On Implicit Assumptions Underlying Software Engineering Research

Lutz Prechelt
prechelt@inf.fu-berlin.de
Freie Universität Berlin
Berlin, Germany

ABSTRACT

Background: Software engineering research articles should make precise claims regarding their contribution, so that practitioners can decide when they might be interested and researchers can better recognize (1) whether the given research is valid, (2) which published works to use as stepping stones for their own research (and which not), and (3) where additional research is required. In particular, articles should spell out what assumptions were made at each research step. *Question:* Can we identify recurring patterns of assumptions that are not spelled out? *Method:* This is a position paper. It formulates impressions, but does not present concrete evidence. *Results:* Assumptions that are wrong or assumptions that are risky and not explicit threaten the integrity of the scientific record. There are several recurring types of such assumptions. The frequency of these problems is currently unknown. *Conclusion:* The software engineering research community should become more conscious and more explicit with respect to the assumptions that underlie individual research works.

CCS CONCEPTS

• **Software and its engineering**;

KEYWORDS

research quality, software engineering, credibility, relevance, assumption

ACM Reference Format:
Lutz Prechelt. 2021. On Implicit Assumptions Underlying Software Engineering Research. In *Evaluation and Assessment in Software Engineering (EASE 2021), June 21–23, 2021, Trondheim, Norway*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3463274.3463356

1 INTRODUCTION

The present article talks only about research works in software engineering and whithin that realm only about *empirical* works, not about tool construction as such and not about theoretical works.

That said, the progress model of scientific research is the "Standing on the shoulders of giants" idea: We make progress by building our research on top of previous research that others have performed.

EASE 2021, June 21–23, 2021, Trondheim, Norway

ACM ISBN 978-1-4503-9053-8/21/06…$15.00
https://doi.org/10.1145/3463274.3463356

For this idea to work well, it is important that we know what we know and what we do not know:

- What research questions have already been investigated? Mapping studies help us understand this.
- What have the studies found out? We might want to build on top of these findings in various ways.
- What have they not found out? We might want to do research to fill such a gap.

The latter question refers to the fact that most empirical findings are not universal; they apply to some set of contexts but not to others. Our knowledge in this regard is vague and uncertain: For a given empirical result X and three contexts A, B, C, we may be very inclined to believe (but not fully certain) X holds in A, inclined to believe (but not fully certain) it does *not* hold in B, and explicitly uncertain whether or not it holds in C.

In this article, we will call the inclination to believe a result the result's *credibility*. It forms the core of a simple model of research quality presented in Section 2. Assumptions are a key factor in that model and when assumptions are not spelled out (rather hidden or implicit) or are wrong they threaten or damage research quality.

The present article assumes (not: demonstrates) that this is a real problem: That some types of assumptions are often not spelled out, that they are often dubious or wrong, and that this is actually damaging the quality of software engineering research to a relevant degree. Section 3 explains several such types of assumption.

2 A SIMPLE MODEL OF RESEARCH QUALITY

The model explains how to think about research quality in terms of Credibility and Relevance and what role assumptions are playing. See Figure 1 for an overview.

2.1 Credibility

The Credibility of a research article is the degree to which we are willing to trust its conclusion.

For our purposes here, Credibility is a conceptual metric in the range 0% to 100%. We do not concern ourselves with the question how to operationalize it.

Conceptually, Credibility starts at 100% and is then gradually damaged by

- the fear that there may be mistakes in the raw data of the study
- the fear that some of the steps taken to process the raw data into the study's conclusions were inappropriate or not warranted.

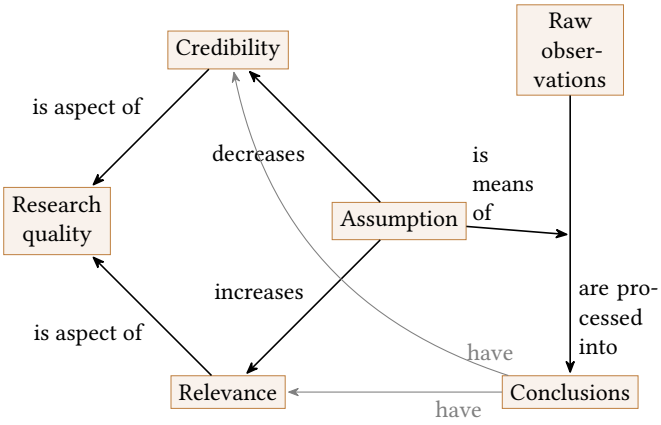

Figure 1: Core elements of a simple research quality model

2.2 Relevance

The Relevance of a research article is the value we see in having available the benefits that come with its conclusion.

Just like Credibility, for our purposes here, Relevance is a conceptual metric. Its operationalization would be even more difficult than for Credibility. For practitioners, the unit of measurement could be a monetary unit. For researchers, an appropriate unit of measurement is not obvious.

Relevance is increased by

- broad applicability of a result to many software engineering contexts,
- low difficulty of picking up and making use of the result, and
- the size of the benefit expected if one picks up and uses the result.

Note that, for a given result, these factors may look very different to a practitioner than to a researcher. Note also that each invididual *within* either group can perceive the amount of Relevance of a result very differently, depending on their engineering context or research interests.

The canonical, overall Relevance is conceptually the sum of all these perceived individual Relevances. Once again, it is highly non-obvious how one might operationalize such a metric, but even the conceptual one is helpful for thinking about research quality.

2.3 Results in the literature

Why do Credibility and Relevance refer to conclusions only? Couldn't the ideas be applied to smaller bits of evidence *within* the article?

Yes, they could in principle, but this would have two problems. First, it fragments the discussion, because then a single research work could have dozens of different levels of Credibility and Relevance at once. That is impractical for our use of those terms here. Second, internal bits of evidence is not an appropriate level of consideration for overall research quality: In practice, most references to a work refer only to statements that appear in the abstract or conclusion.

Therefore, we consider conclusion statements only, not individually the evidence that lies beneath them.

2.4 Credibility/relevance tradeoff

This difference between the raw empirical results and the conclusions drawn from them is a crucial one: The *raw* results of an empirical study are extremely narrow. By themselves, their only context of applicability is the very study itself. For giving the study Relevance, the raw results need to be generalized and that is what authors do in their conclusion steps. In the final conclusion steps they formulate what they believe we (as a community) should consider to have learned from the study. In previous conclusion steps (which in software engineering articles we allow to be sprinkled liberally over the methods, results, and discussion sections of our articles), they pave the way from the raw results to the final conclusion steps.

If they formulate their conclusions very generally, the Relevance generated by those conclusions is large; if they formulate conservatively (narrowly), the Relevance is smaller.

For Credibility, it is the other way round: Narrower conclusions are less risky and hence more Credible. Wide conclusions are less Credible – or at least they will be once we recognize that they are wide.

2.5 Categories of assumptions

When performing a study and writing an article, we unavoidably make a lot of assumptions that roughly fall into four different categories:

- Type **T:** We need to assume some amount of joint terminology from the start or else our articles would need to be 100 pages long.
- Type **M:** We need to assume some joint belief in the soundness of the research methods we are using or else we could hope for mediocre Credibility at best.
- Type **Q:** We need to assume some interest in our research question or else there is no hope for Relevance.
- Type **G:** We need to make specific assumptions when we generalize from raw results to conclusions.

Assumptions are part of the backbone of any empirical study; observe their central position in Figure 1.

2.6 Correct vs. incorrect assumptions: The R-scale

A given assumption can be correct or incorrect; an incorrect one may become correct if some additional conditions hold. Unfortunately, we know so little about software engineering that it will often be difficult to agree in which of these categories a given assumption belongs or what those additional conditions might be. Even an individual researcher might not be sure about an assumption in this respect.

It might therefore be useful to use the following vague scale of assumption quality: An assumption is either *reliable* (considered almost always correct), *reasonable* (expected to be often correct), *risky* (expected to be not-so-often correct), or *ridiculous* (considered almost always incorrect).

It may still be difficult to agree on a grade on this scale, but hopefully only between neighboring levels.

2.7 Explicit vs. implicit assumptions

Obviously, assumptions in an article ought to be clearly visible to the reader. And some of them indeed are obvious in this sense: If, for example, a tool works for Java programs only, this fact will be mentioned somewhere in the article, all readers will be interested in the fact, and nobody will perceive the situation wrongly. All statements about benefits of the tool, unless they generalize explicitly, will *assume* that one is talking about Java programs and this is alright.

Other assumptions are less conspicuous by themselves, but are made explicit in the article text by authors making an effort to do so – ideally in a subsection labeled 'assumptions'.

Still other assumptions remain implicit. They are never mentioned, not even alluded to, and only readers who are attentive and critical will be able to detect them. Such assumptions create the danger that authors and readers greatly over-estimate the Credibilty and/or Relevance of a work. This effect, by virtue of our coarse granularity of referencing results (Section 2.3), threatens to distort the scientific record.

How can we make assumptions explicit? Assumptions of types M and, to a lesser degree, T can be made explicit by suitable literature references. Making type-Q assumptions explicit[1] is one task of an Introduction section. Making the type-G assumptions explicit is less straightforward and should receive more attention of authors and reviewers than it usually does today.

2.8 Why do assumptions stay implcit?

Why are implicit assumptions not made properly explicit¿ There is a long list of possibilities; here are some of them.

2.8.1 *If the author is aware of the assumption.*

- **A1.** She decides the explanation would consume to much space or distract too much.
- **A2.** She forgets to describe it.
- **A3.** She decides to omit it to make her article look more impressive.

Such omissions can be anywhere from totally harmless to extremely damaging.

2.8.2 *If the author is not aware of the assumption.*

- **U1.** She does not recognize the assumption because it was made so early in the research process that she has since forgotten it.
- **U2.** She has somehow never recognized the assumption although she could have. If somebody points out the assumption, it immediately turns into an aware assumption.
- **U3.** She has never recognized the assumption because it appears to be an unquestionable fact, not an assumption at all. Such assumptions are more or less invisible, not just to this author but to most people in the field.
  When somebody would call such an assumption an assumption, it would extend the R-scale to the left: *religious* assumptions that cannot even be seriously questioned.
  Such assumptions occur in Kuhnian "normal science" (phase 2 of a paradigm shift [2]) as cornerstones of the current research paradigm. They are recognized as assumptions only during crisis (phase 3) and then perhaps overthrown in a scientific revolution (phase 4).

Just like aware assumptions, unaware ones can be anywhere from harmless to extremely damaging and we as a community should make a serious attempt at bringing them to light.

3 SOME RECURRING TYPES OF ASSUMPTION

3.1 The "being interesting is enough" assumption

Fred Brooks has pointed out that there is an important difference in the pecking order of outputs between science on the one hand and engineering on the other: Science aims at producing knowledge, whereas in engineering the knowledge is instrumental and the final aim is producing useful artifacts [1].

That means an engineering contribution that produces knowledge only, not artifacts (which for empirical work is a common case), has to explain how that knowledge can be used to eventually produce better artifacts. Without such an explanation, the Relevance of the contribution remains low; such contributions confuse engineering with science.

So we must not assume that being "interesting" is a sufficient justification for a software engineering contribution.

This type of problem is often found in works of the "mining software respositories" [4] type.

3.2 Overgeneralization

When trading off Relevance against Credibility as discussed in Section 2.4, authors may assume the results generalize far more than they actually do, thus increasing the supposed Relevance, but resulting in invalid conclusions.

If the underlying assumptions are explicit, this will proportionally reduce the contribution's Credibility. If they are not, it will distort the scientific record instead.

Fictitious example: Assume we have performed the following controlled experiment. 42 student subjects from University U; 2 pairs of toy programs of about 300 lines of code each; the experiment compares program variants with vs. without some design pattern; it measures the time a subject takes to finish a program extension task correctly. Subjects finished the task 16% faster ($p = 0.03$) for the program with (vs. the one without) the Observer pattern. Subjects finished another such task 29% faster ($p = 0.005$) for the program with (vs. the one without) the Decorator pattern.

Then an acceptable conclusion may look like this: *"For subjects with similar background as ours, if programs use the Observer or Decorator patterns this can help the subjects finish program extension tasks faster — at least for programs that are as small and clean as our toy experiment programs."* This is reasonable, but also humble.

If authors overgeneralize instead, we might get for instance this invalid, but much flashier-sounding conclusion: *Programs using design patterns are 16% to 29% faster to maintain than equivalent programs that do not use design patterns.*

Doesn't that sound familiar?

[1] Implicit type-Q assumptions ought to be harmless, because the readers ought to be able to decide the relevance of a question. In practice, this does not always work well; see Section 3.1.

3.3 Zero-cost assumptions

Some studies show benefits, but assume that the cost for obtaining these benefits is negligible.

Fictitious example: A tool analyzes source code and points out various classes of potential defects ("bugs"). The precision of its warnings is shown to be 50%. Such studies often silently assume that each of these defects is worth analyzing and understanding.

This assumption ignores the cost of analyzing those defects for which one then decides they are not worth fixing. It also ignores the cost of recognizing that a false positive is indeed a false positive. By ignoring these costs, the study overestimates the benefit of the technique and thus exaggerates its Relevance.

3.4 Ideal-behavior assumptions

Some studies assume that the user of a tool will always do the right thing with the tool.

Fictitious example: For the defect-finding tool postulated in Section 3.3 above, such studies ignore the possibility that a user considers to be a defect what in fact is a false positive. They will then "fix" code that is correct and break it, triggering potentially expensive subsequent failures and debugging work.

By ignoring these costs, the study overestimates the benefit of the tool and thus exaggerates its Relevance.

3.5 This-means-what-I-need assumptions

Some studies apply the most favorable interpretation of some measurement, ignoring one or more alternative interpretations. In particular, some studies assume, after observing a certain correlation, that some plausible cause-effect relationship is indeed solely responsible for that correlation.

Fictitious example 1: A study determines the fraction of methods that exhibit the "long method" code smell [3] for 100 Java projects and compares that to the same measurement on 100 Python projects. It finds the Java projects have a significantly lower fraction of such methods. It concludes that Java developers care more about the code than Python developers do.

This explanation ignores the alternative explanation that it is a large number of trivial one-line getter and setter methods in the Java projects that is driving the fraction down.

Fictitious example 2: Same study as above, but now with the opposite finding. The Java projects have a *higher* fraction of methods with the "long method" code smell. The study now concludes that *Python* developers care more about the code. The explanation ignores a number of caveats:

- Java is more verbose, so that applying the same threshold for long methods in both languages may be inappropriate.
- Whether a method of X lines length should be considered smelly depends on how linear the control flow is and that may be different in the Python projects versus the Java projects.
- Binary classification into only smelly vs. non-smelly might be inappropriately crude, e.g. if one language has fewer offenders, but far worse ones.

The possibilities for simplistic or one-sided interpretations of measurements are nearly endless.

3.6 I-make-no-assumptions assumptions

Many authors of articles that involve machine learning techniques, in particular neural networks, appear to assume that their model assumes nothing, although in reality those techniques compute regression models, which has lots of implications [5, Chapter 8].

A more realistic attitude would be that we know little about what properties our data have with respect to the neural network model and which properties of the model are relevant in which ways. Put differently, we have a hard time spelling out what the assumptions even are that we are making when we apply such a model.

4 SO WHAT?

It is my impression that implicit assumptions are frequent, that many of them are risky, that some are ridiculous, and that all this is needlessly reducing the value of software engineering research overall.

I believe that software engineering research could benefit greatly from

- learning to uncover implicit assumptions,
- agreeing on which assumptions are considered risky and which are considered ridiculous,
- demanding that articles have to spell out their assumptions and rejecting articles in which reviewers find substantial hidden assumptions,
- rejecting articles with ridiculous assumptions, and
- demanding a critical discussion of risky assumptions.

ACKNOWLEDGMENTS

I thank Ina Schäfer and Lars Grunske. Preparing the talks to which they invited me produced the seed of the ideas presented in this article. I thank reviewer 2 for the idea for Section 3.6. I thank Paul Ralph (reviewer 1) for suggesting Section 2.8.

REFERENCES

[1] Frederick P Brooks Jr. 1996. The Computer Scientist as Toolsmith II. *Commun. ACM* 39, 3 (1996), 61–68.
[2] Thomas S. Kuhn. 1962. *The structure of scientific revolutions.* University of Chicago press.
[3] Mika Mantyla, Jari Vanhanen, and Casper Lassenius. 2003. A taxonomy and an initial empirical study of bad smells in code. In *Proceedings International Conference on Software Maintenance (ICSM) 2003.* IEEE, 381–384.
[4] MSR 2020. *Proceedings 17th International Conference on Mining Software Repositories (MSR).* ACM.
[5] William N Venables and Brian D Ripley. 2002. *Modern applied statistics with S-PLUS.* Springer Science & Business Media.