On the Worthiness of Software Engineering Research

Antony Tang
Swinburne University of Technology

Rick Kazman
University of Hawai‘i

Abstract

We reflect on the worthiness and impact of software engineering research and we observe that much of the research output by the academic community brings little value to SE practitioners. To understand the situation, we analyze the motivations behind conducting SE research, and the issues that prevent SE research from becoming more valuable and impactful. To address some of these issues, we propose five criteria to help SE researchers reflect on their research and align their research goals with practical needs. Finally we provide some concrete suggestions on how to improve the quality of SE research and how to better transition research results to practice.

1. Introduction

The “Impact Project” tried to assess the impact of software engineering (SE) research on SE practice [1]. Some of its findings indicated that (1) SE research has significantly affected practice; (2) lasting impacts require ongoing interactions between research and practice; (3) research impact might not be felt for at least 10 years. Whilst this study reflected the bright side of SE research, there is also a not-so-bright side.

Thousands of SE research papers are published in conferences and journals each year\(^1\). However, feedback from SE practitioners indicates that the vast majority of this research is not useful to them. In this article, we discuss the motivations behind SE research. We note issues that prevent research outputs from becoming more useful. We discuss criteria for evaluating the worthiness of such research from a practice perspective. We hope that this article will start a frank discussion on the worthiness of SE research to practitioners and directions for the future.

2. The Value of SE Research

Researchers may claim “novelty” for their ideas but such claims do not necessarily establish a contribution or value. SE is an engineering discipline: an applied science. The value of SE research must therefore be judged by benefits to practitioners. We employ the Redwine-Riddle Maturation Model [2] to analyze two value-chain scenarios between researchers and practitioners (see Figure 1).

---

\(^1\) We counted 85 conferences from [http://web.engr.illinois.edu/~taoxie/seconferences.htm](http://web.engr.illinois.edu/~taoxie/seconferences.htm). DBLP ([http://dblp.uni-trier.de/](http://dblp.uni-trier.de/)) yields many more that could be classified as SE.
This maturation framework [2] suggests that SE research should go through phases of (0) Basic Research; (1) Concept Formulation; (2) Development/Extension; (3) Enhancement/Exploration; (4) Popularization/Propagation. We map SE research into this model to understand where and how value is passed.

Successful SE research—that positively influences the community—is often popularized and transitioned via working with intermediaries and practitioners, by creating appropriate tools, trialling ideas, training practitioners etc. (Figure 1, Scenario B). Several innovations over the past two decades have profoundly affected practice (Agile methods, cloud computing, relational/NoSQL databases, IDEs, the WWW, DevOps, SOA, open source, design patterns, etc.) and have had some of their roots in academia. However, most SE research is never communicated to or used by practitioners (Figure 1, Scenario A). For example, in a study on architecture description languages, researchers found that most have no practical applications [3].

3. Gaps between SE Research and Practice

We conducted a survey of software practitioners; we recruited participants from a SATURN² workshop, industry software architecture courses and personal industry contacts and received 61 responses. We asked them if they read any “research papers” from SE conferences, or journals in the last 3 years. Out of these respondents, only 19 (31%) answered in the affirmative. Amongst those 19, only 4 could name a true research journal or conference; the other 15 named trade publications. This survey suggests that practitioners are disconnected from research publications. What are the implications? First, practitioners do not actively seek out academic research results and the majority of the research results that SE researchers produce are not communicated to practitioners. It seems the most popular audience for research is other researchers, for the purpose of improving one’s H-index; second, there are few venues where SE practitioners and researchers meet to exchange ideas. In a recent public forum, the first author asked why practitioners do not work with researchers. The answer was: “Professors are interested in publishing papers only. They don’t understand real-life problems.” Similarly, Dijkstra observed that SE often does not serve its supposed purposes [4].

---

² SATURN (SEI Architecture Technology User Network conference) is an annual practitioner-oriented conference focused on transitioning software architecture research into practice.
4. Why Is There a Gap between SE Research and Software Practice?

Research in the natural sciences relies on experiments, measurements and repeatability. However, many SE studies take place in complex, unique environments making testing, measurement, and repeatability difficult. Human factors add to the variability of the results, making it even harder to draw conclusions with confidence. We observe the following issues:

1. **SE Researchers Lack Understanding of Industry's True Needs** – many researchers who emerge from graduate programs have no exposure to commercial software development. They have little understanding of the issues faced by developers in the real world.

2. **Unrealistic Assumptions** - SE researchers often fall into the trap of making implicit, unrealistic assumptions. Researchers often fail to formulate appropriate problems that need studying (Phases 1 and 2).

3. **No Knowledge of Existing Industry Solutions** – often SE researchers have little knowledge of existing solutions in the marketplace, and may be unaware of industrial practices (Phases 2 and 3).

4. **SE Research Incentive Misalignment** – SE researchers are incentivized to publish ever-more papers and garner citations for promotion and tenure. The Guardian reported: “The blame for this sad situation [peer-reviewed publishing] lies with the people who have imposed a publish-or-perish culture, namely research funders and senior people in universities. To have ‘written’ 800 papers is regarded as something to boast about rather than being rather shameful.” A similar comment was made by Vardi [5]: “if 10 papers are better than five, then surely 15 papers are better than 10”.

As a result, SE research suffers from the following issues: **Irrelevance** – a research output has little relevance to software practice. **Disregarding practical constraints** – there are real-world constraints that could make the proposed research impractical. **No understanding of the application context** – some SE researchers invent solutions without considering how they can be used in real-world contexts. **No attempt to evaluate costs, benefits, and risks** – an engineering solution is useful if the benefit gained is significantly greater than the cost. There is little discussion in research of the full costs of a technique (including acquisition, training, and disruption to existing processes). There are seldom arguments made for why the benefits are greater than the costs and how this technique will reduce project risk. **No attempt to evaluate the usability** of the research. Often the only users of the technique are the researchers themselves, and they have made scant attempt to empirically evaluate the technique by practitioners. Some of these same criticisms are found in [6]

5. SE Research Evaluation Criteria (5Cs)

We suggest using 5 criteria (5Cs) to help SE researchers align their research goals with practical needs, based on previous ideas such as [7]:

- **C1: Relevance** - What is the relevance and significance of this research results? A researcher should specify the conditions under which the research results would be meaningful, such as the conditions when the tools or methods could be used.

---

3 http://www.theguardian.com/science/2011/sep/05/publish-perish-peer-review-science
- **C2: Practicality** - Does a method/technique help solve a practical software development issue of significance? Have people other than the inventor(s) been shown to successfully use the method/technique? Here we suggest that a researcher needs to understand and justify the practicality of their research.

- **C3: Novelty** - Does this research results duplicate existing solutions? Researchers need to justify that novel solutions are not minor variants of each other, or minor variants of what exists in commercial products.

- **C4: Beneficiaries** - Who would benefit from this work, and how does the research results have a positive impact on the beneficiaries? SE researchers need to specify the key stakeholders who could benefit from their research.

- **C5: Costs and Benefits** - What are the costs and benefits of applying a method/technique? Is it measureable? The benefits of applying the research results must significantly exceed the costs when being applied.

We realize that not all criteria are applicable to all research. For instance, some research aims at understanding software development phenomena, therefore cost-benefit criteria does not apply. And some SE research may be theoretical, and may not be directly applicable by practitioners.

6. Conclusion

There are gaps between SE research and practice. Our academic culture does not focus on how SE research can aid practice. We need to fix this. As Vardi discussed [5], scholarly inflation is detrimental to computing research. SE researchers need to take a serious look at their research, and evaluate the long-term value they are creating. This is essential for the research community to thrive in the long term: we must provide value to software practice, and not be content with paper and citation counts. Here are a few steps we can take:

- Get more experience in the software industry. Researchers can spend time embedded in software development teams in industry sabbaticals, get some development experience before undertaking research roles, or participate in industry meetups and showcases.
- Put more emphasis on empirical research in collaboration with industry partners (perhaps through collaborative projects funded by agencies and supported by industry) and new ways to assess of research impact by university management and funding agencies.
- Consider how to evaluate research, using the 5Cs to assess impact on the community.
- Improve communication and knowledge transfer to software developers. In addition to reporting our results in a scientific manner, we should translate them into lay language, and disseminate summaries to practitioners.

We welcome the Computing Research Association’s advice to hiring units to focus on quality and impact [8]. But we fear that “paper counting” still prevails. There are challenges in changing the culture and this will not happen overnight. We hope that this article will germinate meaningful discussion and honest reflection to help us improve this situation.
7. References


