Teaching Statement

From tutoring programming and IT security as an undergraduate to professorships at the University of Bayreuth and Heidelberg University, I have accumulated over ten years of teaching experience spanning small graduate seminars, large undergraduate lectures of up to 350 students, and face-to-face and fully remote settings in Germany and Australia. At Bayreuth I designed both an undergraduate software engineering lecture and a new graduate Advanced Software Engineering lecture from scratch; since October 2025, I have been a Professor of Software Engineering at Heidelberg University.

Approach

My central goal is to make students understand *why* the knowledge they acquire matters—in research, in industry, or both. Our empirical research on software development expertise shows that communicating clear goals and visions keeps developers motivated, and I transfer this insight directly into the classroom. As a certified Scrum Master with industry experience as a senior software engineer, I adapt real-world methods and case studies to the courses I teach. Several students have told me that my practically grounded approach turned software engineering—a subject they had previously found dry or abstract—into a topic of genuine and lasting interest.

I apply different models depending on class size, topic, and student background: traditional lectures with assignments, research seminars tied to live projects, live-coding sessions, and the flipped classroom. In a graduate software engineering seminar in Trier, the flipped classroom format led students to co-author a research paper that won a best paper award in 2018; the same approach proved invaluable during the pandemic, when I pre-recorded lectures and used class time for interactive Q&A sessions. I have continued recording and sharing my lectures ever since. I connect courses to current research and recent publications wherever possible, and I use paper critiques as a regular assignment to stimulate critical thinking—students learn to question the proclaimed contributions of a paper and the context in which they hold.

Building an inclusive classroom requires more than stated ground rules. I actively diversify course materials, accommodate different learning styles and disabilities, and adapt my teaching mode when I observe it is not working.

Experience

A complete list of courses taught and theses supervised follows below.

Lectures and Seminars

From 2013 to 2019 at Trier University, I taught seminars, research internships, and tutorials on software engineering, information visualization, and programming, and was responsible for the graduate empirical software engineering lectures from 2015 onward; in winter 2016/17, I revised and delivered the undergraduate software engineering lecture (40–80 students). At the University of Adelaide (2019–2020), I taught courses from introductory to postgraduate software engineering and a graduate research methods course, reaching up to 350 students, and revised the project-based software engineering course to replace a waterfall model with agile processes. At the University of Bayreuth (2023–2025), I designed both the undergraduate software engineering lecture ($\sim$100 students) and the graduate Advanced Software Engineering lecture ($\sim$30 students) from scratch, and additionally taught seminars, a game-development practical course, and a company-partnered project course. Since October 2025, at Heidelberg University, I have revised the undergraduate software engineering lecture ($\sim$120 students), prepared a master's seminar on "Prompts as Software Artifacts", and co-supervised a practical course on object-centric process mining.

Student Supervision

In Trier, I supervised nine bachelor, master, and diploma students; five theses led to peer-reviewed publications, one to a company tool prototype, and two students went on to pursue PhDs. During my time at Adelaide, I co-supervised master students at TU Eindhoven—yielding a publication in IEEE Software (2020) and an ACM SIGSOFT Distinguished Paper Award at ICSE (2023)—as well as students at the University of Victoria, the University of Innsbruck, HPI Potsdam, and the University of Adelaide itself. Since joining Bayreuth and Heidelberg, I have supervised multiple bachelor and master students with several theses feeding directly into active research projects, and continue to co-supervise theses with TU Eindhoven. Three PhD students joined my group at Bayreuth—in late 2024 and early and mid 2025—and I am now building and extending my group at Heidelberg University.

List of Courses Taught

Winter Semester 2025/26 (Heidelberg University)

- Lecture *Software Engineering* (Bachelor): Revised the lecture to align with the Heidelberg University curriculum.
- Seminar *Software Engineering* (Master): Seminar on "Prompts as Software Artifacts".
- Practical course *Software Engineering* (Bachelor/Master): A student team worked on "Object-Centric Process Mining on GitHub".

Summer Semester 2025 (University of Bayreuth)

- Lecture *Software Engineering* (Bachelor): Updated the lecture based on feedback from the previous semester.
- Practical course *Software Engineering for Game Development* (Bachelor): Practical course in which the students developed games in teams.
- Seminar *Software Engineering* (Bachelor/Master): Seminar on current topics in software engineering.

Winter Semester 2024/25 (University of Bayreuth)

- Lecture *Advanced Software Engineering* (Bachelor/Master): Designed the whole lecture, including assignments, from scratch.
- Project *Software Engineering* (Bachelor/Master): Guided and mentored a project group who worked with the local organization PERSONET to develop an internal applicant platform.
- Seminar *Software Engineering* (Bachelor/Master): Seminar on current topics in software engineering.

Summer Semester 2024 (University of Bayreuth)

- Lecture *Software Engineering* (Bachelor): Designed the whole lecture, including assignments, from scratch.
- Practical course *Software Engineering for Game Development* (Bachelor): Practical course in which the students developed games in teams.

Winter Semester 2023/24 (University of Bayreuth)

- Seminar *Software Engineering* (Bachelor): Seminar on current topics in software engineering.

Summer Semester 2021

- Guest lecture in the *Software Engineering* course (Bachelor), University of Applied Sciences Darmstadt: Lecture on empirical software engineering.

Semester 2020-2 (University of Adelaide)

- Project *Software Engineering* (Undergraduate/Postgraduate): Coordinated a major revision of the course, adopting agile process models for the student projects. Further delivered lectures on software process models, Scrum, and software development tools.

Semester 2020-1 (University of Adelaide)

- Lecture *Engineering Software as Services I* (Undergraduate): Lectures on the architecture of SaaS applications, an introduction to Ruby, Ruby on Rails, and BDD.
- Lecture *Software Process Improvement* (Undergraduate/Postgraduate): Lectures on software development process models, software metrics, and estimation techniques.
- Lecture *Research Methods in Software Engineering and Computer Science* (Postgraduate): Lectures on research design, research methods (qualitative, quantitative, mixed-methods), descriptive statistics, clustering, measurement, reporting empirical results, sampling, controlled experiments, ethnography and interview studies, grounded theory, case studies, and surveys.

Semester 2019-2 (University of Adelaide)

- Lecture *Introduction to Software Engineering* (Undergraduate): Lectures on software evolution, security engineering, software testing, project management and planning, quality management, and configuration management.
- Lecture *Engineering Software as Services II* (Undergraduate): Lectures on JavaScript and design patterns.
- Project *Software Engineering* (Undergraduate/Postgraduate): Lectures on project management, software process models, software development tools, risk management, software quality management, and configuration management. Further coordinated two student project groups.

Summer Semester 2019 (University of Trier)

- Tutorial for lecture *Information Visualization* (Master): Responsible for practical assignments and tutorials, covering general properties of visualizations, infographics, UML, multivariate and timeseries data, control flow graphs, visualizations of software architectures, implementation of an algorithm visualization and a tree map in Java, paper critiques.

Winter Semester 2018/19 (University of Trier)

- Lecture *Advanced Software Engineering* (Master): Lectures on continuous integration, static analysis tools, and empirical software engineering as well as corresponding assignments.
- Research seminar *Software Engineering* (Bachelor/Master): Topic selection and grading.
- Guest lecture in the *Research Methods in Software Engineering* course (Master), University of Stuttgart: Two lectures.

Summer Semester 2018 (University of Trier)

- Tutorial for lecture *Programming II* (Bachelor): Responsible for programming assignments in Java, covering topics such as GUI programming, concurrency, data structures, I/O, serialization, reflection, Java lambda expressions and the Stream API.
- Tutorial for lecture *Information Visualization* (Master): Responsible for practical assignments and tutorials, covering general properties of visualizations, infographics, UML, multivariate and timeseries data, control flow graphs, visualizations of software architectures, implementation of an algorithm visualization and a tree map in Java, paper critiques.

Winter Semester 2017/18 (University of Trier)

- Lecture *Advanced Software Engineering* (Master): Lectures on continuous integration, static analysis tools, and empirical software engineering as well as corresponding assignments.
- Research internship on *Software Engineering* (Master): Supervised students working on independent research projects.
- Research seminar *Software Engineering* (Master): Topic selection and grading.

Summer Semester 2017 (University of Trier)

- Tutorial for lecture *Programming II* (Bachelor): Responsible for programming assignments in Java, covering topics such as GUI programming, concurrency, data structures, I/O, serialization, reflection, Java lambda expressions and the Stream API.
- Tutorial for lecture *Information Visualization* (Master): Responsible for practical assignments and tutorials, covering general properties of visualizations, infographics, UML, multivariate and timeseries data, control flow graphs, visualizations of software architectures, implementation of an algorithm visualization and a tree map in Java, paper critiques.
- Lecture *Study Project* (Bachelor): Lecture and tutorial.

Winter Semester 2016/17 (University of Trier)

- Lecture *Software Engineering* (Bachelor): Redesigned lecture and tutorial, covering requirements engineering, GUIs, UI/UX, modelling, object-oriented design, and software architecture.
- Lecture *Independent Studies in Software Engineering* (Master): Supervised students working in a flipped classroom setting on software engineering topics.
- Research internship on *Software Engineering* (Master): Supervised students working on independent research projects.

Summer Semester 2016 (University of Trier)

- Tutorial for lecture *Programming II* (Bachelor): Responsible for programming assignments in Java, covering topics such as GUI programming, concurrency, data structures, I/O, serialization, reflection, Java lambda expressions and the Stream API.
- Tutorial for lecture *Information Visualization* (Master): Responsible for practical assignments and tutorials, covering general properties of visualizations, infographics, UML, multivariate and timeseries data, control flow graphs, visualizations of software architectures, implementation of an algorithm visualization and a tree map in Java, paper critiques.
- Research seminar *Software Engineering* (Bachelor/Master): Topic selection and grading.
- Lecture *Study Project* (Bachelor): Lecture and tutorial.

Winter Semester 2015/16 (University of Trier)

- Tutorial for lecture *Software Engineering* (Bachelor): Responsible for assignments covering paper critiques, GUIs, UI/UX, requirements engineering, static and dynamic modeling with UML, design patterns, test coverage, and software metrics.
- Lecture *Advanced Software Engineering* (Master): Lecture on empirical software engineering as well as corresponding assignments.
- Research internship on *Software Engineering* (Master): Supervised students working on independent research projects.

Summer Semester 2015 (University of Trier)

- Research seminar *Software Engineering* (Bachelor/Master): Topic selection and grading.
- Tutorial for lecture *Information Visualization* (Master): Responsible for practical assignments and tutorials, covering general properties of visualizations, infographics, UML, multivariate and timeseries data, control flow graphs, visualizations of software architectures, implementation of an algorithm visualization and a tree map in Java, paper critiques.
- Lecture *Study Project* (Bachelor): Lecture and tutorial.

Winter Semester 2014/15 (University of Trier)

- Tutorial for lecture *Software Engineering* (Bachelor): Responsible for assignments covering paper critiques, GUIs, UI/UX, requirements engineering, static and dynamic modeling with UML, design patterns, test coverage, and software metrics.
- Tutorial for lecture *Advanced Software Engineering* (Master): Tutorials covering different aspects of empirical software engineering.
- Research internship on *Software Engineering* (Master): Supervised students working on independent research projects.

Summer Semester 2014 (University of Trier)

- Research seminar *HCI/UX* (Bachelor/Master): Topic selection and grading.
- Tutorial for lecture *Information Visualization* (Master): Responsible for practical assignments and tutorials, covering general properties of visualizations, infographics, UML, multivariate and timeseries data, control flow graphs, visualizations of software architectures, implementation of an algorithm visualization and a tree map in Java, paper critiques.

Winter Semester 2013/14 (University of Trier)

- Tutorial for lecture *Software Engineering* (Bachelor): Responsible for assignments covering paper critiques, GUIs, UI/UX, requirements engineering, static and dynamic modeling with UML, design patterns, test coverage, and software metrics.
- Tutorial for lecture *Advanced Software Engineering* (Master): Tutorials covering different aspects of empirical software engineering.
- Research seminar *Software Engineering* (Master): Topic selection and grading.

Summer Semester 2013 (University of Trier)

- Research seminar *Software Engineering* (Master): Topic selection and grading.

List of Supervised Theses

- Levi Böhme — *Trunk-Based Development in Open Source Software Projects: A Data Driven Approach* (Bachelor's Thesis, University of Bayreuth, Germany, 2025)
- Valerian Allingham — *Test Flakiness in Open-Source Database Systems: A Study of PostgreSQL* (Bachelor's Thesis, University of Bayreuth, Germany, 2025)
- Joshua Friedrich — *Einsatz von Spot-Instanzen in CI/CD-Pipelines: Eine multivokale Literaturanalyse und praxisnahe Umsetzung mit AWS und GitHub* (Bachelor's Thesis, University of Bayreuth, Germany, 2025)
- Andreea Maican — *Exploring Returnship Programs: Combining Academic Insights with Real-World Experiences in the IT Sector* (Master's Thesis, Eindhoven University of Technology, Netherlands, 2025)
- Andrei Arkhipov — *Process Mining in Software Engineering* (Master's Thesis, University of Bayreuth, Germany, 2025)
- Philipp Göhl — *Integration und Evaluation von LLM-basierten Agentensystemen in den frühen Phasen des Software Development Life Cycles* (Bachelor's Thesis, University of Bayreuth, Germany, 2025)
- Kilian van Rooijen — *Perceptions of Age and Career Longevity Among Younger Software Developers in Reddit Communities* (Bachelor's Thesis, Eindhoven University of Technology, Netherlands, 2025)
- Alexander Steigerwald — *Tiefere Integration KI-basierter Assistenten in Entwicklungsumgebungen zur Unterstützung des Debuggings* (Bachelor's Thesis, University of Bayreuth, Germany, 2024)
- Adriano Torres — *Applying Information Theory to Software Evolution: What can we Learn from Surprising Changes?* (Master's Thesis, University of Adelaide, Australia, 2024)
- Brian Pfitzmann — *Enhancing Enterprise Software Documentation with Community Content* (Master's Thesis, Hasso Plattner Institute, University of Potsdam, Germany, 2023)
- Maria Christina Kirchner — *Stack Overflow Code Snippet Selection: An Experiment on the Effects of Source Code Comments and Answer Scoring* (Master's Thesis, University of Innsbruck, Austria, 2022)
- Sterre van Breukelen — *The Survivorship of Older Women in Software Development: An Intersection between Age and Gender* (Master's Thesis, Eindhoven University of Technology, Netherlands, 2022)
- Nimmi Weeraddana — *How Solution Snippets are Presented in Stack Overflow and How those Solution Snippets Need to be Adapted for Reuse* (Master's Thesis, University of Victoria, Canada, 2022)
- Tingsheng Lai — *Using Machine Learning to Classify Programming-related Online Snippets* (Master's Thesis, University of Adelaide, Australia, 2020)
- George Park — *Age(ing) in Software Development* (Master's Thesis, Eindhoven University of Technology, Netherlands, 2019)
- Lorik Dumani — *Reconstructing and Linking the Version History of Stack Overflow Posts* (Master's Thesis, University of Trier, Germany, 2017)
- Richard Kiefer — *Stack Overflow Code Snippets in GitHub Repositories: Referenced and Unreferenced Occurrences* (Master's Thesis, University of Trier, Germany, 2017)
- Mert Demir — *Diskussionsverhalten englisch- und japanischsprachiger Entwickler auf Q&A-Seiten im Vergleich: Eine explorative Analyse am Beispiel von Stack Overflow* (Bachelor's Thesis, University of Trier, Germany, 2017)
- Bob Prevos — *Skizzieren mit Hilfe animierter Zeichnungen* (Master's Thesis, University of Trier, Germany, 2016)
- Fabrice Hollerich — *LivelySketches: Lifecycle Support für Skizzen* (Master's Thesis, University of Trier, Germany, 2016)
- Oliver Moseler — *Profiling mit Skizzen* (Master's Thesis, University of Trier, Germany, 2015)
- Pascal Robert — *Visuelle Worthäufigkeitsanalyse mit THREE.js* (Master's Thesis, University of Trier, Germany, 2015)

- Sascha Rudolph — *Berechnung und Visualisierung ähnlicher Ordnerpaare in einem Verzeichnisbaum* (Diploma Thesis, University of Trier, Germany, 2015)
- Peter Schmitz — *Sketchlink Plugin: Improving software documentation and comprehension by linking source code to relevant sketches and utilizing them for navigation tasks* (Diploma Thesis, University of Trier, Germany, 2014)