Teaching Statement

Kurtis Heimerl (kheimerl@cs.berkeley.edu)

In late 2007, I began looking into both graduate school and industrial opportunities for myself. Having no familial or personal experience with anything but industrial work, I asked one of my advisors for direction and he told me the following, "If what you enjoy is gaining impact by shaping products, then industry is the route for you. If what you enjoy is gaining impact by shaping future generations, then academia is the route for you." I took this advice to heart and chose to go to graduate school. I remain focused on students and teaching.

My teaching experience is both broad and deep. During my undergraduate career I was a teacher's assistant for *Computer Programming II* (introductory CS) twice, once under Stuart Regis and once under Hal Perkins, where I won an honorary mention for the Bob Bandes Memorial Teaching Award. I also assisted with *Introduction to Operating Systems* three times, twice under Ed Lazowska and once under John Zahorjan. In all of these classes, I ran self-directed sections, organized teaching material, developed assignments, and graded homework, projects and exams.

During my graduate career, I was the TA for Eric Brewer's graduate CS294 *Cell Phones as a Computing Platform*. I was also able to organize and run my own version of that class for the undergraduate population, CS 194 *Cell Phones as a Computing Platform* in the spring of 2012. This class was organized as a long-term project class, with students forming teams focused on group projects. My own role included supervision of student projects, creating lecture content, and evaluating student performance. Though not officially, I was effectively the lead instructor for the course and planned and executed the entire syllabus. Numerous fantastic projects resulted from the class, including GameHere, a large-scale group gaming experience using cellular phones and HelpOut, a crowdsourcing 911 replacement.

Aside from this direct lecturing and organizing experience, I also have many examples of personal tutoring and instruction. I have been a mentor for three Summer Undergraduate Program in Engineering Research at Berkeley (SUPERB) undergraduates. Kehontas Rowe (Mills College) and Christine Dierk (Elon University, now UC Berkeley) both worked on crowdsourcing projects in the HCI space. Kehontas went on to win an Anita Borg scholarship. Mark Watts (University of Texas at Austin) joined our team to work on security in GSM mobile networks. I have also worked directly with numerous undergraduates at Berkeley on research, including Amil Khanzada, Anuvind Menon, Omar Ramadan, and Sean Roberts on GSM systems for security and access, and Jack Lin, Shubham Sinha, and Bryan Trinh on crowdsourcing systems.

This diverse background of classes, projects, and levels of student involvement informs my methods and philosophies about teaching. Foremost, students at different levels of computer science require different styles of instruction. Introductory classes are some of the most diverse in CS; a key property that, if supported by instructors, could greatly benefit the field. I myself am not a traditional CS student; I never coded until introductory CS at my undergraduate institution. Diverse groups of students are best served by as much one-on-one interaction as possible with faculty and/or assistants. This lets each individual student experience the field of computer science in their own way by asking questions and receiving answers relevant to their own lived experience. More senior classes, such as Operating Systems and HCI, showcase the breadth and speed at which CS moves; these students must be instructed how to find answers on their own using the wealth of tools available on the Internet. This skill is critical as students leave the classroom and begin to engage with new technologies in industry and academia. Mentorship roles are the most extreme version of this, as students must be able to make progress without direct intervention from their mentor. For those, I had weekly meetings with students where blocks are discussed and potential solutions devised.

More broadly, there are basic strategies that span all instructional situations. Hands-on learning is critical for computer science. Most students have the basic tools required to program (a laptop), and giving them a real-world dataset or tool empowers them to engage with and solve real-world problems. At UW I was one of the first participants in a Google-led course utilizing Hadoop, teaching me how to work with real-world large-scale systems. The experience was incredibly valuable. Similarly, face time, be it one-on-one or group office hours, is critical for keeping students interested and motivated when waylaid by technical difficulties.

As a new faculty member, I would be qualified to teach a diverse range of classes. I have extensive research and practical knowledge in HCI, ICTD, and ubiquitous computing, enabling me to similarly teach *Human-Computer Interaction,* and *Ubiquitous Computing*. Much of my research output focused on computer networks (specifically mobile networks), allowing me to instruct *Computer Networks*. I have prior experience teaching *Introductory Computer Science* as well as *Operating Systems*, as noted above. I worked in industry and academia for years, learning many best practices for software development, enabling me to instruct *Software Engineering*. Lastly, I am also very interested in teaching seminars on the broad, cross-disciplinary fields in which I am active. These include *Mobile Computing* -- covering the entirety of mobile from user interfaces and app design to the internals of cellular networks and their protocols; and *Information and Communication Technologies for Development* – covering the design, implementation and evaluation of computing applications in the developing world.

Finally, I wish to reemphasize how important teaching is to me as both a student and instructor. My entire academic career is the result of a single educator caring about his students. I began as an academically strong, but disconnected, CS student in an operating systems class. This instructor noticed me as I asked a lot of questions but seemed to put the answers to good use. He suggested I participate in research, something I had never considered before. This led to work with Dan Grossman, Anna Karlin, and Steve Gribble, then graduate school, advising by Eric Brewer and Tapan Parikh, and eventually this statement. I believe myself to be an example of the wisdom that started this essay; teachers and instructors having a broader impact on CS and the world through their students. I hope to do the same.