of a literature analysis on Empirical Research Approaches in Software Engineering (SE). The analysis explores reasons why traditional methods, such as statistical hypothesis testing and experiment replication are weakly utilized in the field of SE. It appears that basic assumptions and preconditions of the traditional methods are contradicting the actual situation in the SE. Furthermore, we have identified main issues that should be considered by the researcher when selecting the research approach. In virtue of reasons for weak utilization of traditional methods we propose stronger use of Multi-Method approach with Pragmatism as the philosophical standpoint.

Keywords: Empirical Methods, Experimentation in Software Engineering, ESE, Multi-Method Research, Reporting Experiments.

1 Introduction

Researchers in the field of software engineering (SE) are facing dilemma: which empirical research approach should be taken? As Shaw [1] has pointed out that there is no shared understanding of preferred research approaches inside SE community, and therefore there is no clear response to the question. This encourages us to revisit the issue.

Researchers are usually confronted by following questions: Can the traditional scientific approach[1] of experimentation be effectively utilized for SE setting? What is an alternative? What should be taken in account while considering alternative approaches? Questions stated here resemble first decisions that a researcher has to make.

The objective of our research was to explore the current literature in order to seek sufficient sources regarding problems of utilizing quantitative methods like experimentation, statistical hypothesis testing, and experiment replications. Based on our literature study and analysis, we are able to suggest some alternative approaches. The results of the our analysis are packed in a simple decision making process. This process can help the researchers in their decisions regard the selection of research approaches and appropriate methods.

[1] Examples of traditional research concepts are statistical hypothesis testing and experiment replications.

F. Bomarius et al. (Eds.): PROFES 2009, LNBIP 32, pp. 96–110, 2009.

This paper is result of the literature analysis. Literature review process was not systematic in terms as Kitchenham [2] suggests. The process started by reviewing two book classics on experimentation in SE [3,4]. After that, the review was complemented and deepened with additional references on specific issues, such as statistical hypothesis testing, experiment replications, and experiment reporting. Also the method of following bibliographical trails [5] was used. The following resources were used for the analysis: Google Scholar, IEEE Xplore, SpringerLink, Wiley InterScience, ACM, and reference databases available in University of Oulu Library. From the large number of potential references we selected 46 most relevant references for further analyzis. The structure of analyzed references is given in Table 1.

Table 1. Reference structure

Ref. Type	Journal	Book	Ed. Book	Conference
Percentage of total	54%	15%	20%	11%

The references were categorized using the following criteria:

1. **Meta-studies:** meta-studies on the topic of empirical and experimental methods in software engineering. Number of references: 7.
2. **Reporting experiments:** papers that report some empirical studies. Number of references: 7.
3. **Empirical methods:** papers that define methods and techniques for empirical research or comment on utilization of the methods in SE. Number of references: 26.
4. **Other:** References which were not categorized by first three criteria. Number of references: 6.

By reviewing the literature we found that researchers in the field of software engineering still seems to base their findings more on experiences and personal feelings then on empirical evidences (section 2). One of the most powerful scientific methods, experimentation, was introduced to SE research as one possible solution to the problem (section 3). However, due to the strong dependence of the objects under investigation upon context and the field immaturity, adaptation of the experimentation is lacking sufficient level of statistical significance (as shown in section 4). A concept of corroboration and/or refutation of findings through replications of the experiments is important for justifying results and knowledge creation process. Reported studies on experimentation in SE settings revealed us that external replications are not easily applicable (section 5). Besides reporting quantitative result, a structured qualitative analysis is needed to overcome contextual dependences and to explain design of experiment at such level of details to enable external replications (section 6). Multi-method approach advocates use of other methods in combination with purpose of achieving more creditable results (section 7). At the end we discuss how the approach can produce a near-close effect as the concept of experiment replications (section 8).

2 Motivation for the Use of Empirical Methods in Software Engineering

In the field of software engineering so called "advocacy research", has often been used in last decades [6,7,8,9]. Shortly we can illustrate this approach with a following scenario [6, p. 87]:

Authors describe a new concept in considerable detail; recommend the concept to be transferred to practice. Time passes, and other researchers derive similar conclusions. Eventually the consensus among researchers is that the concept has clear benefits. Yet practitioners often seem unenthused. Researchers, satisfied that their communal analysis is correct, become frustrated. Heated discussion and finger-pointing ensues.

Given scenario is lacking empirical proofs that the proposed new concept is beneficial. Such empirical proofs can dramatically change the scenario. All communal analysis will shift from a personal, subjective, judgment regarding substance to objective reasoning based on the empirical evidence.

One of the roles of the experimentation is to enable researchers in the field of software engineering to derive conclusions based on empirically made observations.

The main concern of the researchers is with what degree of certainty it is possible to claim that a hypothesis is true or false [7, p. 457].

Basili [10] describes analogies with other fields of research. Separation on two groups of people and existence of strong feedback loop among them is the common element in all those analogical models. The basic idea is to have a clear separation on two groups: researchers and practitioners. In this tentative model we can identify three loops:

Loop 1. Describes activity of the researcher. A researcher relies on the global body of knowledge, and entire process which is encapsulated by the loop 1, has a basis in academic research and academic writing. The researcher's role is to understand the nature of processes and products, and relationships between them [10, p. 443].

Loop 2. Describes activity of the practitioner. Practitioners use tools, methods and techniques in daily work. The feedback of using tools, methods and techniques always exists; the question is how well is it formulated and/or documented.

Loop 3. Is the feedback loop, which was the main reason to consider this kind of model. According to Basili *et al.* [7,10] this kind of a loop has a significant influence on knowledge creation process.

Unfortunately, the implementation of the proposed model is not straightforward, even worse it is questioned if it is feasible at all. Some problems that affect communication paths between researchers and practitioners are [11]:

(1) Data sharing, this includes problems of work sharing and intellectual property rights.

(2) Data Interpretation problem is illustrated with following questions made by Basili [11]: *When we find agreement how much can we generalize, how do*

we incorporate the context variables in the interpretation, how do we assign the degree of confidence in the interpretation? When we find disagreement do we expand the model, identify two different contexts, or reject the model?

Vegas *et al.* [12] propose some possible mechanisms for dealing with those issues like licensing, software support tools, and etc. General conclusion is that each field has its own particular problems and issues and we have to tailor such rules for the SE field [12, p. 116].

We will formulate another question regarding feasibility of the proposed approach in software engineering field. *Has the global body of knowledge reached "critical mass", and became capable of supporting the separation on the researchers and the practitioners?*

Physics and medicine certainly fall in well-developed disciplines [13, p. 1145]. Well-developed disciplines have well defined a relationship structure within body of knowledge.

Such established structures provide a comfortable environment for researchers, and enables them to create new concepts, theories, with high degree of confidence. *Researchers in the field of software engineering are facing: human subjects with large ability variations, ill-defined processes, products with poorly defined characteristics, a limited number of facts, nothing that can be regarded as a universal constant,...* [14, p. 188].

Can the lack of the structure in the software engineering body of knowledge be compensated with strong, direct, feedback loop from the practice to the researcher? Our response to the question: yes, it has to be.

3 Basic Terminology of the Software Engineering Experimentation

First we will define the basic terminology adopted from Wohlin *et al.* [4], alternative terminology is commented and referenced.

In Figure 1 the basic elements of an experiment are illustrated. Figure is adopted from [4, p. 34], with an addition of *context*. It is very important to be aware of an existing context and its influence on experiment. Partially the

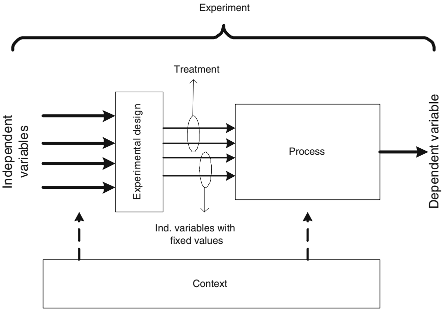

Fig. 1. Illustration of the experimental process

influence of the context will be taken in account through experimental design. It is not possible to model, take in account, all numerous variables existing in the context. The objective of experimental design is to reduce the context interference to the level of noise. Context plays important role in reporting and sharing results of the experiments, therefore it is advisable to document it as detail as possible [15,16]. Endres *et al.* [17] formulated *Conjecture*: *Empirical results are transferable only if abstracted and packaged with context.* Kitchenham *et al.* [18] proposed an entire set of guidelines for dealing with context during experiment.

The variables that are in the focus of a study are called *dependent* or response variables, all other variables are called *independent.*

Independent variables can have constant value during experiment and then they are *fixed variables.*

Independent variables that change value (in controlled manner) during experiment are called *factors.* One particular value of the factor is called *treatment.* Alternative terminology for treatment is *alternative or level* [3, p. 60].

Subjects of the experiment are usually people that have to apply a treatment. *Object* is any artifact of the process on which a treatment is applied. Objects can be referred as *experimental units* [3, p. 57]. An experiment consists of a set of *tests or trials*, where each test is a combination of treatment, subject and object.

Cook *et al.* [19] define ***quasi-experiments*** as experiments that have treatments, outcome measures, and experimental units, but do not use random assignment to create comparison from which treatment-cause change is inferred.

Experimental Design. Figure 1 illustrates the role of the experimental design in an experiment. The goal of experimental design is to isolate variation of the interest. Juristo *et al.* [3, p. 84] give an overview of the experimental designs based on parameters like: number of factor, number of alternatives per factor, and existence of the blocking variables. The basic experimental designs are: one-factor design, block design, factorial design, nested design, fractional design, and factorial block design.

Randomization in Experimental Designs. Randomized design means that the factor alternatives are assigned to experimental units in absolutely random order. Concerning SE, both the factor alternatives and the subjects have to be randomized, as the subjects (people) have a critical impact on the value of dependent variable [3]. The request for randomizing both subjects and factor alternatives might sound odd, unless the idea of randomizing subjects is a proposal how to deal with a fact that in SE field subject characteristics vary a lot even within same class (Example: productivity of the programmers with same number of years of experience). Still remains a question how well the randomization of the subjects can effectively solve the problem. When the idea of randomization was introduced into experiments, the goal was to ensure that errors were independent. With new applications of the significance testing, a representative sample has been added. Miller [14] observed that very often a mistake is made by using randomization to "discard" representativeness.

More often feasibility of the random sampling in the field of software engineering is questioned. Miller *et al.* [20] define the sampling problem as: *Regardless of the characteristic under investigation, the software engineering field has no defined sampling frame (i.e. description of the entire population) for its practitioners, and hence we cannot know if the sample is truly representative of the underlying population.* However there are no universal sampling frameworks in other fields as well, practice is that research setting determines sampling strategy. But we can notice that other fields have some elementary, basic, knowledge about population which is used for defining sampling strategy. That kind of basic knowledge is lacking in the field of SE.

4 Quantitative Aspect of the Experimentation

Quantitative methods are maybe the only approach that can provide researchers with concrete information about certainty of their conclusions. Other approaches are also considered to be suitable for the field of software engineering at this moment, like explorative studies and qualitative confirmatory analysis [9].

Experimental analysis is dependent on the characteristics of data that are collected or measured during experiment. Depending on the nature of data several measurement scales can be used: nominal, ordinal, interval, or ration. Information about measurement scale is important because it determines which statistical methods can be and cannot be used for analyzing results. Generally methods are divided in two groups: parametric and non-parametric methods [3,4]. Most common methods are given in Table 2.

Table 2. Overview of parametric/non-parametric tests for different designs

Design	**Parametric**	**Non-parametric**
One factor, one treatment		Chi-2 Binomial test
One factor, two treatments, completely randomized design	t-test F-test	Mann-Whitney Chi-2
One factor, two treatments, paired comparison	Paired t-test	Wilcoxon Sign test
One factor, more than two treatments	ANOVA	Kruskal-Wallis Chi-2
More than one factor	ANOVA	

Statistical hypothesis testing. The Neyman-Pearson type of significance testing is the form of testing a null hypothesis, where the null hypothesis is formulated with the purpose if it is rejected to allow the researcher considering an alternative hypothesis and conclude that an effect exists [20, p. 286]. Basic steps of statistical hypothesis testing are [14, p. 183]:

1. The construction of a null hypothesis;
2. The collection of data;
3. A statistical test against the null hypothesis is undertaken;
4. The generated $P-$value[2] is considered against the null hypothesis; and one or more interpretations are made.

The probability of committing Type I error is statistical significance, denoted by Greek letter α.

Test significance value, α is set in advanced, after having all data form experiment the $P-$value is calculated and compared to α [14].

Statistical power analysis. As a part of statistical significance testing is statistical power analysis. Power analyses involve three components [20]:

- **The significance criterion (α).**
- **The sample size (n)**: the larger the number of samples, the smaller the error, the greater accuracy.
- **The effect size (γ)**: the degree to which the phenomenon under study is present in the population (sample).

Methods how to calculate or estimate sample size are given in [3,20].The only critical step in this process is estimate of the effect size. Coehn has established a convention that *small effect* is not observable with bare eyes, *medium effect* is observable with researcher's eyes and *large effect* is high over an average.

In the study [21, p. 749] a systematic literature analysis has been performed in order to conclude how Coehn's convention maps to the field of software engineering. The findings of the study showed that in SE effect size is for 50% smaller for small effect size and about 25% to 20% for medium and large effects. This decrease in effect size calls for larger sample size, which is very often difficult to achieve in SE experiments.

5 Software Experiment Replication

The first experiment is usually referenced as an original, later experiments which have the same null hypothesis as original are called replications. Replicated experiments can be categorized in two groups [22]:

Exact replications or partial replications of the original, they have the same alternative hypothesis as the original, usually in the form H_1^{rep}: The results of the replication will be in same direction as the first (original) experiment [23].

Replications with goal to improve on the original. This type of replicated experiment will have different, improved formulation of the alternative hypothesis.

[2] The $P-$value can be viewed as the probability that results obtained due to chance, therefore small values are taken to indicate that results where not just a chance.

First type of the replicated experiments is common for *internal replications*, when the same researcher performance replicated experiment, while second type would be expected in *external replication.*

Replication of the experiments is important for at least two reasons: (1) it is the best way to validate experiment (experimental results and experimental design) [22, p. 237] and (2) as the instrument of Popperian inference. General statements (hypotheses) cannot be proved, but they can be disproved. This is the basic idea of Popper's conjuncture [24].

The statistical hypothesis testing is an instrument of Popperian inference; or more correctly that statistical hypothesis testing was designed as an instrument of the hypothetic-deductive scientific method and that this method and Popperian inference are effectively equivalent approaches [14].

Following this philosophy, we can note that replication of the experiments (test) is crucial for making a theory to become well-proved and trusted. How to get that level of replications in software engineering?

Several studies have shown that experimentation is not utilized well enough in software engineering at the level of the original experiment (first experiment) [6,25,26], and the field is far away from performing replications. Brooks *et al.* [27] noted that in cases when people are dominant factor, controlled experiments are less effective. Miller [22] defined dimensions of the replication framework for software engineering field. Those dimensions can be seen as major categories of causes for the weak utilization of the replications in SE. We present those causes in the cause-effect diagram Figure 2.

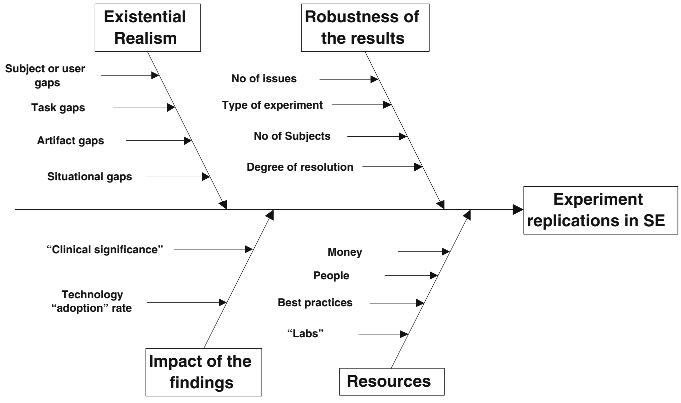

Fig. 2. Major categories of the problems regarding experiment replications in the field of software engineering

Existential realism. Software engineering experiment differs from the real world. Numerous differing points are characterized as: *subject gaps*, *task gaps*, *artifact gaps*, and *situational gaps* [22]. Factors affecting different types of gaps are varying from socio-psychological up to mixed influence of socio-technological factors.

Experimental results are robust when they produce relatively stable results across a range of minor variations in experimental setting. High robustness can be a motivation factor for replicating experiments. If experiment results are robust, replicated experiments even if they reject original hypothesis they can be used by researchers to generate new conclusions.

Impact of the findings is dealing with question: Will the finding convince practitioners in real world setting to change, adapt or adopt new practices? This issue is correlated with technology transfer and adaption rate of the technology.

Resources. Beside material resources an attention is raised on non-existing experimental practices for software engineering. Basili [10] pointed out that 'laboratories' exist only where practitioners build software systems. This fact complicates entire experimental process and increases cost.

Analyzing results of the replicated experiments. Once, having results of the replicated experiments, the method for analyzing the results should be selected. Based on practices in the social sciences, two groups of analyzing methods are identified [22]: (1) Meta-analytical procedures and (2) informal approach.

Meta-analysis provides a simple quantitative framework for comparing and combining results of the experiments. Most common techniques used in software engineering are: comparative and additive meta-analysis. Both methods are usually done to compare results of two experiments.

6 Reporting Experiments

Communication between communities of the researchers and practitioners is very important, especially when reporting results of the experiments in such way to enable, encourage, others to replicate or conduct similar experiments. Therefore a mutually accepted standard or form, of the reporting results is welcome. According to Miller [22] only three serious initiatives were proposed. First one is Basili's approach for classifying experiments. Originally this scheme was developed for a meta-study on experimentation in software engineering and later was used as basis for experimental paradigm [28,16]. Main elements of the scheme are: definition, planning, operation, and interpretation. Beside the original idea for developing the scheme, it is possible to use it as a guideline for reporting experiments.

Second scheme developed by Lott *et al.* [29] has many similarities with first one, including the use of GQM to derive the subsequent scheme. The main elements of this scheme are: (1) goals, hypothesis and theories, (2) Experimental planning, (3) Experimental procedures, and (4) results. Third scheme, actually entire package for experimentation, is developed by Kamsties and Lott. Unfortunately it is least likely that entire package can be implemented in software engineering.

Beside those three schemes, recently Jedlitschka *et al.* [30] proposed a new scheme based on comparative study of existing schemes, mainly in software engineering area. The scheme suggests following elements: structured abstract, introduction, related work, experiment planning, execution, analysis, interpretation, discussion and conclusions, future work, acknowledgements, references, and appendices.

7 Multi-Method Research Approach

The multi-method or mixed-method approach originates from the social sciences [31,32]. The basic idea of the approach is to investigate a phenomenon using a combination of empirical research methods, with intention that the combination of the methods complements each others. The combination may include quantitative and qualitative methods to collect, analyze, and interpret both types of data [33]. This approach offers potential for more stable and generalizable results in empirical software engineering research.

Discussions on utilizing multi-method approach for information systems research started in late 80's and early 90's [34,35], continued in 2000's [33,36,37]. Despite the agreement of the researchers in information systems that there are benefits of utilizing multi-method approach, there is no such agreement among researchers in the field of software engineering. Reported SE related studies using multi-method are still very rare. One example is use of multi-method approach to study collaboration of global virtual teams [38]. Wood *et al.* [23] used multi-method to investigate object-oriented technology with particular focus on how the inheritance levels affect maintainability of software.

The use of multi-method approach is shaded with philosophical discussions if such methodological pluralism is acceptable [39,37,36]. Particular methods are paired with paradigms or philosophical standpoints [37, p. 243], which raises the question whether mixing of the methods would mean mixing of the paradigms. The question evolved in debate over *incompatibility* vs. *compatibility* thesis.

Howe [39] points out that: *The incompatibility thesis, like the drunkard's search*[3]*, permits the "lights" to determine what is to be looked for and where.* Howe took bottom-up approach in proving his *compatibility thesis* [39]. He discussed what quantitative and qualitative means at levels of *data*, *design and analysis*, and *interpretation of results.* The conclusion was that mixing of the methods is acceptable if it provides additional evidences, and it does not imply mixing of the paradigms. Conclusion made by Howe is known as *compatibility thesis.*

Mingers [37] arguments that phenomena studied by researchers in the field of information systems are extremely complex. Such complexity can be studied if it is decomposed on dimensions of *the multidimensional world.* Therefore it is less likely that one method can be successfully applied to all dimensions.

The multi-method approach is not limited to the combining qualitative and quantitative methods. Also the combination of different quantitative methods is

[3] Kaplan's story illustrating the "principle of the drunkard's search." *There is a story of a drunkard searching under a street lamp for his house key, which he had dropped some distance away. Asked why he didn't look where he had dropped it, he replied, "It's lighter here!"* (Kaplan, 1964).

possible. When designing a multi-method research, the following strategies can be used [23,32]:

Evolutionary or sequential is followed when there is little research conducted on a particular phenomenon, or where research hypothesis require increased focus.

Complementary or concurrent or triangulation aims to enhance the validity of research findings. Different research methods are used independently to study phenomenon. An example how to structure a study which uses triangulation as method is given in [40].

Transformative strategy, procedures which use theoretical lens or perspective in qualitative research. Examples of the perspectives are: Feminist perspective, Critical theory, and Racialized discourse [32].

Guidelines for categorizing mixed methods can be found in *mixed method research framework* [33]. The classification matrix (Table 3) is based on *purpose* dimension: triangulation, complementary, development, initiation, and expansion. And *approach* dimension how the method is applied: sequential, parallel and independent.

Table 3. Mixed method (Multi-method) research framework [33, p. 1]

		Approach		
		Sequential	**Parallel**	**Independent**
Purpose	**Triangulation**			
	Complementarity			
	Development			
	Initiation			
	Expansion			

The following methods are usually combined: *observational studies, pre-experiment studies, quasi-experiments, controlled experiments, surveys.* More comprehensive list of the methods can be found in *taxonomy of information systems research approaches* [35, p. 96]. The taxonomy classifies methods by the object of a study: (1) society, organization/group, (2) individual, (3) technology, and (4) methodology. In studying technology or methodology objects, both groups of approaches (qualitative and quantitative) can be utilized. In studying socio-psychological phenomena qualitative approaches are suggested.

The main challenge of the multi-method design/planning is how to select a good combination of methods. For that purpose Wood *et al.* [23] proposed a set of criteria:

- **Internal validity:** The extent to which causal conclusions can be made from the study.
- **External validity:** The extent to which results may be generalized to the population under study and other settings.

Reaching a high validity is a balancing game because some validity types are opposing each other [7, p. 457]: *...make it less likely that the validity types can all be satisfied at the same time: e.g., making a study more realistic to achieve a high external validity is in tension with the ability to manipulate the context to get a high internal validity.*

- **Ease of replication:** The ease with which the study can be repeated under the same conditions.
- **Potential for theory generation:** The potential to generate new causal theories.
- **Potential for theory confirmation:** The potential to test a theory providing robust conclusions.
- **Cost per subject:** The relative cost of the study.

Based on this characterization, Daly *et al.* [41] provide the following advice:

A Maximize internal validity, external validity, and ease of replication by selecting a combination of the methods that jointly satisfy these criteria. For example a controlled experiment (high internal validity) and a survey (high external validity), both being relatively easy to replicate, provide good coverage of the criteria [41].

B Since the cost of a multi-method approach is usually significant, combine methods to minimize overall cost.

C Determine the need for theory generation and theory confirmation, considering whether the perspective of the approach is complimentary or evolutionary. For example, if it is evolutionary, observational studies may be use for theory generation combined with controlled experiments for theory confirmation.

In the context of the multi-method approach, observational studies may be used to characterize, baseline, and/or identify relationships. They are also very often seen in combination with other methods.

8 Conclusions

In order to avoid the habit of *advocacy research*, it is necessary to justify conclusions with empirical evidences. Empirical evidences have also a psychological effect as a very strong element of persuading other researches and practitioners to trust the validity and usefulness of the results. Without this persuasion, especially practitioners will not strive to use the result of the research. This phenomenon is known as *clinical significance* and it is a major factor for not having wide replications of software experiments within researcher's community. Also, the everyday use of methods and tools in practice can be considered as a form of replication, unfortunately reported in a very free form of experience reports or lessons learned.

The complexity of the phenomena under study in the field of SE sets a very sophisticated conditions and constraints on performing software experiments and replications. Existential realism argues that a gap between experimental setting and real world situations is too large. Because of the lack of sufficient body of knowledge which would allow researchers to bridge the gap. In such kind of situations qualitative research approaches are much more applicable then quantitative.

Robustness of results could be achieved with high statistical significance in experimentation. Unfortunately, it is common to have low statistical significance in SE experiments which is followed with less robust results. Use of different methods with purpose of triangulation can significantly increase robustness of the results, especially if the methods are applied independently.

The impact of the findings can be improved only if trust and confidence in new theories and research findings is increased. That can be achieved by using multi-method approach. This approach is compatible with Pragmatism as the philosophical standpoint. It is an effective tool for confirming results with sufficient flexibility to cope with specifics of the software engineering research.

Our proposal is based on analysis of available literature and previous experiences in the field of software engineering. The proposal is not a silver bullet, but it is good starting point. The main advantage of the multi-method approach is the possibility to balance method's rigor for a given research setting. Probably the biggest disadvantage is that it requires the researcher to be proficient in several empirical methods instead of just one method.

This preliminary literature analysis will be a base for the future work. We plan to expend literature review in more systematic way. Our further contributions on this topic will be focused on exploring relationships between different philosophical standpoints and empirical methods, and their applicability in software engineering settings.

Acknowledgements. This article is based on the work carried out in the VASPO project (Value-based Software Process Improvement and Organizational Change Management) financed by the Finnish Funding Agency for Technology and Innovation (Tekes).

References

1. Shaw, M.: What makes good research in software engineering? International Journal of Software Tools for Technology Transfer 4(1), 1–7 (2002)
2. Kitchenham, B.: Guidelines for performing systematic literature reviews in software engineering. Technical report, TR-EBSE-2007-01, UK (2007)
3. Juristo, N., Moreno, A.: Basics of Software Engineering Experimentation. Kluwer Academic Publishers, Dordrecht (2003)
4. Wohlin, C., Runeson, P., Horst, M., Ohlsson, M., Regnell, B., Wesslen, A.: Experimentation in Software Engineering: An Introduction. Kluwer Academic Publishers, Dordrecht (2000)
5. Turabian, K.: A Manual for Writers of Research Papers, Theses, and Dissertations. The University of Chicago Press, Chicago (2007)
6. Fenton, N., Pfleeger, S., Glass, R.: Science and substance: A challenge to software engineers. IEEE Software 4(11), 86–95 (1994)

7. Basili, V., Shull, F., Lanubile, F.: Building knowledge through families of experiments. IEEE Transactions on Software Engineering 25(4), 456–473 (1999)
8. Oivo, M.: New opportunities for empirical research. In: Basili, V.R., Rombach, H.D., Schneider, K., Kitchenham, B., Pfahl, D., Selby, R.W. (eds.) Empirical Software Engineering Issues. LNCS, vol. 4336, p. 22. Springer, Heidelberg (2007)
9. Oivo, M., Kuvaja, P., Pulli, P., Similä, J.: Software engineering research strategy: Combining experimental and explorative research (eer). In: Bomarius, F., Iida, H. (eds.) PROFES 2004. LNCS, vol. 3009, pp. 302–317. Springer, Heidelberg (2004)
10. Basili, V.: The role of experimentation in software engineering: Past, current, and future. In: 18th International Conference on Software Engineering, pp. 442–449. IEEE, Berlin (1996)
11. Basili, V.: Measurement and model building, introduction. In: Basili, V.R., Rombach, H.D., Schneider, K., Kitchenham, B., Pfahl, D., Selby, R.W. (eds.) Empirical Software Engineering Issues. LNCS, vol. 4336, pp. 68–69. Springer, Heidelberg (2007)
12. Vegas, S., Basili, V.: Measurement and model building, discussion and summary. In: Basili, V.R., Rombach, H.D., Schneider, K., Kitchenham, B., Pfahl, D., Selby, R.W. (eds.) Empirical Software Engineering Issues. LNCS, vol. 4336, pp. 115–120. Springer, Heidelberg (2007)
13. Curtis, B.: Measurement and experimentation in software engineering. In: Proceedings of IEEE, pp. 1144–1157. IEEE, Los Alamitos (1980)
14. Miller, J.: Statistical significance testing – a panacea for software technology experiments? Journal of Systems and Software 2(73), 183–192 (2004)
15. Zelkowitz, M., Wallance, D.: Experimental validation in software engineering. Information and Software Technology 11(39), 735–743 (1997)
16. Basili, V., Selby, R., Hutchens, D.: Experimentation in software engineering. IEEE Transactions on Software Engineering 12(7), 733–743 (1986)
17. Endres, A., Rombach, D.: A Handbook of Software and Systems Engineering: Empirical Observations, Laws and Theories. Pearson Education, Harlow (2003)
18. Kitchenham, B., Pfleeger, S., Pickard, L., Jones, P., Hoaglin, D., El Emam, K., et al.: Preliminary guidelines for empirical research in software engineering. IEEE Transactions on Software Engineering 8(28), 721–734 (2002)
19. Cook, T., Campbell, D.: Quasi-Experimentation: Design and Analysis Issues for Field Settings. Houghton Mifflin Company, USA (1979)
20. Miller, J., Daly, J., Wood, M., Roper, M., Brooks, A.: Statistical power and its subcomponents - missing and misunderstood concepts in empirical software engineering research. Information and Software Technology 4(39), 285–295 (1997)
21. Dybå, T., Kampenes, V., Sjøberg, D.: A systematic review of statistical power in software engineering experiments. Information and Software Technology 8(48), 745–755 (2006)
22. Miller, J.: Replicating software engineering experiments: a poisoned chalice or the holy grail. Information and Software Technology 4(47), 233–244 (2005)
23. Wood, M., Daly, J., Miller, J., Roper, M.: Multi-method research: An empirical investigation of object-oriented technology. Journal of Systems and Software 1(48), 13–26 (1999)
24. Popper, K.: The Logic of Scientific Discovery. Routledge Classics, New York (1959)
25. Ramesh, V., Glass, R., Vessey, I.: Research in computer science: an empirical study. Journal of systems and Software 2(70), 165–176 (2004)

26. Sjøberg, D., Hannay, J., Hansen, O., By Kampenes, V., Karahasanovic, A., Liborg, N.K., et al.: A survey of controlled experiments in software engineering. IEEE Transactions on Software Engineering 31(9), 733–753 (2005)
27. Brooks, A., Roper, M., Wood, M., Daly, J., Miller, J.: Replication's role in software engineering. In: Shull, F., et al. (eds.) Guide to Advanced Empirical Software Engineering, pp. 365–379. Springer, London (2008)
28. Basili, V., Selby, R.: Paradigms for experimentation and empirical studies in software engineering. Reliability Engineering and System Safety 1(32), 171–191 (1991)
29. Lott, C., Rombach, D.: Repeatable software engineering experiments for comparing defect-detection techniques. Empirical Software Engineering 1(3), 241–277 (1996)
30. Jedlitschka, A., Ciolkowski, M.: Reporting experiments in software engineering. In: Shull, F., et al. (eds.) Guide to Advanced Empirical Software Engineering, pp. 201–228. Springer, London (2007)
31. Easterbrook, S., Singer, J., Storey, M.A., Damian, D.: Selecting empirical methods for software engineering research. In: Shull, F., et al. (eds.) Guide to Advanced Empirical Software Engineering, pp. 285–311. Springer, London (2008)
32. Creswell, J.: Research Design: Qualitative, Quantitative, and Mixed Method Approaches. Sage Publications, Inc., London (2008)
33. Petter, S., Gallivan, M.: Toward a framework for classifying and guiding mixed method research in information systems. In: The 37th Hawaii International Conference on System Sciences, Big Island, HI, USA, pp. 1–10 (2004)
34. Nunamaker, J., Chen, M., Purdin, T.: Systems development in information systems research. Journal of Management Information Systems 7(3), 89–106 (1991)
35. Galliers, R.: Research issues in information systems. Journal of Information Technology 2(8), 92–98 (1993)
36. Sawyer, S.: Studying organizational computing infrastructures: Multi-method approaches. In: Baskerville, R., et al. (eds.) Organizational and Social Perspectives on Information Technology, IFIP TC8 WG8.2 International Working Conference on the Social and Organizational Perspective on Research and Practice in Information Technology, pp. 213–232. Kluwer, Aalborg (2000)
37. Mingers, J.: Combining is research methods: Towards a pluralist methodology. Information Systems Research 12(3), 240–259 (2001)
38. Steinfield, C., Huysman, M., David, K., Yang Jang, C., Poot, J., Huis in 't Veld, M., et al.: New methods for studying global virtual teams: Towards a multifaceted approach, Wailea Maui, Hawaii, USA. In: The 34th Hawaii International Conference on System Sciences 2001, pp. 1–10 (2001)
39. Howe, K.: Against the quantitative-qualitative incompatibility thesis. Educational Researcher 17(8), 10–16 (1998)
40. Bratthall, L., Jørgensen, M.: Can you trust a single data source exploratory software engineering case study? Empirical Software Engineering 7(1), 9–26 (2002)
41. Daly, J., El Emam, K., Miller, J.: An empirical research methodology for software process improvement. In: El Emam, K., et al. (eds.) Elements of Software Process Assessment and Improvement. Wiley-IEEE Computer Society Press, London (1998)