Intel California Education Team RFP: Accessible CS Capstone Design – A Hierarchical Approach

William G. Griswold
Computer Science and Engineering
Jacobs School of Engineering
University of California, San Diego
wgg@cs.ucsd.edu
http://www.cs.ucsd.edu/users/wgg

1 Organization: Curricular Issues for CSE

The Department of Computer Science and Engineering at UC San Diego rates as one of the most rapidly improving top research program in computing in the country (source: National Academy of Science, U.S. News & World Report). During this period of rapidly improving quality, our programs and faculty have grown considerably, to where we now serve over 1000 majors with 40 faculty.

One of the stresses that come with growth and success is maintaining a rigorous curriculum and an individualized experience for our students. CSE met the state's rapidly growing demand for degrees in computer science over the last decade by increasing class sizes and allocating more graduate and undergraduate teaching assistants to help professors teach those classes. This approach has been successful, but the students did not get a personalized experience with extensive faculty contact. Nor did it permit the teaching of 'capstone' design courses, which have become a model for inspirational engineering education that incorporates many of the ineffables of engineering such as teamwork.

Over the past three years CSE has successfully experimented with the introduction of capstone design courses with topical foci in networked games and ubiquitous computing. (http://pisa.ucsd.edu/cse190/ and http://www.cse.ucsd.edu/users/wgg/CSE190/ represent the most recent instantiations of these courses.) Their success has spurred department-wide interest in offering such courses. However, in their current instantiation, these courses can be available only to a select few—our very best students. The enrollments must be kept quite small to run them, and they cannot be offered in lieu of larger courses because the demand for course enrollments is so high.

In the following, I propose a curricular innovation that can open a capstone design course to many more students, while at the same time improving our preparation of Ph.D. students as well. It adds a significant software engineering and project management component that addresses diversity issues in teams. This proposal leverages existing campus and departmental infrastructure, as well as the curriculum development in my newly designed Ubiquitous Computing course and my Compiler Construction course [Gri03].

2 Project: Modular, Hierarchical Capstone

CSE's challenge is not unique. The 1998 Boyer Commission on Educating Undergraduates in the Research University issued a report detailing the challenges of large-scale high-quality undergraduate education, and proposed a solution based on existing curricular models [Boy98]. The Boyer Report's recommendation can be summarized as a "hierarchy of learners" approach. The idea is simple: in order to sustain itself, a research university must train the next generation of educator–researchers. Consequently, the education that undergraduates and graduate students receive should prepare them to make the choice to continue their education, and give them the experience and intellectual tools to succeed if they should choose to do so. (Of course, such tools are also valuable outside the university, as intellectual leadership and mentorship are vital elements of corporate culture.) For example, undergraduates should be prepared to go on to graduate school, not just with knowledge about the discipline, but also experience in teaching and research that prepares them for graduate work as well as the choice to pursue it.

The Boyer Commission's recommendation, then, is to give students first-hand experience in performing at the next level. They recommend achieving this through a scalable hierarchical approach, with faculty preparing graduate students to be faculty, and graduate students preparing undergraduates to be graduate students. Moreover, they recommend that more senior undergraduates prepare more junior undergraduates. A concrete instantiation of these recommendations is for a course to employ a professor, a few graduate students, and several more senior undergraduates in teaching. Typically, the professor will teach the main lecture, graduate students will conduct discussion sections, and the undergraduate assistants will help students in the lab. To maximize faculty and graduate student contact with the students, they of course hold office hours, but it is also recommended that they spend time in the lab with the students.

CSE in fact already implements this approach for most

of its core courses, but the fanning factor is not very great—both manpower (graduates and undergraduates) and the funds to employ them are limited. This is adequate for most courses, but not for capstone design courses, in which undergraduates demand and deserve continuous intimate interaction from the professor and graduate students. The way to address this is to treat the graduate student and undergraduate assistants as students in the course, *in the role of learning to teach computer science*. Of course, "the best way to learn a subject is to teach it," so these students are learning in multiple dimensions. A first step in this direction, which I propose to leverage, are two courses recently introduced in CSE, CSE 195 at the undergraduate level and CSE 599 at the graduate level (http://www-cse.ucsd.edu/classes/fa02/cse599/).

A small capstone course. To be specific, consider CSE 190 Applications in Ubiquitous Computing, which I designed last year and taught for the first time in Fall 2002, and will teach again this Fall 2003, proposed in the format below. The class had two elements, a twice-a-week research seminar meeting, plus a research project. The research seminar format is an apt forward look to graduate life, while the research project is an opportunity to conduct research like a graduate students do, albeit on a tighter schedule. The project is ambitious and performed in teams, so it includes the essential elements of a capstone design course. Several of the seminar meetings were replaced by colloquium talks from outside researchers, guest speakers (both faculty and graduate students), and a visit to a research lab on campus. One seminar meeting was dedicated to how to prepare a research project proposal, and another for giving and critiquing the teams' proposals themselves. At a separate event at the end of the quarter, each team publicly presented its project in the form of a research talk that included a demo of the system they had built.

Even at this small scale, the course has the essential elements of the "hierarchy of learners", as the students did not just learn about ubiquitous computing, but the students got an intimate look at—and preparation for—graduate student life. Of the eight students who were graduating, I wrote graduate school recommendation letters for four of them. These letters were substantive (and strong) due to the rigor of the course and the intimate seminar format.

Scaling up capstone design. The question, then, is how such a course's impact could be increased both by including more undergraduates and providing an educational context for graduate students, without decreasing the quality of the course. The following proposal serves as a template for replication throughout our capstone alternatives.

One thing that is apparent is that many students could not succeed in the class without specific training in software engineering and project management. To deal with this issue, a once-a-week lecture would be introduced, given by me, teaching the basics of software engineering, teamwork, and project management. Software engineering and project management are best conducted in a domain-specific manner, so motivating background, methods, and examples would be developed in the Ubiquitous Computing context. Because California (and our student body) is highly diverse, I would develop the teamwork unit around cultural issues. More aggressively, each team could include a senior student who had taken the course before, who could serve as a project leader. This is quite realistic (and consistent with the hierarchy of learners concept), as a project leader should have experience drawn from a similar project.

Another challenge is that holding a research seminar with 60 students in the room is impossible. Not everyone could meaningfully participate, and many students would be too intimidated to speak up in front of so many people. Building on the Boyer Commission's ideas, we propose to break a class of 60 students into four groups of 15, each led by a discussion facilitator—myself or a Ph.D. student. The Ph.D. students, having taken research seminars before and working with only 15 students, would be well equipped to succeed. Ideally, there would be more than 3 Ph.D. students in the course, so that a rotation could be set up to allow them to observe me and the others. Additionally, I would meet weekly with the Ph.D. students to discuss the papers in advance, develop questions for discussion, and manage project issues.

The current course is taught in a generic course number and title, CSE 190 Topics in Computer Science and Engineering. With the formalization of both the content and structure of the course, I would submit it to the University as its own course, Capstone Design in Ubiquitous Computing. The umbrella will be sufficiently broad that a half-dozen faculty in the department could teach the course from their own perspective. For example, Stefan Savage, a faculty member in networking and security, has proposed a CSE 190 capstone in Ubiquitous Computing whose project would consist of each team building a component of a system for aiding the training of runners.

3 Budget and Project Details

Growing an existing capstone course as outlined above will require two additional kinds of resources: equipment sufficient for a large number of teams and curricular development of the 'engineering' lecture track. The topical content of the course itself has already been developed, although the content will continue to evolve with the discipline, and new visitors and trips will need to be arranged. I would use the resources developed in Dean Tullsen's CSE

599 course for preparation of the Ph.D. student. In addition to the teaching resources that CSE would normally provide, the department has promised a match of one 10 hour/week TA in support of the project.

3.1 Curriculum Development

To support the development of the project lecture track and the graduate student teaching resources, we request 1 month of summer support for myself and one graduate student. For the development of this track, I will use my experience in teaching our compilers project course CSE 131B with a significant project and team software engineering component [Gri03]. I will also draw on UCSD's Center for Teaching Development (CTD) for resources in teamwork and diversity (http://www-ctd.ucsd.edu/.

3.2 Equipment, Infrastructure, & Software

When taught last Fall, the course depended upon the loan of 802.11b wireless HP Jornada PDA's from 6th College for mobile computing development and to give students a first-hand experience with ubiquitous computing, for example user our ActiveCampus Explorer application. Other lab resources—a Windows lab—were provided by Academic Computing Services. My ActiveCampus project provided its ActiveCampus software and hardware resources to support server-side development [GBBT03, ACP]. The UC San Diego campus has over 600 802.11b access points deployed, as well as a Qualcomm-donated 1xEV-DO site with a small number of network clients available for loan. Qualcomm has also deployed cell sites in CSE's and ECE's buildings along with 50 programmable phones, many still available for loan. This was a superb environment for my students to develop their projects. Only one group, which wanted to help commuters find open parking spots, was constrained by available hardware in its development.

However, the loan of PDA's was a one-time arrangement, and could not be scaled to 60 students in any event. Also, motes and sensor nets reached the tipping point this year, so hardware resources to support the development of sensor applications would be an astounding resource for student projects. Even this year such resources would have put the commuter parking group over the top. To this end, I am working with Bill Hodgkiss of the Scripps Institution of Oceanography at UCSD on the development of sensor net resources for course projects. He is now testing motes from Mica and DUST in ECE 191, with the expectation of a software-oriented project to be run in my graduate software engineering course in the spring. 802.11b-enabled PDA's are suitable for many aspects of sensor net applications, but specialized motes also play a role. David Tennenhouse's recent visit to CSE (http://www-cse.ucsd.edu/Events/02_03/Tennenhouse.html) highlighted Intel's research in this area, as well as software efforts in TinyOS, TinyDB, and nesC. Given the emergent quality of this area, we hesitate to name specific hardware products to support student projects in this area. Instead, we propose a cash reserve for the purchase of motes and other components that students require for their projects.

To this end, we propose the acquisition of 75 802.11b PDA's (60 for the students, plus PDA's for specific field deployments and breakage), 6 basic 802.11b laptops (for field development), and a cash reserve for motes and emergent project needs.

PI Griswold (1 month summer, 100%)	$10,407
75 Dell 300MHz Axim 5 PDA's w/802.11b+cradle	$28,990
6 Dell Inspiron 2650 laptops, 256MB w/802.11b	$8,009
Cash reserve for motes and emergent project needs	$2,595
Total	**$50,000**

References

[ACP] The UC San Diego ActiveCampus Project. http://activecampus.ucsd.edu.

[Boy98] Boyer Commission on Educating Undergraduates in the Research University. Reinventing undergraduate education: A blueprint for America's research universities, 1998. http://notes.cc.sunysb.edu/Pres/boyer.nsf/.

[GBBT03] W. G. Griswold, R. Boyer, S. W. Brown, and T. M. Truong. A component architecture for an extensible, highly integrated context-aware computing infrastructure. In *2003 International Conference on Software Engineering (ICSE 2003)*, 2003.

[Gri03] W. G. Griswold. Teaching software engineering in a compiler project course. *ACM Journal on Educational Resources in Computing (JERIC)*, in press, 2003.