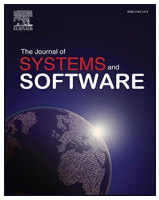
The Journal of Systems and Software 222 (2025) 112318

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss



Letter to the Editor

Dear researchers step 1: Find a team with a problem

1. Introduction

I have been working in the software engineering field for over 30 years and during that time the gap between software engineering research and practice has continually puzzled and troubled me. I have tried to bridge the two communities, through academic publishing and involvement with both academic and research events, but frankly, I have had no success in improving the situation.

There are a few bright spots of cooperation, or at least awareness, such as the research track at the annual XP conference and the occasional researcher speaking at large practitioner conferences like QCON, but in general it appears that the two communities exist largely independently and, if anything, I perceive the gap to be getting wider rather than narrower.

A good example of the disconnect is the topic of technical debt. Industry is swimming in technical debt, as any practitioner will tell you and the constant reference to the problem at industry events confirms. There is an active research community in academia studying technical debt and creating ever more sophisticated models and approaches for understanding, and potentially addressing, technical debt. And yet the two appear to be almost totally disconnected.

To investigate this a little further, I scanned the web sites of the well known TechDebt conference series (https://conf.researchr.org/series/TechDebt) and noticed that at TechDebt 2022 there were two industrial people on the technical paper programme committee, and 2 of 14 papers appeared to have significant industrial input. Then at TechDebt 23 it was 0 of 31 on the committee and 2 of 14 again for the papers, and at TechDebt 2024, 2 of 36 on the committee and 5 of 15 papers. So while there is some industrial participation at this conference, it doesn't look as if industry is a major part of it. And I have yet to come across more than a handful of practitioners who have heard of it (and those who had heard of it are industrial and academic crossover people like me).

I know that some of the people with academic affiliations may have worked in industry and there may be meaningful industrial collaboration hiding within the programme. But it doesn't look like an event with a lot of industrial participation.

It is also possible that, although the conference does not have direct industry involvement, the research results reported here are being transferred to industry. But I can't find any evidence to suggest that is true. If significant transfer to industry is happening, then I think there would be published work reporting this. I cannot be sure why such transfer to practise is not happening, but I suspect it is a result of poor communication between the research and industrial communities, the research community not knowing what they need to do to achieve industrial transfer of results, mismatched value systems between the two communities and perhaps the research results being perceived of limited value by practitioners. I discuss these difficulties more later in this article.

I want to make it clear that I am not criticising this conference specifically and I am aware of the many difficulties of getting researchers and practitioners together meaningfully at a single event. However I think this is a good example of the problem we have. A topic that is of interest to both industry and researchers where the two communities appear to be quite disconnected.

In this short article I will provide my perspective on this problem and I have a few suggestions on how we might start to address it.

2. The gap between research and practice

Many of the reasons for the gap between research and practice were eloquently described by Titus Winters in a previous column in this series (Winters, 2024) and I will not state all of that again. I agree with everything that he says in that article.

However just to provide another perspective I spent some time looking at the ECSA and ICSA software architecture conferences. Both are events that I have been involved with over the years and are focused on my favourite software engineering topic - software architecture.

I know that these are conferences that have tried to achieve industrial collaboration for many years, having industrial tracks, inviting industrial keynotes and trying to attract industrial practitioners onto their programme and organising committees. So they are certainly aware of the industry and research gap and are trying to bridge it.

However, when you review the programmes of ECSA and ICSA for 2022, 2023 and 2024, they seem to suffer from many of the same problems as the technical debt conference I mentioned earlier.

Firstly the good news is that they both have an industrial track, both have a few practitioners on their programme committees, some of the papers are industrial case studies, and quite a few of the papers relate to industrially relevant topics such as microservices, architectural decisions, AI, security and sustainability. All topics that you will see at current industrial conferences.

However when you dig a little further …

- The *practical motivation* for many of the papers isn't very clear. In most papers the motivation is logical and clearly described but it is unclear that any industrial team has ever identified this as a problem and it is very rare that any industrial contact to validate the motivation is reported.
- *Validation* of the method, tool or idea is typically attempted by testing small example systems, or mathematically, via small well defined abstract examples, or via practical testing using open source systems by the researchers themselves. Many of the systems and examples used for validation are very simple by industrial standards and they rarely contain the sort of problems, such as technical debt

https://doi.org/10.1016/j.jss.2024.112318

Available online 5 December 2024

and operational complexity, that practitioners face on a daily basis. It is also very rare to find a paper reporting validation in partnership with any practitioners.
- There is rarely any mention of *technology transfer, industrial application or even publicising results* of the research to the practitioner community.

This is not to criticise what are generally well written papers, reporting methodologically sound, potentially interesting pieces of research work. However, given these factors, it is difficult to see how the hard work of the researchers is ever going to have an impact on software architecture practice.

Of course I am not suggesting that all research papers will ever be, or should be, of direct interest to practitioners. Early stage research is important and necessary and some research may be pursued to support the research process rather than to solve practical problems. My concern is that nearly all research papers, at two quite industrially aware conferences, are unlikely ever to be used by an industrial software team, unless a lot more work is done to make this happen (and I don't see any evidence that this work is being done).

3. Some immediate advice

A few specific actions that I think researchers could usefully do, if they are interested in having their research applied to practice, are:

- In the paper explain *where the motivation came from*. As well as finding motivation from the research literature, is there an industrial motivation? Do you think many software teams have this problem? Why? Did you talk to a friend in industry? See a video from a practitioner conference? Read a blog that revealed the problem? Discovered it from a survey of industrial practice? If not, where did the motivation for the work come from?
- As part of your research consider *what a practitioner would need to know and to have to apply it*. Consider making code or data available via Github or a similar site (as some researchers do already). Consider if you can align with or extend existing widely used frameworks and libraries.
- When you are validating your work consider if there is any *industrial input* you could get to augment your existing plans. Could a practitioner review your validation plan and make suggestions? Do they find your validation results convincing? Could you even get a practitioner, perhaps an acquaintance or alumni, involved in the process in some way?
- In your Further Work section explain how you plan to *publicise the work* and encourage pilot use by a targeted practitioner community. Perhaps you can submit a talk to a practitioner conference, or give a talk at a developer community event, or consider integrating it with an open source project or publicising it via a social media campaign? Or something more imaginative!

I know that there can be difficulties with all of these ideas. It can be hard to get practitioners interested and many researchers are not natural "sales" people. The extra effort to package and release something usable may not be rewarded with a publication and citations. Publicising your work takes time and effort (and if it gains attention not every reaction to it will be positive). Even if you do these things not all of them will be successful. However I think these are accessible initial steps towards industrial application and the process of publicising your work might result in finding industrial practitioners who are interested in collaborating with you.

4. In the longer term

As I outline above, I think that the fundamental reason for the gap between research and practice stems from how researchers find problems to work on. A lot of research builds on existing research and extends it and fills in gaps. This is satisfying for the research community involved, but doesn't help industry if that research area is not of interest to many practitioners. Other research appears to start because the researchers either perceive a problem with building software, which industrial teams may or may not have, or are simply interested in answering a question which intrigues them. Neither of these strategies is without merit, but realistically neither is likely to result in a researcher producing a compelling result for a practising software engineer.

So how do we find good research problems to work on?

It may be easier said than done, but I think the answer is to find a software delivery team that has a problem and work on that. Not all of the problems that software teams have will be interesting research topics, but I'm pretty sure that some of them will be.

I believe that there would be several advantages from this approach:

- We know that the research problem we are working on has some *value* to at least one team and in all likelihood there are many similar teams in the world with a similar problem.
- We can work to understand the *priorities and context* of the team, allowing research results to be tailored to maximise their chances of success.
- We can work with the team during the research process to *iteratively validate* and extend the work of the researchers in valuable directions.
- We have a team who are likely to be prepared to use the research outputs and so *validation is in an industrial context* rather than a purely research context.
- We have a definite *context for the work* which we can report in the research literature which helps to guide readers as to its likely applicability (for example a research result which is successful in embedded safety-critical teams may well not be equally applicable to mobile development for a fintech firm).

Naturally there are also some difficulties with this approach, otherwise we would probably all be doing it already. I think the main complications are:

- *Finding the team* may not be easy for some researchers. However many university departments do have links to industrial groups via graduate recruitment, industrial liaison groups and their alumni. The other place to look for teams is within the research environment. Within the research environment there are many people developing software, some of it quite large and complex. Some of these teams are developing software to support research (perhaps in medical research or biochemistry) others are developing software to support administration (such as educational technology teams). Could one of those teams be an option?
- *Selecting the right problem* also needs some thought. It needs to be a problem which is valuable and interesting for the research team, ideally with a good degree of transferability, and something that will have lasting research value. However it also needs to be of short term value to the team involved. Identifying a good problem needs some effort and collaboration from both sides and the research team needs to get to know the delivery team well enough to have some idea of their needs, priorities and constraints.
- The differing *timescales* of industrial software delivery and research can also cause problems. In many software teams with interesting problems things change rapidly. It is easy to select a problem which is very relevant today, but becomes irrelevant in 9 months time when a major decision is taken (e.g. to rewrite or decommission a software module).
- What is seen to be of *value* in the two environments is also different and has caused frustration and misunderstandings over the years. Researchers need research results reported in peer reviewed publications to drive citation counts, which are assumed to be a proxy for

"impact". Software teams need a problem solved so that they can deliver faster or more efficiently for their stakeholders, and most of them are totally uninterested in research publications. It takes goodwill and cooperation to balance these competing needs.

- *Prioritisation and time allocation* from the industrial team can also cause friction in the research process. While a team may, in principle, be interested and supportive of some applied research in their project, are they prepared to prioritise it sufficiently to spend time working with the researchers? This probably requires stakeholders outside the team to be convinced as to the value of the exercise and may mean that the software team or the researchers have to "sell" the idea to non-technical people like product owners and managers in the business, which can be difficult.
- Finally it is important that the *researcher's time is valued* for doing research. Researchers often have good practitioner skills and so there is a danger that they get sucked into the project as "cheap labour" and end up performing work that doesn't directly address the goals of the research.

5. What if you can't find a team?

However I would suggest that even if you can't find a real team to work with, visualising the team who you hope would use your research is a valuable technique. Find case studies or invite industrial people in to give talks on their work and their concerns. Go to some industrial conferences and get to talk to people. Perhaps approach companies to see if you can "shadow" a team for a short period. Any of these activities will help you to understand at least one industrial team and in most cases, a lot of the knowledge will generalise to be applicable to many teams.

If you need to "work with" an imaginary team, use the approach we use in industry when thinking through our stakeholders - bring them to life with realistic scenarios. What is the team's name? How many people are in the team and what are their names and roles? What are the team's ways-of-working? What are they building? Who are their stakeholders and what pressures are the team under? What difficulties do they have day-to-day? What are their priorities? What are the next three big features they need to deliver and what are the next 10 user stories that contribute to those features?

The book *The Phoenix Project* (Kim et al., 2014) is an example of using a fictional but realistic team to consider how to change industrial practice.

Again see if you can get feedback. Describe the team and ask industrial acquaintances or alumni to review it for you. Perhaps they will be prepared to rework parts for you to make it more realistic. Ask a team from an industrial research lab to review it and comment on how realistic it is in their experience. Again they might be prepared to collaborate on improvising it. Who knows? You might get some sort of citable publication out of the process!

This may seem to be a time consuming distraction from the "real work" but by helping you to think through how a realistic team works, what their real concerns are, and what they are likely to value, it can help you to view your research through the lens of a practitioner.

6. Conclusions

All over the world, countless very accomplished software engineering researchers are working hard to produce methodologically sound, well presented research results. However the reality appears to be that very little of their work comes into the consciousness of any software practitioners and even less is ever applied to an industrial project. This means that there is a significant gap between software engineering research and industrial software practice, which is wasteful and a missed opportunity.

Some immediate things that I think researchers can do to make their work potentially usable in industry would be to be clear about where the motivation for the work came from and how this relates to practise, consider what a practitioner would need to know and have to use the work, try to get some industrial input, if not full collaboration, on the problems being solved and the validation performed, and finally consider how to publicise the work in such a way that it catches the attention of the target practitioner community, through events, social media and open source.

In the longer term I suggest that a lot of software engineering research would benefit from being considered in the context of a software team in a specific domain with some specific problems. This allows the context and applicability of the research to be clear, provides clear motivation, ensures that the priorities and difficulties of the team are reflected in the research, increases the likelihood of some transferability if the research is successful, and provides some interested industrial collaborators who want their problem solved.

By ensuring that we think about context and relevance early in the research process, and find industrial collaboration and validation wherever we can, we may start to bridge the gap between software engineering research and practice, maximising the chances that the hard work of researchers can have a meaningful impact on the software industry

Declaration of competing interest

None.

References

Kim, G., Behr, K., Spafford, K., 2014. The Phoenix Project: A Novel About IT, DevOps, and Helping your Business Win. IT Revolution.
Winters, T., 2024. Thoughts on applicability. J. Syst. Softw. 215, 112086.

Eoin Woods
Endava and Imperial College London, London, UK
E-mail address: contact@eoinwoods.info.