Research Statement

My research falls into what I call the triangle of empirical software engineering. My first research projects focused on a specific observation I made while working as a software developer in industry: the widespread usage of informal diagrams without proper tool support. I empirically studied the problem [C1] and developed tool prototypes to address the identified issues [S2, S6]. I then moved towards interdisciplinary topics related to the legal implications of code plagiarism [J1] and knowledge transfer from psychology [C6]. While integrating an interdisciplinary perspective, the studied phenomena were always rooted in central software engineering problems and were thoroughly empirically studied before proposing solutions in form of tool prototypes or process adaptations.

The guiding theme of my research has always been the idea that thoroughly analyzing and understanding the state-of-practice is an essential first step towards improving how software is being developed. Too often, we still see decisions in software projects being rather based on opinions than being informed by empirical data. My industry experience – in the past as a student as well as recently after moving back from Australia to Germany – helps me to consider both the perspectives of researchers as well as practitioners. My research focuses on problems that I consider relevant for software developers in practice, but in addition to that I also put effort into communicating results back to practitioners by giving talks in user groups, companies, as well as other channels such as social media.

The problems I studied were, however, not exclusively rooted in software development practice. Early in my PhD research, I noticed potential ethical issues of common practices in the software engineering research community [S5] as well as a lack of knowledge when it comes to central aspects of empirical research in general [P1]. Therefore, I am part of ACM SIGSOFT’s Empirical Standards initiative that maintains a continuously evolving collection of guidelines for the various empirical methods being utilized in software engineering research.¹

I consider myself a methods pluralist. To complement qualitative results derived from interviews [C1], observational studies [C2], or open-ended survey questions [C1, C6], I apply data-mining techniques to open source software projects [J1, W3] or other data sets [C5, J2, S8], including data shared by companies under non-disclosure agreements [C7]. I further maintain the open dataset SOTorrent that other researchers are utilizing to study the origin, evolution, and usage of Stack Overflow content. This dataset was selected as the official mining challenge of MSR 2019. I am also interested in information visualization and visual analytics [S1, C3], exploring how interactive visualizations can support humans in analyzing data. I regularly develop custom visualization that we have been using in different research projects to explore data or to derive patterns [S8]. I follow open science and open data practices, meaning that I try to publish data, software, analysis scripts, and paper preprints whenever possible.

My vast methodological experience allowed me to react quickly when more and more developers were forced to work from home due to the COVID-19 pandemic. Together with a colleague from Canada, I initiated one of the first global studies on the impact of forced remote work on the productivity and wellbeing of software developers, including an assessment of potential support strategies that organizations can employ [J5].

¹https://github.com/acmsigsoft/EmpiricalStandards
Outlook

Against the above-mentioned background – and my previous work that has been published in highly ranked venues including FSE, ICSE, TSE, and EMSE – I see three potential avenues for my future research.

The first avenue is centered around the Google Faculty Research Award that was awarded to myself and two colleagues from the University of Adelaide. In the corresponding project, we are exploring automated techniques for making software documentation more consistent and inclusive. While we currently focus on improving the natural language used in documentation resources, the more general question is how software documentation ecosystems can be made more accessible, not only for newcomers but also for experienced developers seeking information during their daily work. A first step into that direction was our work on the information diffusion between Stack Overflow and other documentation resources via hyperlinks [12], which intends to inform first tool prototypes.

A second avenue is based on the idea of developing an holistic view on software engineering across the lifespan. While there is research on teaching programming to children as well as research on specific challenges of older software developers [14], there is no holistic framework yet that integrates existing results from various disciplines such as software engineering, learning sciences, and in particular cognitive and developmental psychology. The goal would be to structure the current body of knowledge along the typical human lifespan – from learning programming as a child to keeping older developers in the workforce. Having such a framework would allow us to identify knowledge gaps, helping us to steer future research, and inform stakeholder in education as well as professional software developers throughout their careers, where role transitions and continuous learning are prevalent in the fast-moving software industry [C6, J4].

A third avenue is the emerging research field of how the current global pandemic shapes software development processes and tools of tomorrow. While we contributed initial results based on data from an early phase of the pandemic, the next milestone is transitioning from the current setting dominated by remote work to the “new normal” setting with hybrid and remote work modes, posing additional challenges. The agile manifesto states that “the most efficient and effective method of conveying information to and with a development team is face-to-face conversation”, 2 a statement which shows how deeply rooted the agile work culture is in on-site work. I intend to utilize my own experience working in a large-scale agile project that moved from on-site to a remote work, exploring effective ways to preserve the core values of agile software development in the remote and hybrid settings that will accompany us in the years to come.

Beside the above-mentioned avenues, I am constantly monitoring online software development communities and I regularly talk to my industry contacts to learn about specific challenges faced by software engineers in practice. I then use those insights to steer my short- and long-term research directions. Such feedback channels are particularly important, because in the last decade, too many trends in the software industry were studied with a significant delay by software engineering researchers, leading to missed opportunities in providing empirically founded guidelines for practitioners early on.

---

2 https://agilemanifesto.org/principles.html
Publication List

DBLP: https://dblp.uni-trier.de/pid/145/3950.html
Google Scholar: https://scholar.google.com/citations?user=xO09KrYAAAAJ
Website: https://empirical-software.engineering/publications/

★ Among the five most important publications

Journal Papers (peer-reviewed)

Empirical Software Engineering (EMSE 2020).
https://empirical-software.engineering/publications#emse20-pandemicprogramming

Sebastian Baltes, George Park, and Alexander Serebrenik.
https://empirical-software.engineering/publications#ieee-sw20-ageing

Oliver Moseler, Felix Lemmer, Sebastian Baltes, and Stephan Diehl.
https://empirical-software.engineering/publications#jss20-formulas

[J2] Contextual Documentation Referencing on Stack Overflow.
Sebastian Baltes, Christoph Treude, and Martin Robillard.
IEEE Transactions on Software Engineering (TSE 2020).
https://empirical-software.engineering/publications#tse20-condor

[J1] Usage and Attribution of Stack Overflow Code Snippets in GitHub Projects. ★
Sebastian Baltes and Stephan Diehl.
https://empirical-software.engineering/publications#emse19-snippets

Conference Full Papers (peer-reviewed)

[C7] Automated Query Reformulation for Efficient Search Based on Query Logs from Stack Overflow.
Kaibo Cao, Chunyang Chen, Sebastian Baltes, Christoph Treude, Xiang Chen.
Acceptance rate: 23% (138/602).
https://empirical-software.engineering/publications#icse21-reformulation
Towards a Theory of Software Development Expertise. ★
Sebastian Baltes and Stephan Diehl.
*Proceedings of the 26th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2018).*
Acceptance rate: 21% (61/289).
https://empirical-software.engineering/publications#fse18-expertise

SOTorrent: Reconstructing and Analyzing the Evolution of Stack Overflow Posts. ★
Sebastian Baltes, Lorik Dumani, Christoph Treude, and Stephan Diehl.
*Proceedings of the 15th International Conference on Mining Software Repositories (MSR 2018).*
Acceptance rate: 33% (37/113).
https://empirical-software.engineering/publications#msr18-sotorrent

Natalie Stors and Sebastian Baltes.
*Proceedings of the ACM on Human-Computer Interaction, Vol. 2, Issue CSCW, Article 166 (PACMHCI/CSCW 2018).*
Acceptance rate: 26% (185/722).
https://empirical-software.engineering/publications#cscw18-airbnb

Tanja Blascheck, Fabian Beck, Sebastian Baltes, Thomas Ertl, and Daniel Weiskopf.
*Proceedings of the IEEE Conference on Visual Analytics Science and Technology (VAST 2016).*
Acceptance rate: 32% (50/157).
https://empirical-software.engineering/publications#vast16-codingtool

Navigate, Understand, Communicate: How Developers Locate Performance Bugs.
Sebastian Baltes, Oliver Moseler, Fabian Beck, and Stephan Diehl.
*Proceedings of the 9th International Symposium on Empirical Software Engineering and Measurement (ESEM 2015).*
Acceptance rate: 25% (20/81).
https://empirical-software.engineering/publications#esem15-debugging

Sketches and Diagrams in Practice. ★
Sebastian Baltes and Stephan Diehl.
*Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2014).*
Acceptance rate: 22% (61/273).
https://empirical-software.engineering/publications#fse14-sketches

Conference Short Papers (peer-reviewed)

Code Duplication on Stack Overflow.
Sebastian Baltes and Christoph Treude.
*Proceedings of the 42nd International Conference on Software Engineering (ICSE 2020).*
Acceptance rate: 30% (28/93).
https://empirical-software.engineering/publications#icse20-clones
Sebastian Baltes, Christoph Treude, and Stephan Diehl.
*Proceedings of the 16th International Conference on Mining Software Repositories (MSR 2019).*
Acceptance rate: 33% (1/3).
https://empirical-software.engineering/publications#msr19-sotorrent

[S6] **Round-Trip Sketches: Supporting the Lifecycle of Software Development Sketches from Analog to Digital and Back.**
Sebastian Baltes, Fabrice Hollerich, and Stephan Diehl.
*2017 IEEE Working Conference on Software Visualization (VISSOFT 2017).*
Acceptance rate: 59% (10/17).
https://empirical-software.engineering/publications#vissoft17-livelysketches

[S5] **Worse Than Spam: Issues In Sampling Software Developers.**
Sebastian Baltes and Stephan Diehl.
*Proceedings of the 10th International Symposium on Empirical Software Engineering and Measurement (ESEM 2016).*
Acceptance rate: 37% (23/61).
https://empirical-software.engineering/publications#esem16-sampling

[S4] **Effects of Sketching on Program Comprehension (Research Plan).**
Sebastian Baltes and Stefan Wagner.
*Proceedings of the 17th International Conference on Agile Processes in Software Engineering and Extreme Programming (XP 2016).*
Acceptance rate: 38% (5/13).
https://empirical-software.engineering/publications#xp16-sketching

[S3] **VisualCues: Visually Explaining Source Code in Computer Science Education.**
Benjamin Biegel, Sebastian Baltes, Bob Prevos, and Stephan Diehl.
*Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 2015).*
Acceptance rate: 48% (36/75).
https://empirical-software.engineering/publications#vlhcc15-visualcues

[S2] **Linking Sketches and Diagrams to Source Code Artifacts.**
Sebastian Baltes, Peter Schmitz, and Stephan Diehl.
*Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2014 Research Demos).*
Acceptance rate: 65% (15/23).
https://empirical-software.engineering/publications#fse14-sketchlink

[S1] **RegViz: Visual Debugging of Regular Expressions.**
Fabian Beck, Stefan Gulan, Benjamin Biegel, Sebastian Baltes, Daniel Weiskopf.
*Proceedings of the 36th International Conference on Software Engineering (ICSE 2014 NIER).*
Acceptance rate: 24% (35/146).
https://empirical-software.engineering/publications#icse14-regviz
Workshop Papers and Extended Abstracts (peer-reviewed)

[W4] **An Annotated Dataset of Stack Overflow Post Edits.**
Sebastian Baltes and Markus Wagner.
*Genetic and Evolutionary Computation Conference Companion Proceedings (GECCO 2020 Companion), 9th Genetic Improvement Workshop.*
https://empirical-software.engineering/publications#geccogi20-soedits

[W3] **(No) Influence of Continuous Integration on the Commit Activity in GitHub Projects.**
Sebastian Baltes, Jascha Knack, Daniel Anastasiou, Ralf Tymann, and Stephan Diehl.
*Proceedings of the 4th International Workshop on Software Analytics (SWAN 2018).*
Acceptance rate: 64% (7/11).
https://empirical-software.engineering/publications#swan18-ci

[W2] **Attribution Required: Stack Overflow Code Snippets in GitHub Projects.**
Sebastian Baltes, Richard Kiefer, and Stephan Diehl.
*Proceedings of the 39th International Conference on Software Engineering Companion (ICSE 2017).*
https://empirical-software.engineering/publications#icse17-snippets

[W1] **CodeBasket: Making Developers’ Mental Model Visible and Explorable.**
Benjamin Biegel, Sebastian Baltes, Ivan Scarpellini, and Stephan Diehl.
*Proceedings of the 2nd International Workshop on Context for Software Development (CSD 2015).*
https://empirical-software.engineering/publications#csd15-codebasket

Preprints (under review)

[P2] **Challenges for Inclusion in Software Engineering: The Case of the Emerging Papua New Guinean Society.**
Raula Gaikovina Kula, Christoph Treude, Hideaki Hata, Sebastian Baltes, Igor Steinmacher, Marco Aurelio Gerosa, and Winifred Kula Amini.
https://empirical-software.engineering/publications#bridges

Sebastian Baltes and Paul Ralph.
https://empirical-software.engineering/publications#sampling