A structured approach to Evidence-based software engineering in empirical software engineering research.

M. Danz, T. Gräf, C. Michel*

Advisor: Andrei Miclaus†

Karlsruhe Institute of Technology (KIT)
Pervasive Computing Systems – TECO
*student@student.kit.edu
†miclaus@teco.edu

Abstract. TBD -> in the structured style we propose.

Keywords: TBD

1 Introduction

motivation

2 Fundamental Principles

short introduction to EBSE,..

- adopted from medicine (EBM), starting point: decisions in SE often not based on evidence for suitability, quality.., thus increasing the risk of poor decisions in motivation?
- **Evidence-based approach**: integrate all available research (evidence) in decision making process
- **Aim**: "EBSE aims to improve decision making related to software development and maintenance by integrating current best evidence from research with practical experience and human values." [8]
- **Five steps** of practising EBSE [15]:
  1. Ask an answerable question.
  2. Find the best evidence that answers that question.
  3. Critically appraise this evidence.
  4. Apply the evidence (and critical appraisal).
  5. Evaluate the performance in previous steps.

  → important tool: Systematic Literature Review (SLR)
- **SLR** [13]: identify and interpret all available literature regarding a research question → papers should be written for synthesis (TODO requirements for this, common mistakes/problems?)

– Problems inherent to SE[15]:

  1. Skill factor: performing SE methods and techniques often require skilled practitioners. This prevents blinding and can therefor cause problems related to subect and experimenter bias. (2 approaches to reduce these effects in [15])
  2. lifecycle issue: prediction of behaviour (long time?) of deployed technology difficult, hard to isolate efffects because of interaction with other methods/technologies (also 2 approaches to cope with these effects in [15])

– Step 2: SLR in SE: lack of systematic reviews (still correct? source?), lack of replication studies (source?), problems regarding SLR TODO

3 Related Work

SEED, "a preliminary empirical investigation of the use of EBSE by undergraduate students"

4 Our Approach/Guidelines

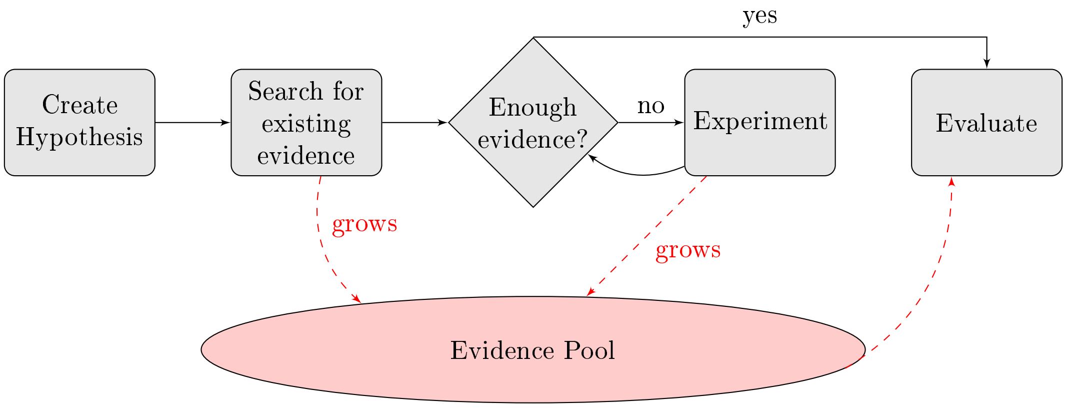
4.1 Setting up our design guidelines

The checklist (TODO name?) is meant to implicitly guide the user's approach to experimenting. By guiding the user, typical mistakes might be prevented. To create guidelines that help preventing typical users' mistakes, these mistakes first need to be identified. In this section, experiences and guidelines found in related work are discussed. The conclusions are used as basis for design of our guidelines. The first set of guidelines is based on the report of Rainer et al. [17]:

Observation	Conclusion/Guideline
"Students had problems constructing well-formulated EBSE questions." (p. 6)	Give examples for good questions to make sure the user understands a good question's scope of information. Also, explicitly list which building blocks should be contained in the question.
"Students used limited criteria for identifying the best or better evidence[...]" (p. 6)	Support decision-making to get a decision as unbiased and suited as possible. Since a decision's quality is highly dependent on the individual case, we only give a very general hint to the user. The idea is to sensitize the user to consciously prevent bias as good as possible.
"Students used a very limited number of search terms." (p. 6)	If users look for something very specific without knowing the technical term, search engines might yield better results when used with more detailed search terms. Also, synonyms or similar words might widen the search's scope to find more related work. Encourage more search terms by providing examples containing enough search terms.
"Students provided poor explanation in their reports of how their searches were conducted." (p. 7)	TODO
"Students varied in their use of the EBSE checklist." (p. 7)	Design the checklist in a way to support the user's workflow instead of hindering it. Keep it possibly simple and provide enough examples to make the user never guess an item's meaning.
"Some students critically appraise the technologies rather than the publications (evidence) on the technologies" (p. 7)	TODO Give a hint/indication?
"But we also think that the kinds of problems students were tackling [...] are not the kinds of problems researchers commonly investigate." (p. 8)	Scientific and practical evidence can have very different requirements regarding content and other aspects such as duration of evaluation. To limit this paper's scope, we focus on scientific evidence.

4.2 Checklist

4.3 Workflow



TODO:
Add numbers to each node, explain each node
Missing node: "Make Decision" at the end?
Layout/Style/Color

4.4 Study Sheet

- in this paper we propose a one page briefing of research study
- supports EBSE and searchability of study
- similar to SEED but more detailed structure to support searchability even more.
- supports researcher in understanding the field of research
- supports researcher in understanding the study itself
- guides researcher through study design and result documentation process
- contains of: research question, hypothesis, experiment/context or deduction, conclusion
- experiment contains: independent and dependent variables (control variables), method, results

For researchers to produce relevant results and understand their research domain fully, the step of developing a good research question, with a supporting hypothesis and sometimes objectives is integral [10]. These components should be carefully designed *before* conducting the study that tries to answer the question. Otherwise it is more likely to produce questions that are already answered, or *"could potentially lead to spuriously positive findings of association through chance alone."* [10, p. 280]

Research Question The question the later study is designed to answer is called research question [7]. It should be an answerable question and address a relevant issue in the research area [9]. Preceding a research question is the need for a deep understanding of the topics that have already been studied, in order to produce questions which drive knowledge further. The questions that arise during the acquisition of knowledge, and cannot be answered by means of EBSE, are likely appropriate questions for further research [10].

There are two general classes of research questions: qualitative and quantitative questions. Qualitative research states questions which report, describe, or explore a subject [6, p. 139-141]. In computer science as the research field matures these questions become more and more rare (find source). Therefore focus is on quantitative research questions in this paper. *"Quantitative research questions inquire about the relationships among variables"* [6, p. 143], and from them emerge quantitative hypotheses.

To understand the structure of research questions Shaw provides a model where she categorizes research questions from software engineering papers in five types [19] (maybe cut out).

To design a good research question Haynes coined the acronym PICO: Population, Intervention, Comparison group, and Outcome [1]. Sometimes Time is added as fifth component, when it is important over what time frame the study is conducted, see Table 1. A research question structured with the PICOT approach supports in restricting the research question and steers thereby hypotheses and study. By restricting the research question researchers can limit bias and increase the internal validity of the study, but a too narrow question may also lead to decreased external validity [10].

Before PICOT Sackett and colleagues suggested that good research questions consist out of three components: Intervention, Context and Outcome [18], which is a more coarse grained decomposition than PICOT. Dybå *et al.* displayed a fitting example for this template in software engineering: *"Does pair programming lead to improved code quality when practiced by professional software developers?"* [9, p. 60] Here the intervention is pair programming, the context of interest are professional software developers, and the outcome is improved code quality [9]. To verify the quality of a freshly designed research question Hulley *et al.* suggest the use of the FINER criteria. It highlights key aspects of the question and provides thereby new angles to view the proposed study from. The FINER criteria consists of: Feasible, Interesting, Novel, Ethical, and Relevant [10]. A more detailed view of the FINER criteria can be seen in Table 2. (TODO specify more tips for writing a good question. Creswell2014)

Hypothesis For each quantitative research question there should be a hypothesis - an educated guess about the outcome of the research question [2,10]. A good hypothesis needs to be a testable, prediction of the studies outcome, but it is important that it does not contain any interpretation [16]. A simple template for writing a hypothesis would be:

> If [I do this], then [this] will happen. [2]

Vickers *et al.* propose a more refined structure, whereas a good hypothesis needs to include three components: Two or more variables, population/context, and the relationship between the variables [7]. TODO specify the thing with the

P	Population	What specific population are you interested in?
I	Intervention (technology)	What is the investigational technology/ intervention?
C	Comparison group	What is the main alternative/baseline to compare with the intervention
O	Outcome	What do you intend to accomplish, measure, improve or affect?
T	Time	What is the appropriate follow-up time to assess outcome?

Table 1. PICOT criteria adjusted to fit better in computer science research.

F	Feasible	• Adequate number of subjects • Adequate technical expertise • Affordable in time and money • Manageable in scope
I	Interesting	• Getting the answer intrigues investigator, peers and community
N	Novel	• Confirms, refutes or extends previous findings
E	Ethical	• Amendable to a study that institutional review board will approve
R	Relevant	• To scientific knowledge • To clinical and health policy • To future research

Table 2. FINER criteria for a good research question [10]

variables more. For example a good hypothesis in software engineering research could be:

> Pair programming used by professional software developers improves code quality, in comparison to teams that use conventional techniques. (revise this)

Furthermore, when conducting empirical research - as we propose in this paper - (revise this) the hypothesis should be formulated as a *null hypothesis* H_0, and be accompanied by an *alternative hypothesis* H_1 [10]. The null hypothesis is a theory that is believed to be true but not proven jet. The alternative hypothesis is the opposite prediction of the null hypothesis [16]. At the end of the study the null hypothesis is empirically tested, and only if it is rejected (i.e., there is a significant difference between groups) the alternative hypothesis is taken as true. This confirms that effects did not show by chance alone [10]. A null hypothesis to the example above would be:

> Pair programming used by professional software developers does not affect code quality. (revise this)

To support the validity of the study even more, the hypotheses should be formulated as 2-sided hypothesis. *"A 2-sided hypothesis states that there is a difference between [groups, but without specifying the direction of the outcome]."* [10, p.280] 1-sided hypotheses should only be used when there is a strong justification for one direction of the outcome [10]. A 2-sided revision of the H_1 from above would be:

> Pair programming used by professional software developers does affect code quality. (revise tis)

(TODO specify more tips for writing a good hypothesis. Creswell2014)

Objectives Sometimes researchers define objectives to their hypotheses. They are active statements that *"define specific aims of the study and should be clearly stated"* [10, p. 280] at the beginning of research. Objectives help to define the study (e.g. helping to calculate sample size). [10,7] Although we do not include objectives in our briefing sheet we would like to mention them for reasons of completeness.

Experiment

- experiment or deduction do we include deduction?
- dependent variables
- independent variables
- control variables maybe already in method?
- method
- results

Conclusion

- is interpretation of experiment results
- verification or rejection of H_0 and acceptance of H_1
- Scope of generalization.

Question

Contains ***technology*** in a ***context*** showing an ***effect***. TODO Kitchenham Quote (practitioners)?

"Does pair programming in professional software development teams increases code quality?"

Hypothesis

Needs to contain a ***prediction*** and needs to be ***testable***.

"If you do x, then y will happen"

Experiment

Context

Dependent Variables

Variables that are ***measured*** during the study.

Independent Variables

Variables that are ***changed*** during the study.

Method

Lab-/Field study, number of participants, metrics, ...

Results

Experiment's outcome

No interpretation or conclusion!

Conclusion

Interpretation of experiment's results.

Verifying or Falsifying Hypothesis.

Scope of generalization.

4.5 Structured abstracts in software engineering

– importance: Abstracts, together with the title, are used to identify relevant research, not only in SLRs. Often the abstract is the only part of the paper that can be accessed for free. Therefore abstract and title should contain all necessary information to decide whether a paper (in case of SLR: primary study) is relevant in this context. (TODO source: "Procedures for Undertaking Systematic Reviews") → quality of abstract crucial for research, how to support researcher in writing useful abstracts? Structured Abstracts provide guidance for writer and reader.
– Suggestion of elements proposed by Jedlitschka et al.[11]:
  1. Background or Context: motivation for conducting the study, previous research
  2. Objective or Aim: Object that is studied, focus and perspective of the study, hypothesis
  3. Method: e.g. experimental design, participants and selection criteria, measurement and analyzing technique...
  4. Results: most important findings (treatment outcome), no interpretation!
  5. Limitations: scope of study, limits of generalization (often as part of conclusion)
  6. Conclusion: Interpretation of results, put results in context

  (short and early version:[12])
– About completeness and clarity of structured abstracts:
  1. Structured abstracts include more relevant information and are easier to read than conventional abstracts. [5] [4]
  2. Inexperienced authors are likely to produce clearer and more complete abstracts when using a structured form.[3]
  3. On average structured abstracts are longer and have better readability than unstructured abstracts. [14]
  4. these findings are in accordance with the ones in other disciplines (which?, source)
– guidelines for construcing structured abstract (from unstructered ones) in [14]
– use standard terminology (commonly used industry terms) [11]
– structured abstracts are longer and often size is limited (journals): prioritize traditional elements, still structured: background (one sentence), objective, method, results, and conclusion [11]

5 Discussion

i just cleaned this mess up..

References

1. Brian Haynes, R.: Forming research questions. Journal of Clinical Epidemiology 59(9), 881–886 (2006)
2. Buddies, S.: A Strong Hypothesis (2010), http://www.sciencebuddies.org/blog/2010/02/a-strong-hypothesis.php
3. Budgen, D., Burn, A.J., Kitchenham, B.: Reporting computing projects through structured abstracts: A quasi-experiment. Empirical Software Engineering 16(2), 244–277 (2011)
4. Budgen, D., Kitchenham, B., Charters, S., Turner, M., Brereton, P., Linkman, S.: Preliminary results of a study of the completeness and clarity of structured abstracts. Proc. of the 11th Int. Conf. on Evaluation and Assessment in Software Engineering pp. 64–72 (2007)
5. Budgen, D., Kitchenham, B.A., Charters, S.M., Turner, M., Brereton, P., Linkman, S.G.: Presenting software engineering results using structured abstracts: A randomised experiment. Empirical Software Engineering 13(4), 435–468 (2008)
6. Creswell, J.W.: Research Design: Qualitative, Quantitative, and Mixed Methods Approaches (2014)
7. Dr. Peter Vickers, D.M.O.: Developing a Healthcare Research Proposal: An Interactive Student Guide, http://www.health.herts.ac.uk/immunology/Webprogramme-Researchhealthprofessionals/hypothesisresearch_question.htm
8. Dyba, T., Kitchenham, B.A., Jorgensen, M.: Evidence-based software engineering for practitioners. IEEE software 22(1), 58–65 (2005)
9. Dybå, T., Kitchenham, B.A., Jorgensen, M.: Evidence-based software engineering for practitioners. IEEE Software 22(1), 58–65 (2005)
10. Farrugia, P., Petrisor, B.A., Farrokhyar, F., Bhandari, M.: Practical tips for surgical research: Research questions, hypotheses and objectives. Canadian journal of surgery. Journal canadien de chirurgie 53(4), 278–281 (2009)
11. Jedlitschka, A., Ciolkowski, M., Pfahl, D.: Reporting experiments in software engineering. In: Guide to advanced empirical software engineering, pp. 201–228. Springer (2008)
12. Jedlitschka, A., Pfahl, D.: Reporting guidelines for controlled experiments in software engineering. 2005 International Symposium on Empirical Software Engineering, ISESE 2005 pp. 95–104 (2005)
13. Keele, S.: Guidelines for performing systematic literature reviews in software engineering. In: Technical report, Ver. 2.3 EBSE Technical Report. EBSE (2007)
14. Kitchenham, B.A., Brereton, O.P., Owen, S., Butcher, J., Jefferies, C.: Length and readability of structured software engineering abstracts. IET Software 2, 37 – 45 (2008), http://www.redi-bw.de/db/ebsco.php/search.ebscohost.com/login.aspx%3fdirect%3dtrue%26db%3daph%26AN%3d30193038%26site%3dehost-live
15. Kitchenham, B.A., Dyba, T., Jorgensen, M.: Evidence-based software engineering. In: Proceedings of the 26th international conference on software engineering. pp. 273–281. IEEE Computer Society (2004)
16. Prasad, S., Rao, A., Rehani, E.: Developing hypothesis and research question. 500 Research Methods pp. 1–30 (2001), http://www.public.asu.edu/{~}kroel/www500/hypothesis.pdf
17. Rainer, A., Hall, T., Baddoo, N.: A preliminary empirical investigation of the use of evidence based software engineering by under-graduate students. 10th International Conference on Evaluation and Assessment in Software Engineering (EASE 2006) (2006)

18. Sackett, D.: Evidence-based medicine : how to practice and teach EBM (2000), http://www.ncbi.nlm.nih.gov/pubmed/12037026
19. Shaw, M.: What makes good research in software engineering? International Journal on Software Tools for Technology ... 4(1), 1–7 (2002), http://link.springer.com/article/10.1007/s10009-002-0083-4