Teaching Statement

Computer science is a rapidly evolving and ever-changing field. Many of today's programming languages and software tools did not exist twenty years ago, and, most likely, today's state-of-the-art techniques will be obsolete or superseded twenty years from now. Therefore, teaching computer science is not only about transmitting knowledge to students; it is about interesting them in computer science in general, attracting a specific subject area, equipping them with a firm understanding of the fundamentals, and disseminating recent developments in the field.

Teaching Philosophy

Teaching is one of the essential activities of academia, not only to impart knowledge to students necessary for their future studies and careers but also to interact with students, impart skills, and inspire students to inquire and learn independently. However, in an increasingly connected world with a vast of internet tutorials, explanations, and learning material, the role of classroom courses and traditional teaching methods may decrease to some extend. Nevertheless, a teacher still plays a crucial role in increasing curiosity in learning the various concepts and techniques and showing the joys of exploring the different topics. Sustainable learning success intensely correlates with the level of engagement, and a well-organized lecture can be highly effective in presenting new and complex material.

Therefore, to keep students engaged and attract students to questions and problems in computer science, it is essential to provide intuition and thoroughly motivate the problem or the methodology. It is much easier to understand a complex concept after grasping the general idea from a simplified model or an example. Good visual aids and explanations of the material are the first steps into understanding an extensive theoretical problem and can easily spice up complex and relatively dry subjects.

To facilitate this, I will keep the lectures relevant by updating course material to cover new trends, tools, and techniques in industry and research, while maintaining a solid education in the fundamentals of the methods. This helps students develop a solid understanding of the cornerstones of software engineering while imparting new techniques and discoveries disseminated to the students in terms of these fundamental concepts.

Moreover, computer science is a practical field spanning over theoretical disciplines such as algorithms, computation, and information theory, all of which are motivated by the design and implementation of hardware and software. Therefore, to gain comprehensive and sound knowledge in a specific area, students should have hands-on experience by building small systems or working through problems rather than just getting a theoretical treatment of the subject. Due to my experiences as a student and teaching assistant, I believe that a mix of theoretical content combined with sound hands-on practical experience achieves the best learning results.

Furthermore, classroom teaching is essential for knowledge transfer, but most learning happens outside of the classroom. Therefore, creating exciting and challenging homework assignments where students design and implement software can be a great learning experi-

ence and easily teach theoretical concepts and essential practical software skills. I intend to develop homework assignments that draw on the students' own experiences and are relevant to current trends and techniques in both industry and research. In addition, I will keep my classes interactive by introducing discussion sessions on open-ended topics.

Besides, while teaching, I prefer an interactive style by encouraging active discussions, posing questions myself, and soliciting questions from the students during a lecture. Therefore, besides traditional classroom discussions, I would like to promote online discussions and interactive tools to further student participation. Though these lack the personal one-on-one touch of a classroom, they offer the advantage of anonymity which would enable students, who are either naturally reserved or those from minority groups, to participate freely in the discussions. In addition, intense dialogue with the students and constant feedback helps me gauge the class and pace my lecture accordingly. Given the subjective nature of evaluation, I will pay careful attention to the evaluation criterion. I believe good learning results come along with constant and collective improvement for students and teachers.

Teaching Interests

As research, in general, focuses on specific details, teaching provides us with a broader outlook. It encourages us to constantly rethink our research in the general context of computer science and the real world.

I am interested in teaching various lectures from the programming languages area, including—among others—courses on principles of programming languages, program analysis, compiler construction, and interpretation. These lectures will cover the three major subject areas of programming languages and provide a solid education in the fundamentals of techniques and tools applied when analyzing, compiling, or interpreting programs.

Besides the programming languages specific lectures, I'm also interested in teaching the fundamentals of software engineering and bringing in my knowledge of software engineering, agile software development, and cloud computing, which I gained in the last years.

Apart from lectures, I am interested in offering seminars to link students with academic research and recent programming language developments. Seminars are among the best forums to interest students in computer science research in general and programming languages in particular.

Teaching Experience

During my Ph.D. studies, I taught programming language fundamentals and introductory software engineering topics.

In 2012 and 2013, I co-organized the *Software Engineering* course. *Software Engineering* is an introductory course introducing students to the fundamentals of software engineering, covering fields like project management, requirements engineering, design patterns, testing, and formal verification. Together with my colleague in the programming languages research groups, I have organized the homework assignments. Besides, I took several lecture parts, like

the Design Patterns and Program Verification lectures.

In 2015, I co-organized the *Essentials of Programming Languages* course, an advanced course focussing on the fundamentals of programming languages, in particular, Abstract Syntax, Operational Semantics, and Data Types. The course was split into a theoretical lecture and practical homework requiring the students to implement certain aspects of the course in PLT Redex. I have been responsible for designing the homework exercises and managing the exercise classes.

In 2016, I co-organized *Compiler Construction*, another advanced course introducing students to programming language compilers' fundamental concepts and techniques. The lecture was again split into a theoretical part and a practical exercise. The theoretical part covered Grammars, Types, Type Soundness, Liveness Analysis, or Garbage Collection. The practical training requested the student to implement a language compiler transforming MiniJava to MIPS. Since the course had a heavy practical bias, the teaching tasks were split nearly equally between the lecturer and the teaching assistant.

In addition, from 2012 to 2015, I taught an *Android Smartphone Development Lab* course. The *AAndroid Smartphone Development*, is a newly created that introduces students to the fundamentals of mobile app development. The curse consisted of small theoretical sessions introducing the fundamental concepts of smartphone development, combined with extensive practical exercises. As a teaching assistant and initiator of that course, I have been responsible for building the course material, lecturing the theoretical parts, and guiding the students through the practical exercises. As this was a newly created course, all the syllabus, lecture and training material, and homework had to be prepared from scratch.

Besides, I organized an *AAdvances Android Smartphone Development* Lab for two consecutive years, and an advanced practical lab comprises more complex Android smartphone and software development topics, focusing on real-world applications. In the first year, the students developed a patient monitor simulation app, a training app implemented in combination with the St. Josefskrankenhaus in Freiburg. In the second year, the development project included designing and implementing a lecture app, an interactive smartphone app for managing lectures and exercises. In both iterations, I acted as product owner and supervisor in case of technical problems or doubts and have been responsible for the overall course organization.

In addition, as a teaching assistant, I have been responsible for several seminars and pro- seminars, covering various topics of the programming languages research area, and I supervised several undergrad students in preparing their Bachelor's or Master's thesis.

Within the last four years, my teaching activities focused on mentoring other colleagues in banking practice, agile software development, and Java and JavaScript technology matters. The IBM Mentoring Program pairs IBM senior professionals with university students and junior professionals. It enables them to receive personalized career guidance, support in business and technology matters, or have opportunities for improving knowledge and skills.

Besides the mentoring program, I trained other colleges to make good presentations, tailor presentations and talks to the audience, and create meaningful supporting material. In addition, I taught others in Java and JavaScript technology matters, especially on design patterns, writing good core, or improving the overall code quality.