I believe that a great systems/networking course must contain a challenging, project-based, experimental component. For example, in the Fall of 2008 I was the TA for an undergraduate networks course at the University of Washington. The course projects involved writing code for real networking hardware, the WRT54G Linksys router, one of which was given to each student. The class was challenging for me and the instructors, because we have never used these routers ourselves! At the same time, it was exciting because students got to experiment with live hardware and were working on something significant. I enjoy teaching courses of this kind.

Teaching philosophy

No single teaching method can successfully convey material to all students. I actively strive to understand what motivates my students to learn and what teaching style is most appropriate for them. Therefore, my teaching philosophy revolves around feedback and effective communication. Methodologically, I believe that project-based learning is an especially suitable approach to teaching computer science. I also enjoy supplementing lecture courses with periods of active learning.

The power of feedback. College-age students are adults and must be taught in ways that build on their personal motives. In my teaching I plan to use feedback mechanisms to engage students to influence the course. For example, I will integrate feedback questions directly into assignments. This way, students can, for example, comment on the difficulty of an assignment right after completing it. I also plan to use standard feedback mechanisms, such as surveys. For example, a survey after the first class will tell me why students are taking the class, their prior background with the topic, and ways in which I can accommodate their learning. This will help me gauge the level of the class, and specialize my course materials to fit student interests and needs.

Student feedback will improve my communication with the class and let me quickly discover when (and why) a student is confused by the material. Naturally, student' feedback will also help me improve my teaching and help me correct my own misconceptions of what students are expecting from the course.

As students provide me with routine feedback about the class, so will I provide them with regular feedback on my evaluation of their progress. Project-based courses are especially well suited to continuous evaluation, as there is no straight line to the final artifact and what counts is the accumulated experience in working through numerous issues that come up over the course of the project.

Seattle
Open Peer-to-Peer Computing

Project-based learning. A difficulty for many systems faculty is a lack of software systems that are sufficiently interesting, yet simple enough and well documented, to be used in a project-based course. To help with this, as part of my research on distributed systems, I have built Seattle[1], which is a testbed deployed world-wide and used by systems and networks courses around the world. Seattle exposes a narrow API that students can master in a few assignments. I helped develop a set of pre-packaged assignments and projects[2] for the platform, and I plan to use Seattle in teaching a hands-on project-based course on distributed systems. In this course students will implement, deploy, and evaluate widely-used distributed algorithms and network protocols on a global scale.

More generally, I'm exited about participating in the wider community of computer science educators to share with and benefit from the materials and experiences of other teachers. For example, I plan

[1] https://seattle.cs.washington.edu
[2] https://seattle.cs.washington.edu/html/education.html

to continue to develop Seattle and other helpful software that I can use in my classes and share freely with others.

Active learning and group work. Some of my most positive experiences in the classroom involved active learning and group work. Unfortunately, all too often, classes organized as lectures prevent interaction between students by design (e.g., the lights are dimmed, and all attention is focused on the speaker at the front of the class). But, computer science classes have a unique opportunity of teaching students by emulating real-world software industry practices, most of which are team-oriented. In my classes I plan to introduce periods of active learning, during which students break up into groups and work together on a problem (e.g., brainstorming a technique to predict bugs in software). Group work provides a balance to the lecture format and students often learn better from their peers than from the teacher. Moreover, by regularly scheduling group activities, and by shuffling groups, by the end of the course the students will have had a chance to meet with and share a learning experience with all of their classmates.

Teaching focus

My research specialization and prior teaching experience in systems and software engineering make me especially suited to teaching a broad range of graduate- and undergraduate-level courses in these areas. These include networks, network protocol design, operating systems, distributed systems, software engineering, software analysis, software evolution, and software testing. More generally, I can teach a broad set of undergraduate-level courses, both applied (e.g., web programming) and theoretical (e.g., algorithms).

Advising philosophy

I have had extensive experience in mentoring over a dozen undergraduate students working on senior theses and research projects. For me, working closely with students is one of the most exciting aspects of academia.

In advising, as with teaching, no one style fits all students. I try to adapt to my students' needs, but I am generally a hands-on adviser and believe that students, especially PhD students, often require significant involvement from their adviser. PhD students must internalize many complex aspects of academic culture, like reviewing and critically thinking about papers, academic language, paper rebuttals, and so on. And, PhD students primarily learn all of this through mimicry—by modeling themselves after successful researchers who surround them. I believe that the adviser must epitomize good research practices so that students are prompted to follow high standards in their daily work.

The teacher is not only a teacher in the classroom but in all interactions with students. Likewise, I strive to build meaningful advising relationships with my students that go beyond the day-to-day research work. I want my students to feel that they have an active relationship with me and that they can depend on me. Well established management practices, such as regular meetings and responsiveness over email, are critical to this. But, I think that maintaining a positive attitude and developing deep empathy with one's students are just as important to a healthy advising relationship.