Teaching Statement

Marcel Böhme

I strongly believe that cyber security, software engineering, and related topics in computer science are practical subjects that directly impact the success of our computer science graduates as a competitive and productive workforce. To prepare students adequately, my learning objectives are threefold: i) to cultivate an "engineering mindset" whereby students learn to appreciate smart but practical techniques, established design patterns, and creative strategies that replace unsafe adhoc approaches, ii) to insure that students understand foundational concepts that transcend the latest frameworks and programming languages in this fast-paced world of cyber security and software engineering, and iii) to create a hands-on, collaborative, and creative problem-solving experience where students learn to work independently in small teams on real-world projects.

> *"I hear and I forget. I see and I remember. I do and I understand."* -- Chinese Proverb

I trust that active learning and hands-on training is the best means to achieve these goals. For a course on software testing,[1] I was given the opportunity to design the hands-on project and laboratory that accompanied the lectures. For the project, I wanted the students to introduce their own bugs during development, yet keep the focus on the testing rather than the development. I designed the project such that the students can safely make mistakes and then learn from them. For instance, when all has been implemented and supposedly well-tested, the students participate in a "Hackathon". I decided to add points for teams that found many bugs in the code of others instead of subtracting points from those who introduced the bugs and did not test well. The effectiveness of this positive reinforcement was evident in the final reports: Most students declared profound surprise that, despite their greatest confidence, their program was still buggy. I am very pleased to know that they are sure to be testing their programs more carefully in future.

> *"Flipping the classroom is more about a mindset: redirecting attention away from the teacher and putting attention on the learner and the learning."* -- A. Sams

I found the flipped classroom to be very effective for advanced courses. Instead of teaching the material by means of frontal instruction, the students read up on their own and summarize the material as preparation for the next lecture. This leaves the lecture time for me to reinforce the key concepts using concrete examples and to inspire more elaborate discussions after a brief summary of my own. In fact, my course on automated debugging (that was organized as a flipped classroom) was voted as *best seminar* and *third best course overall* in Semester 2 of AY14/15 in the Computer Science department of Saarland University.

> *"Give me 6 hours to chop down a tree and I will spend the first 4 hours sharpening the axe."* -- A. Lincoln

I believe it is very important to spend time motivating the key problems rather than discussing a disconnected sequence of solutions. I had the rare opportunity at NUS to be both, a research fellow and a lecturer. I prepared and independently taught the lectures of a course in the second half of the semester.[2] I began each lecture with a strong motivation: "People say, a successful test suite is useless. Why?" and then smoothly lead the students to the key insights: "At least we can use a successful test suite to show the absence of specific kinds of errors". If students know why the topic of a lecture is generally important and how it relates to their own experience, they will easily follow through the relevant details. I expect the teaching performance feedback to be available soon.

I look forward to expanding my teaching experience. I would love to teach classes on software and system security, attacks and defenses, software engineering, and program analysis. I am also very familiar with formal methods, programming languages, and compilers. I look forward to working with other faculty members to make sure that the overall curriculum is up-to-date and prepares students appropriately for their careers.

[1] The first offering of CS4218 Software Testing and Debugging at NUS (Sem 2, AY 13/14).

[2] The third offering of CS4218 Software Testing at NUS (Sem 2, AY 15/16).