Program Self-Study Report
for the
Bachelor of Science
in
Computer Engineering

University of Washington
4 June 2001
## Table of Contents

**Program Self-Study Report**

A. Background Information ........................................................................................................ 1
   1. Degree Titles ..................................................................................................................... 1
   2. Program Modes ............................................................................................................... 1
   3. Actions to Correct Previous Deficiencies ..................................................................... 1
   4. Department Culture and Administrative Structure .................................................... 2
   5. Department Constituencies and Feedback Loops ......................................................... 4
   6. Examples of the Feedback Loops Working ................................................................... 5

B. Accreditation Summary ....................................................................................................... 7
   1. Students .......................................................................................................................... 9
   2. Program Educational Objectives ............................................................................... 12
   3. Program Outcomes and Assessment .......................................................................... 16
   4. Professional Component ............................................................................................. 31
   5. Faculty .......................................................................................................................... 32
   6. Facilities ....................................................................................................................... 33
   7. Institutional Support and Financial Resources ......................................................... 35
   8. Program Criteria .......................................................................................................... 36
   9. Cooperative Education Criteria .................................................................................. 37
  10. General Advanced-Level Program .............................................................................. 38

**Appendix I - Additional Program Information**

A. Tabular Data for Program ................................................................................................... 39
   Table A.1. Basic Curriculum ............................................................................................ 40
   Table A.2. Course and Section Size Summary ............................................................... 42
   Table A.3. Faculty Workload Summary .......................................................................... 45
   Table A.4. Faculty Analysis ............................................................................................. 48
   Table A.5. Support Expenditures .................................................................................... 52

B. Course Syllabi .................................................................................................................... 53

C. Faculty Resumes ............................................................................................................... 97

D. Student exit survey comments for the past two years .................................................. 196
CSE’s educational efforts are shaped by a four-point philosophy:

- We believe that universities are, first and foremost, educational institutions, and that faculty members are, first and foremost, educators. Students are the "product" of the university and its faculty, in both senses of the word: they are the "output," but more importantly they are the "multiplier" that provides leverage for every faculty member. There are other ways to obtain leverage – one can, for example, conduct research without student involvement. But if you seek your primary leverage in some way other than by educating students, then you will be most effective in some other environment.

- We believe that a research university is a unique institution that can provide a unique undergraduate education – an education in which bright and committed students are brought to the very forefront of knowledge, closely mentored by faculty who are (with their students) working to redefine that forefront. More importantly, we believe that a research university must provide this kind of education, because no other institution can, and because any other kind of education can be provided better and/or cheaper elsewhere. We believe in differentiation: there is something that only the University of Washington and its peers can do, and we must focus on it. "If we’re not striving to seamlessly integrate research and education, we’re screwing up."

- We believe in taking a holistic view. Today’s K-12 students are tomorrow’s UW students; we must give them the tools to succeed. The citizens of Washington and their elected representatives pay the bills; they are entitled to a clear explanation of what we are trying to accomplish. The high-technology companies in our region, and our colleagues elsewhere at the University of Washington, represent enormous competitive advantages for us (and we for them); we must reach out in many ways. The future of the University of Washington is inextricably linked to the future of our region; creating a climate conducive to a 21st-century technology-based economy serves everyone’s interests. Finally, special responsibilities fall to computer science programs as we approach the millenium, because of the role that computer science is playing in transforming all aspects of our lives; we must rise to these responsibilities.

- Finally, we believe that the University of Washington’s highly capable students, staff, and faculty will respond to encouragement and example, striving for excellence in response to high expectations.

Our educational efforts and approaches follow directly from these principles. We strive to create the best possible educational experience for our students – one that benefits from, and that benefits, our position as a top-ten research program. We invest in our introductory courses because knowledge of computing is fundamental to success in the modern world, and because these courses are the "attraction waters" for our major. We aggressively recruit, advise, tutor, and mentor students, because we want a diverse collection of the University of Washington’s finest students in our program, and we want these students to succeed. We encourage our undergraduates to work alongside faculty and graduate students as TAs, because this benefits both the students taking the courses and the students who TA them, and creates a "learning community" that extends from the youngest student to the oldest faculty member. We similarly encourage our undergraduates to work alongside faculty and graduate students as RAs, because this is one of many ways in which these students benefit from the unique type of education that only a research university can provide. We facilitate co-op and internship employment because, if properly integrated, it teaches the students things that are complementary to those they learn in our program. We constantly introduce new "Capstone Design
Courses" (many of which are interdisciplinary) because our field is advancing at a remarkable pace, and because these courses provide an unparalleled opportunity for students to synthesize what they have learned throughout their studies. We employ a wide range of "carrots" to encourage outstanding teaching, because encouragement and example work best: a departmental TA award, a departmental faculty teaching award, nomination of faculty and students for University and national recognition, quarterly student evaluations and annual peer evaluations for all faculty, quarterly circulation of a histogram of student evaluations for faculty and for TAs, and more. We invest aggressively in educational technology because it allows us to reach a broad audience of students and citizens, and because we believe that ultimately it will change the nature of education, allowing faculty members to spend more of their time doing the things that only they can do.

That, in fact, is our overriding objective: to do the things that only we can do, and to do them as well as they can be done.

Edward D. Lazowska, Chair
Spring 1999
A. Background Information

1. Degree Titles

The Department of Computer Science and Engineering offers a Bachelor of Science in Computer Engineering degree through the College of Engineering. The program has two options, namely, hardware and embedded software, which were instituted beginning in the fall of 1999.

The Department also offers a Bachelor of Science in Computer Science degree through the College of Arts & Sciences.

2. Program Modes

Both of the department’s degree programs require students to be enrolled full-time on-campus. Courses are offered during day-time hours during all four quarters of the academic year (with a minimal summer quarter). Students must meet a satisfactory progress criterion that requires them to complete three courses towards graduation every quarter they are registered (except summer). Exceptions are granted for cooperative work experiences and special medical leaves.

Most student enter our program in their sophomore year or even the start of their junior year. This is the case for historical reasons to guarantee articulation of transfer students from the state’s Community Colleges. However, it has the negative effect of pushing all the requirements for our program into approximately 6-8 quarters rather than the 12 for which students are typically enrolled at the University. In the past two years, we have started admitting incoming freshmen directly to our major. This was a new program initiated by our department so as to attract strong in-state students who may choose to go elsewhere due to the uncertainty of admission to the major. Having students admitted early also means they can distribute their major requirements through their four years on campus and have more even workloads and possibly more time to engage in research. This has now become a University-wide program and we (as well as many other departments) are now admitting up to 20% of our majors directly from high school.

3. Actions to Correct Previous Shortcomings

The previous ABET review (in 1996) found a “very strong program” with only one minor issue “with respect to the requirements of the ABET engineering criteria and applicable program criteria”. The issue was the degree of exposure our students have to ethical, social, economic, and safety considerations. Although our curriculum does include general course categories relevant to these topics, the reviewers found that it was possible for students to graduate without guaranteed exposure to these topics.

Since 1996, we have been working to remedy this situation. We have experimented with several approaches. On an experimental basis, we have introduced course modules on ethics and societal impact of information technology into several of our senior-level courses as well as our large introductory programming courses. We have prototyped (for two quarters) a weekly seminar with guest speakers from around the campus and beyond. Finally, we have identified several courses offered at the University that address these issues.

Our Curriculum Committee’s recommendation is to insert appropriate modules in the majority of our program’s courses. Our faculty feels strongly that they can’t “teach” ethics but can bring up issues as they arise during the progression of a course. Some examples are provided in the table below for many of the program’s required and elective courses.
We have also expanded our list of recommended humanities/social-sciences (VLPA) electives to include many courses that treat these issues directly and are offered by a variety of departments on campus. We plan to continue a one quarter per year seminar with invited speakers from around campus and the local computing industry. Finally, we plan on adding a module to our capstone design courses and/or the senior-level technical communications course (in which students often enroll contemporaneously and we are working to link more tightly). Students will be asked to write a paper or a portion of their final report on the societal impact of their capstone projects (which can be in areas as diverse as chip design, hardware design, embedded system design, and distributed software design).

We feel strongly that it is important to expose students to ethical issues throughout the curriculum and not in separate, add-on courses that only serve to reinforce an often-perceived separation between technology and societal issues.

At the institutional (College) level, the primary criticism in the previous ABET review was that “the institution still must describe how the design experience is developed and integrated throughout each curriculum, show that it is consistent with the objectives of each program, and identify the major, meaningful design experiences in the curriculum of each engineering program”. In the remainder of this self-study it will be evident that these issues are not now (nor were they ever) a concern for this particular program.

4. Department Culture and Administrative Structure

Collaboration and consensus are the strengths of our departmental culture. The departmental is not hierarchically organized. Rather, committees appointed by the chair, in consultation with an elected Executive Committee, present findings and recommendations to the entire faculty after data gathering and development of alternative proposals and solutions. The committee most relevant to this self-study is the Curriculum Committee that includes faculty, advising staff, undergraduates, and graduate students. The External Advisory Committee, consisting of leading academics and industry researchers, meets yearly to advise the department on its research and educational missions.

The department has two Associate Chairs whose duties are commensurate with the department’s strongest needs. This not only allows department resources to be brought to bear where they can do the most good but also tailors the duties of the Associate Chairs to the particular strengths and interests of the faculty members who take on the position. Currently these positions are dedicated primarily to Educational Outreach and Educational Technology and Distance Learning.

<table>
<thead>
<tr>
<th>Course</th>
<th>Appropriate Ethics/Societal Impact Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>software engineering</td>
<td>safety, reliability, and risk</td>
</tr>
<tr>
<td>embedded software</td>
<td>safety; user control; ubiquitous devices/sensors; privacy</td>
</tr>
<tr>
<td>databases</td>
<td>privacy</td>
</tr>
<tr>
<td>networks</td>
<td>domain name controversies, digital divide</td>
</tr>
<tr>
<td>graphics</td>
<td>it used to be that pictures didn't lie</td>
</tr>
<tr>
<td>programming languages</td>
<td>language/security interactions</td>
</tr>
<tr>
<td>algorithms</td>
<td>encryption - controversies &amp; policy questions</td>
</tr>
<tr>
<td>operating systems</td>
<td>security; hacking; anti-trust</td>
</tr>
<tr>
<td>capstone design</td>
<td>professional ethics; IP issues; surveillance and privacy</td>
</tr>
<tr>
<td></td>
<td>(embedded systems); sex and violence in video games and its effects (software design); safety/reliability (hardware design)</td>
</tr>
</tbody>
</table>
Undergraduate and graduate students have frequent interactions with the faculty through departmental social events (both sponsored by the department and student organizations). The Chair meets with undergraduates once a month at a pizza lunch that is attended by approximately 50 students each time. Although the time at this session is limited to approximately one hour, we have focused discussion topics (conversion of introductory programming from C++ to Java was a recent topic) and many students continue to communicate afterwards via e-mail to specific faculty (leading the discussion), the chair (who is almost always present), and/or advising staff. Graduate students also have frequent and similar interactions.

Administrative staff is supervised primarily by the clusters of faculty they each support. The main office, financial, and facilities staff are centrally supervised. The Lab Director supervises technical staff members who are organized into three separate groups: infrastructure development, operations, and educational support.

![Department administrative structure related to the undergraduate program. Solid lines describe the reporting structure. Dashed lines describe consultation.](image)

The Undergraduate Advising Unit, as shown in the chart above, is the responsibility of the Undergraduate Faculty Advisor who is appointed by the chair for two to three year periods. The Lead Undergraduate Advisor works directly with the Undergraduate Faculty Advisor and supervises the other two staff members that make up the three-person unit.
The lead advisor bears the bulk of administrative duties and is responsible for the day-to-day management of the undergraduate advising staff members. All three have extensive interactions with our majors as well as with prospective students (our advising staff is fielding an exponentially increasing number of inquiries regarding appropriate training and education for careers in computing and information technology), recruitment and outreach activities, and address issues related to the recruitment and retention of women and underrepresented minorities.

5. Department Constituencies and Feedback Loops

The department’s constituencies, in order of importance, are:

- current students in our major,
- prospective students (both at the University, at Community Colleges, and in high school)
- employers of our students and/or the graduate schools our students will enter, and
- the citizens of Washington State.

Input from these constituencies reaches the department in various ways. Current students interact with faculty and advising staff through individual meetings and special events (such as our monthly undergraduate lunches and other sessions on specific topics such as research/graduate-school/job-hunt sponsored by the student chapter of the ACM technical society). Comments and suggestions reach the Undergraduate Faculty Advisor and, through that position, the Curriculum Committee, Executive Committee, and/or Chair as appropriate. The remedies or suggestions can reach the entire faculty for discussion if they are far-reaching enough. Finally, the faculty implements the solutions generated through this process. This is the principal feedback loop that affects curriculum organization and content. Students in all our classes are also provided with a collection of anonymous feedback web forms that are specific to a course or relating to the entire program and department. Feedback is solicited through these forms to encourage an unencumbered exchange of ideas and reactions to the material presented in class and the overall curriculum design. This is particularly helpful in making students feel that their voice can be heard, taken seriously, and have a positive effect.

Information about prospective students reaches the faculty primarily through the undergraduate advising unit and their weekly information sessions for prospective majors. Regular meeting and workshops with Community College instructors provide important feedback on our introductory curriculum (which is mirrored at the state’s CCs). In addition, we interact with the University’s admissions office when we admit freshmen directly to our major. These follow a similar path through the Curriculum Committee to the faculty. Through the advising staff’s information sessions and our undergraduate admissions process, the department can make adjustments to prerequisites to the major at UW as well as and interact with K-12 schools and community colleges regarding their preparation of students.

Current and prospective employers are strongly encouraged to join our department’s Industrial Affiliates Program. An annual meeting, attended by technical representatives of our affiliate companies is an important venue where they can meet and recruit our students in a setting where students have contact to technical staff rather than just human resources specialists. Comments from our affiliates regarding our students’ preparation is fed back to the faculty as a whole by the Industrial Affiliates Program Committee and appropriate action can be initiated through the advising unit and/or the Curriculum Committee. Another form of feedback about employment is through alumni surveys. Former students are strongly encouraged to provide us information about how well prepared they were for their careers once they have a longer-term perspective. Feedback concerning the preparation of our students for graduate school is obtained through our External Advisory Committee as well as faculty colleagues at the top graduate schools. This follows a similar path to industry comments.

Finally, the citizens of our state have ample opportunities to influence the department through our
various outreach and recruitment efforts and their interaction at annual open house events, state legislative representatives, and various community service organizations on which our faculty serve.

Important web sites explaining the ABET process and other aspects of the department:

<table>
<thead>
<tr>
<th>Main CSE web page</th>
<th><a href="http://www.cs.washington.edu">www.cs.washington.edu</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>ABET materials</td>
<td><a href="http://www.cs.washington.edu/abet">www.cs.washington.edu/abet</a></td>
</tr>
<tr>
<td>Information on all our educational programs</td>
<td><a href="http://www.cs.washington.edu/education">www.cs.washington.edu/education</a></td>
</tr>
<tr>
<td>Information for our current students</td>
<td><a href="http://www.cs.washington.edu/education/ugrad/current/">www.cs.washington.edu/education/ugrad/current/</a></td>
</tr>
<tr>
<td>Information for prospective students</td>
<td><a href="http://www.cs.washington.edu/education/ugrad/prospective/">www.cs.washington.edu/education/ugrad/prospective/</a></td>
</tr>
<tr>
<td>Materials on our inaugural Brotman award for educational excellence</td>
<td><a href="http://www.cs.washington.edu/education/brotman.html">www.cs.washington.edu/education/brotman.html</a></td>
</tr>
<tr>
<td>Capstone videos for the last several years</td>
<td><a href="http://www.cs.washington.edu/info/videos/">www.cs.washington.edu/info/videos/</a></td>
</tr>
</tbody>
</table>

6. Examples of the Feedback Loops Working

The table below illustrates some examples of the department feedback loops at work. The highlights shown in this table are only a sampling of the results of a process that has been in place for the past 3 years. It is important to note that a large collection of concerns regarding flexibility in the Computer Engineering program, availability of more courses dealing with embedded systems issues, modernization of requirements, decrease in EE requirements, and increase in free electives were all implemented as part of the Computer Engineering expansion we began to propose in early 1996, refined in the two subsequent years while we began to prototype courses and were finally able to implement beginning in the Autumn of 1999.
<table>
<thead>
<tr>
<th>Qtr</th>
<th>Source of Information</th>
<th>Concern</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wi 98</td>
<td>Students - discussed in Executive Committee</td>
<td>The rotating enrollment priority, which for spring quarter leaves leaves Sophomores as the last to register, although Engr Sophomores all need 142 as a pre-req to get into their majors.</td>
<td>Restricted registration for 142/143 to freshman, soph, and juniors during registration period I. 143 priority to students currently in 142.</td>
</tr>
<tr>
<td>Wi 98</td>
<td>Faculty and students in reflective statements and course evaluations</td>
<td>Lack of motivation for required senior technical writing course</td>
<td>Exploring linking capstone design experience with required senior technical writing course (TC 333) so that students can write about the project they are currently working on in their capstone and provide more varied documentation. This requires co-registration and it has proven a challenge to get students to do this in large enough numbers for a complete section of TC 333 (~25 students). Successful trials were run in Wi 99, Au 99, and Sp 01.</td>
</tr>
<tr>
<td>Au 98</td>
<td>Prospective students and their families</td>
<td>Top students have no guarantee of admission to the department once enrolled in UW.</td>
<td>Implementation of Early Decision allowing us to admit up to 10% of the incoming class directly out of high school. This has been expanded to 20% for the fall of 2001 and has been adopted by many departments throughout the campus and especially in the College of Engineering.</td>
</tr>
<tr>
<td>Wi 99</td>
<td>Students at ugrad lunch</td>
<td>Info about undergrad labs &amp; systems is lacking</td>
<td>A list of all labs, rooms and systems available to undergrads, linked from the &quot;Information for Current Undergraduate Students&quot; page.</td>
</tr>
<tr>
<td>Wi 99</td>
<td>Students at ugrad lunch</td>
<td>Student suggestion venue on Web</td>
<td>Anonymous (as well as non-anonymous) feedback page created where students can address comments to the faculty coordinator for the undergraduate program and advising staff. This page also includes a list of other resources for students: faculty members, department chair, staff advisors, faculty program coordinator, ACM chapter, tech support staff, lab director, etc. It has been useful so far in providing feedback on instructors, labs, and course scheduling.</td>
</tr>
<tr>
<td>Qtr</td>
<td>Source of Information</td>
<td>Concern</td>
<td>Response</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------</td>
<td>--------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Wi 99</td>
<td>Students at undergrad lunch</td>
<td>ACM in formal liaison role</td>
<td>The officers of the ACM (our student technical society chapter) met with the undergraduate faculty advisor at a long breakfast meeting. The students expressed a strong interest in taking a more active role as liaison between students and faculty/staff. ACM now coordinates bi-annual research nights for undergraduates to learn about research in the department as well as annual job-hunt preparation events among many others.</td>
</tr>
<tr>
<td>Wi 99</td>
<td>Students at undergrad lunch</td>
<td>Student exposure to heterogeneity</td>
<td>Curriculum Committee is going to ensure students become familiar with both C++ and Java and Unix and Windows as part of switch of introductory courses to Java.</td>
</tr>
<tr>
<td>Wi 99</td>
<td>Students at undergrad lunch</td>
<td>Ensure Unix environment is always available for students</td>
<td>All good X-terminals are being kept in service and are now split over two labs to make access easier. Windows software (Reflection) has been obtained and is now configured to take over the entire screen of Windows machines and lets students more familiar with Unix have a comfortable desktop for their needs.</td>
</tr>
<tr>
<td>Sp 99</td>
<td>Students at undergrad lunch</td>
<td>Escalate student concerns within Lab staff</td>
<td>Lab staff have been told that they need to route academic questions to faculty, and to escalate student concerns promptly that they think might be getting out of hand so that lab management and/or faculty can take action quickly.</td>
</tr>
<tr>
<td>Sp 99</td>
<td>Graduating students exit survey</td>
<td>Provide training in use of Unix platforms</td>
<td>Working with the advising unit, the ACM developed a set of short tutorials now held bi-annually to help students over the Unix learning curve.</td>
</tr>
<tr>
<td>Au 99</td>
<td>Graduate survey</td>
<td>Not easy to keep in touch with other alumni</td>
<td>Creation of an alumni webpage that includes links to alumni e-mail and webpages, etc.</td>
</tr>
<tr>
<td>Au 99</td>
<td>Exit survey</td>
<td>Increase flexibility so that students can take more foreign language credits</td>
<td>With the program expansion, additional free electives were provided to support a foreign language.</td>
</tr>
<tr>
<td>Wi00</td>
<td>Feedback from students/faculty</td>
<td>Appropriateness of EE332 (bipolar transistors) as a required course</td>
<td>The course was dropped from Computer Engineering requirements as part of expansion of program and institution of two curriculum options.</td>
</tr>
<tr>
<td>Qtr</td>
<td>Source of Information</td>
<td>Concern</td>
<td>Response</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Wi00</td>
<td>Faculty reflective statement</td>
<td>CSE461 (networking course) should have a data structures prerequisite</td>
<td>After curriculum committee review, CSE143 is added to course prerequisites so as to ensure some exposure to data structures without adversely impacting students from other departments (esp. EE).</td>
</tr>
<tr>
<td>Sp 00</td>
<td>Students</td>
<td>Too much paper used for course handouts</td>
<td>Several faculty surveyed classes and determined that although handout are overwhelming popular, more material could be fit on a page to decrease paper use.</td>
</tr>
<tr>
<td>Sp 00</td>
<td>Student exit survey</td>
<td>CSE 461 (networking) should be an approved outer core elective</td>
<td>The curriculum committee recommended that this change be made and the faculty approved it. Also, 461 is now a required course for all CompE majors.</td>
</tr>
<tr>
<td>Au 00</td>
<td>Students</td>
<td>Want more active role in outreach to incoming freshman, women, and minorities</td>
<td>Advisors recruited students at undergraduate lunches and asked others students to get involved. Over 20 students have signed up to assist with various recruitment and outreach activities.</td>
</tr>
<tr>
<td>Au 00</td>
<td>Students</td>
<td>Increasing number of inquiries about graduate school</td>
<td>Conducted first graduate school information night 10/00. Attended by over 60 students. Discussed selection process and preparation of applications.</td>
</tr>
<tr>
<td>Au 00</td>
<td>Students</td>
<td>Many courses in Applied Math are equivalent to some in Math required by the program</td>
<td>After checking with the Math department, the curriculum committee voted to allow substitutions from Amath for select required courses. At the same time, alternate Math courses were also included.</td>
</tr>
<tr>
<td>Au 00</td>
<td>Students and faculty</td>
<td>Undergraduates require a larger space (currently less than 250 sq ft) where they can work on research and independent projects</td>
<td>The Department through grants and gifts obtained by several faculty was able (through the University) to rent and remodel an office campus space of 3500 sq ft where a variety of student projects can be housed. This facility came on line during Spring 2001. See point5.cs.washington.edu.</td>
</tr>
<tr>
<td>Wi01</td>
<td>Faculty reflective statement</td>
<td>CSE461 needs a statistics prerequisite</td>
<td>STAT390 (for CSE students) and IE315 (for EE students) are proposed as prerequisites and should have minimal impact on course scheduling for students.</td>
</tr>
<tr>
<td>Sp01</td>
<td>Students and faculty</td>
<td>Clarification needed for how research credit can be classified and if students can be paid</td>
<td>A web page was developed to cover all issues related to research participation and the dep’t executive committee decided to permit pay for projects for which students are also receiving credit.</td>
</tr>
</tbody>
</table>
B. Accreditation Summary

1. Students

The department operates an Undergraduate Advising Office with three full-time professional advisors (Jennifer Seller, lead advisor; Elizabeth Rowson, advisor; and Crystal Eney, assistant advisor). The Undergraduate Faculty Advisor (Gaetano Borriello) supervises the overall advising effort. The office provides both pre-major advising to prospective students and advising and curriculum planning to current majors. A sample of the CSE Handbook is included in an appendix and at www.cs.washington.edu/education/handbook/. It summarizes in detail the program requirements and advising process. There is a separate Graduate Advising Office supervised by another faculty coordinator for our two graduate programs.

Until recently, there were two separate advisors for the computer engineering and the computer science majors with a third support staff position. However, in 1997, following a turnover in the advising staff, it was decided that the best way to serve our entire undergraduate population was to have all three position be advisors available to all CSE students. Now the three advisors are knowledgeable about both degree programs and can advise any CSE student rather than requiring students to meet with a single specific individual for all their advising needs. We also integrated the computer engineering and the computer science handbooks to ensure that all CSE students are receiving consistent and complete information about all the options available to them from the department.

Admissions

The Undergraduate Advising Office coordinates the departmental undergraduate admissions effort. Admission to the department is highly competitive. Students apply to the program after completing a set of program prerequisites and are evaluated based upon, but not limited to, the following criteria: grades in prerequisite and general education courses, a written personal statement, and potential to contribute to the field of computing. An Admissions Committee, comprised of four faculty members (including the undergraduate faculty advisor) and all four advising staff members, evaluate the applications. Three readers (two faculty members and one advisor) read each application prior to the committee meeting. The committee makes all final decisions by consensus. The past three years, the department has also guaranteed admission to highly competitive entering freshmen that have specified an interest in the department on their application to the UW. These students are not coded as majors until they complete the prerequisites and have maintained a 3.5 GPA in all their courses. All departmental admissions forms are on-line and students can complete the process remotely. Once the on-line application is completed, students print the information, attach their transcripts, and sign a statement that is accurate to the best of their knowledge. The completed packet must arrive at the department in time for two annual deadlines (one on July 1 for Autumn admission, one on Feb 1 for Spring admission).

Upon entry to the major, each student is provided a current copy of the CSE Student Handbook. This guide details the course requirements for the degree, including prerequisites and curriculum. It also outlines the program’s satisfactory progress policy, which specifies criteria for rate and quality of academic progress. The advisors also thoroughly present the curriculum and satisfactory progress policy at mandatory new-student orientations. Early Decision students receive a special orientation session tailored to their needs before starting their first fall quarter. All new students are provided with mentors among our more senior students to help answer their questions and show them the ways of the department.
Transfer students

Transfer students first apply to and are admitted to the University through the University admissions process, and then separately apply to the Department through the same admissions process as students at UW. Our advising staff makes annual visits to local community colleges to hold information sessions that address the specific issues faced by these students.

Transfer credit is accepted subject to the evaluation of each course. The UW provides a centralized evaluation process that we use for non-computing courses. For courses in the major, a faculty member who has recently taught the course in question, evaluates a packet of material provided by the student (for each course) that includes textbooks, assignments and the student’s solutions, as well as syllabi provided by the instructor. The two courses that most often fall in this category are our introductory programming courses. The department has a program for the State’s community colleges to have their course pre-approved (details can be found at: www.cs.washington.edu/outreach/CC/cc_transfer_application.html). We work closely with community college instructors at annual summer workshops to ensure their introductory programming courses are well-aligned with ours and we have provided material ranging from lecture notes and assignment to complete sets of studio-quality video-taped lectures. We are currently actively engaged with the community college regarding the switch in programming language we are beginning to implement this Spring (C/C++ to Java).

In reality, the department admits only a handful of direct transfer students each year. Most of these are from other four-year institutions. Virtually all students who start their degrees at community colleges enroll at the UW campus for at least one quarter and thus do not appear on statistical summaries as direct transfers. Approximately 50% of our students have at least 5 credits that they have completed at institutions other than UW. But few have more than a fourth of their credits from other institutions by the time they graduate. Under the state Higher Education Coordinator Board agreement, students may transfer a maximum of 90 credits towards their degree. In all cases, the final 45 credits of the degree program must be earned while in residence at the University of Washington.

Student monitoring

The staff maintains an interactive database in which transcript information and other data relating to the student’s academic progress is maintained. This information is used to record and monitor each student’s academic progress and to verify that all program requirements will be satisfied by the expected graduation date. The database is available to faculty as well and includes complete student academic records (linked to the University’s main student database), a photograph, and comments entered by the advising staff after every meeting with the student. This allows us to keep a trail of interactions with each student that is accessible to all advising staff members and faculty.

At the end of each quarter, the advisors review the student grade reports to assure compliance with the satisfactory progress policy. Every student determined to be out of compliance with the satisfactory progress policy is required to meet with an advisor who recommends corrective action. Students out of compliance with the program are placed on probation. Subject to confirmation by the Undergraduate Faculty Advisor, students on probation for two consecutive quarters are transferred out of the program. This occurs only on very rare occasions and only after several attempts to remedy the situation.

If students are having academic difficulty in CSE classes, the advisors recommend they seek tutoring assistance from our tutoring program that is organized and staffed by current graduate and undergraduate students. Tutoring may include individual tutoring, directly by the student volunteers, or help in forming a study group with other CSE majors.
Students may graduate at the end of any academic quarter. Satisfactory completion of the Computer Engineering curriculum is assured by an auditing process and by encouragement to regularly meet with the advising staff. A special checkpoint is at two to three quarters preceding the proposed quarter of graduation. Each student must file a formal degree application with an advisor (who also checks with the University of Washington’s DARS - Degree Audit Reporting System – to ensure compliance). This includes a plan of courses in progress and to be scheduled through the quarter of graduation. The outcome is that each student is required to have an acceptable terminal course plan over two quarters in advance of actual graduation.

Exceptions and substitutions regarding prerequisites or degree requirements are only rarely granted, and then only with the explicit permission of the Undergraduate Faculty Advisor. If a student has taken classes at another four-year institution and wishes to transfer credit, they must file a petition. In addition to the petition, they submit copies of the syllabus, homework assignments and exams and name of text/s used. Faculty who teach the course for which credit is being requested, review the petition and accompanying documents and make the final determination on whether transfer credit should be granted.

**Student evaluation**

Students are assigned a decimal grade between 0.0 and 4.0 for each course they complete at UW. Instructors, in cooperation with their teaching assistants, are responsible for grades. There are no formal guidelines for the assignment of grades and most of our faculty do not grade on a curve. Our average course grade for undergraduates is 3.2-3.4 depending on the course. This may seem inflationary but is quite consistent with the high GPAs of our incoming students (approx. 3.5). The minimum grade of 2.0 is required for a course to be used toward the program’s graduation requirements. Students not achieving that grade must repeat the course. Our satisfactory progress requirement is that students must successfully complete three courses towards their graduation every quarter (we do not specify a specific number of credits due to the difficulty of collecting a specific number of credits when credits per course vary widely from department to department).

**Student mentoring**

The undergraduate student organization (the local chapter of the Association for Computing Machinery – ACM) is the principal vehicle for student mentoring. This student-run organization, working under the guidance of the Advising Office, holds a semi-annual orientation events that coincides with the fall and spring quarter entry of new majors. This evening event has near universal attendance on the part of new students and the program serves to educate them about how the department is run, what the faculty considers the most important steps and outcomes, and how they can go about obtaining the full variety of educational experiences open to them.

Current undergraduate majors volunteer to mentor new students. They meet at the orientation event and the mentor serves as an informal advisor for the new student’s first few quarters in the department. The objective is to help students assimilate the culture of the department and learn about the resources available to them as well as how to best approach faculty with their concerns.

No formal process is in place for faculty mentoring. However, our students are strongly encouraged, from their first day in the program, to take part in undergraduate research and teaching assistantship opportunities. In both cases, students work closely with faculty (and their graduate students) to learn more about other aspects of the department than their coursework.
2. Program Educational Objectives

Program Mission Statement

To educate our students so that they will reach their full potential in computer science and engineering research and industrial practice through a deep understanding of the fundamentals of the field, their application in solving problems and creating products, and with an affinity for lifelong educational renewal.

The mission statements of the College of Engineering and the University of Washington are entirely consistent with our departmental educational mission. These are not repeated here for the sake of brevity (the mission of the University of Washington is given in the Faculty Handbook, Vol. 4, Ch. 1, and on the web at: www.washington.edu/faculty/fac senate/handbook, the mission of the College of Engineering is available at: www.engr.washington.edu).

From our program mission statement we derive the following four goals for our Computer Engineering program:

- **Core knowledge**: To offer a challenging and broad-based education in the fundamental ideas, techniques, and tools underlying the discipline of computer engineering.
- **Application**: To impart the ability to apply the core knowledge of the discipline in making the many engineering tradeoffs that are central to solving problems and creating products.
- **Educational renewal**: To prepare our students to continually renew their education in a rapidly developing discipline so that they may realize their full potential throughout their career.
- **Social context**: To foster personal development not only in the social aspects of the discipline itself but also in terms of understanding the impact of our field on society as a whole.

We believe these four goals clearly organize the department’s educational values, namely, graduating highly competent students who can serve as leaders of the field for their entire careers and do so understanding the implications of their work both to themselves and society as a whole.

We further refine these four high-level goals into 17 objectives. Table I is organized with these objectives under their respective goals. In addition, it also demonstrates how the objectives taken together address all 11 of the ABET educational outcomes criteria (i.e., “A-to-K”).

Constituencies

A discussion of the department’s educational program constituents was already presented in section A.5. The list can be further refined into primary, major, and minor constituencies. Of course, feedback from any of these constituents becomes especially compelling when it correlates with that of other constituent groups.

Our primary constituents are:

- students currently enrolled in our major,
- students seeking to become majors in our programs,
- and past graduates of our programs.

Major constituents are composed of those organizations serving our student’s careers immediately after they leave UW, namely:

- our external advisory committee,
- member companies of our industrial affiliates program and other prospective employers, and
• graduate school faculty at the nation’s premier graduate programs.

Finally, the largest, and most diffuse, constituents are:

• parents of past, present, and future majors,
• the citizens of the State of Washington,
• professional societies and accreditation organizations, and
• the State Legislature and Executive.

Process

Initially goals and objectives were derived by the Undergraduate Program Coordinator (then also serving as Associate Chair for Educational Programs) and the advising staff with the help of numerous discussions with the College’s ABET Coordinators Committee (on which the Undergraduate Program Coordinator and Lead Undergraduate Advisor serve). These were then presented to the Curriculum Committee (which includes two undergraduate students, two graduate students, four faculty, and two advisors) and were revised extensively. Finally, they were brought to the entire faculty at a regular faculty meeting and at annual faculty retreats. Presentations by the Undergraduate Program Coordinator explained the reasoning behind the goals and objectives. Extensive feedback was received that lead to a second major revision and the result is shown in Table I. These are now posted on the department web pages and are gathering additional comments from students, affiliate companies, and faculty in other departments. We fully expect further refinements to be made on a regular basis.

Some examples of the how feedback from various groups has led to changes in our program and the service we provide our students are described in Section A.6.

Achievement

The educational objectives are realized through careful curriculum design, continual monitoring of students’ progress, assessment of outcomes, and evaluation of the curriculum by our primary and major constituencies.

The curriculum structure and its rationale is described in detail in Section B.3 and B.4 and consists of components relating to core engineering, science, and mathematics knowledge, the fundamentals of the discipline, and, depending on the student’s chosen program option, a set of advanced courses that put the fundamentals to practice. A capstone design course completes the curriculum with an open-ended individualized design experience. Technical writing courses serve to develop and exercise written and oral communication abilities. Courses in the humanities and social sciences expose students to a breadth of issues including contemporary societal concerns and some of the effects of technology.

Core courses in the department’s curriculum are offered quarterly to make it easier for students to complete their schedules. Careful attention is given to course scheduling to minimize conflicts among our own offerings as well as those of related departments such as Electrical Engineering and Technical Communications.

Curriculum quality is assured through faculty self-assessments of their teaching performance after each course, peer evaluations of teaching for faculty of all ranks on an annual basis, and student evaluations of each course. The self and peer evaluations touch upon not only teaching performance in specific course offerings but also consider curriculum development, development of course-ware, future plans for the course content and assignments, and how previously identified issues were handled. In addition, the Undergraduate Advising Office and the Undergraduate Program Advisor are continually in contact with our primary and major constituencies and bring issues that merit attention to the Curriculum Committee.
When deficiencies or other issues are identified the Undergraduate Program Advisor and the Advising Office work with the relevant faculty to produce a workable solution and, if needed, bring it before the Curriculum Committee and possibly the entire faculty. Again, the reader is referred to Section A.6 for specific examples.
Table I. Goals (1-4) and objectives (1.1-4.4) and their mappings to the ABET criteria (A-K).

<table>
<thead>
<tr>
<th>1. Core knowledge: To offer a challenging and broad-based education in the fundamental ideas, techniques, and tools underlying the discipline of computer engineering.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Apply a broad-based knowledge of math, science, and computing fundamentals to identify, formulate, and solve computer engineering problems.</td>
</tr>
<tr>
<td>1.2 Apply the fundamentals of analysis, synthesis, abstraction, simulation, and experimentation to design and implement solutions to challenging problems.</td>
</tr>
<tr>
<td>1.3 Convey technical concepts through an appropriately targeted oral presentation and interaction with an audience.</td>
</tr>
<tr>
<td>1.4 Convey technical concepts through written materials that satisfy accepted standards for writing style.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2. Application: To impart the ability to apply the core knowledge of the discipline in making the many engineering tradeoffs that are central to solving problems and creating products.</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Decompose design problems and choose appropriate components for their solution.</td>
</tr>
<tr>
<td>2.2 Assemble and integrate software and/or hardware components into new products.</td>
</tr>
<tr>
<td>2.3 Apply the techniques and tools of the discipline in the evaluation of design tradeoffs.</td>
</tr>
<tr>
<td>2.4 Function effectively on a multi-disciplinary team.</td>
</tr>
<tr>
<td>2.5 Participate in a range of learning opportunities including large implementation projects, research apprenticeships, teaching assistantships, and independent study.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3. Educational renewal: To prepare our students to continually renew their education in a rapidly developing discipline so that they may realize their full potential throughout their career.</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Recognize the need for, and will have practiced, self-directed learning.</td>
</tr>
<tr>
<td>3.2 Recognize the need for, and be able to engage in, life-long learning.</td>
</tr>
<tr>
<td>3.3 Understand career options that lead toward graduate school and/or industry practice.</td>
</tr>
<tr>
<td>3.4 Understand the variety of disciplines and industries to which computer engineers can contribute.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4. Social context: To foster personal development not only in the social aspects of the discipline itself but also in terms of understanding the impact of our field on society as a whole.</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Foster a collaborative environment where students, staff, and faculty are all engaged in the continual improvement of the department.</td>
</tr>
<tr>
<td>4.2 Appreciate the complementary roles individuals of different background can play on engineering teams.</td>
</tr>
<tr>
<td>4.3 Understand the ethical issues facing computing professionals and awareness of resources available to those faced with ethical decisions.</td>
</tr>
<tr>
<td>4.4 Knowledge of historical and contemporary issues and the impact of computer engineering solutions in a global and societal context.</td>
</tr>
</tbody>
</table>
3. Program Outcomes and Assessment

Program Outcomes

We view program outcomes as measurable effects of our curriculum. The particular choice of outcomes was strongly influenced by our program objectives. As such, there is close correspondence between them.

Our program outcomes are (grouped loosely into four groups corresponding to the program goals):

1.a Mastered mathematics and science fundamentals.
1.b Applied mathematics and science concepts.
1.c Presented an effective oral presentation.
1.d Prepared effective written materials.

2.a Mastered core of the discipline of computer engineering.
2.b Applied core concepts of the discipline to design/evaluation.
2.c Constructs appropriate abstractions to manage complexity.
2.d Able to evaluate design tradeoffs.
2.e Completed a challenging design project.

3.a Facile in using tools of the discipline.
3.b Worked on a multi-person design team.
3.c Participated in diverse learning opportunities.
3.d Assimilated knowledge of tools or concepts as needed.

4.a Interacted with industry practitioners.
4.b Considered benefits of graduate study.
4.c Understands principal concepts of professional ethics.
4.d Evaluated implications of work in the discipline to society as a whole.

The outcomes are derived from our objectives and represent quantifiers that are relevant to each of the objectives. As such, they have almost a one-to-one correspondence with the program objectives laid out in Table I, and we do not provide a table showing this direct correspondence.

Relation to Program Educational Objectives and ABET Criteria

By virtue of corresponding directly to the program objectives, the outcomes also cover the ABET criteria (A-K) in a similar manner.

The criteria specific to a Computer Engineering Program include: diversity of faculty; their active participation in research in the discipline; curriculum coverage for a balanced view of hardware, software, hardware-software tradeoffs, and basic modeling techniques used to represent the computing process; breadth of program offerings; required depth in a topic area with a sequence of courses that build upon each other.

The department’s faculty covers the discipline handily and all its members are active participants in the research enterprise. The department is ranked among the top 10 in the US in computer science and engineering (and almost always in the top 5 or 6) based primarily on the stature of its faculty and its research and educational impact. Our Department, College, and University tenure committees ensure that this is the case.

The curriculum is structured to provide both depth and breadth. Students majoring in Computer Engineering have the opportunity to choose a hardware-focused or software-focused program with particular emphasis in the context of embedded systems. A set of common core courses forms a foundation on which senior requirements are layered. Appropriate capstone design courses are
designated for the two options. Senior-elective requirements ensure breadth outside the area of specialization.

Processes

Outcomes are achieved via both curricular and extra-curricular activities. The curriculum is designed to provide students with a solid foundation and extensive practice in an area or two of specialization. Cooperative education, internships, teaching assistantships, and research involvement on campus are ways in which students can further enrich their education with real-world experiences that will provide with a more informed outlook on their future career. We strongly encourage our students to obtain these experiences and we have a very high participation rate.

In terms of assessment, in addition to course grades, we focus on the development of portfolios based on students’ work in senior-level project-oriented courses and in capstone design courses. Web-based documentation, oral presentations, and video productions serve to populate the portfolio. The material comes from departmental courses as well as the required technical communication courses (the last of which we are working to link with our capstone design courses so that students enroll in both concurrently and can write about their capstone project in the technical writing courses – providing a stronger motivation).

The audience for student portfolios is quite broad and includes faculty, advising staff, and prospective employers. A much larger audience reviews the videos we produce to capture the resulting projects from many of our capstone courses. The audience includes the general public (they are broadcast on the UW cable television channel) and prospective students. Comments from all these groups (some as formal surveys and others as informal observations) are gathered and continually considered by the Undergraduate Faculty Advisor and the Advising Office for possible action.

Students evaluate ALL our courses each quarter. Besides the standard evaluation questions, students are also asked to rate how well the course addresses the ABET criteria (A-K). This permits us to verify our mappings of outcomes to courses and ABET criteria. In addition, it will permit us to track how our courses are evolving relative to the criteria over time.

Faculty members also evaluate the courses they teach by filing a self-assessment each quarter with the Department Chair. This includes not only a post-mortem examination of the faculty member’s effectiveness as a teacher but also comments regarding course pre-requisites, development, infrastructure, assignments, etc.. Annually, two peers evaluate each and every faculty member, regardless of rank, for teaching performance and effectiveness. Self-assessments, student evaluations, and informal observations by other faculty who attended lectures is collected and overall advice developed to help the faculty member improve their performance.

The Undergraduate Program Advisor continually evaluates the overall program together with the Undergraduate Advising Office and the Curriculum Committee. There is no set timetable for evaluation. However, the staff of the Advising Office meets weekly with the Undergraduate Program Advisor to discuss issues that have arisen. The Curriculum Committee meets once or twice per month to discuss curriculum changes as well as issues forwarded by the Undergraduate Advising Office that are likely to require further attention and a possible faculty vote. Examples of this include everything from minor changes to the approved elective list to proposals to create a new option for the major.

Finally, both the department and the Office of Educational Assessment (OEA) conduct surveys of alumni. The department maintains an alumni web site where students keep their current status up to date and can send feedback to former professors. The focus is on particularly memorable
experiences while they were students and how prepared they felt/feel on the job. The OEA surveys are conducted of students both 1 and 5 years past graduation and include several questions that can be mapped to the program outcomes.

Curricular Coverage of Outcomes

Our curriculum requirements are grouped into 9 categories:

- General education
- Visual, literary, and performing arts/Individuals & societies
- Written and oral communication
- Mathematics and statistics
- Science
- CSE core
- CSE hardware or software option outer core
- CSE electives
- Free electives

Table II illustrates how they are used to obtain our program outcomes. Each of these bears some explanation. General education and VLPA/IS requirements expose our students to a variety of topics in the humanities and the natural world. Written and oral communication requirements include two technical writing courses in addition to basic English composition. Mathematics includes one-year of calculus, differential equations, and linear algebra. A statistics course for scientists and engineers is also required. Science requirements are dominated by a year of physics and corresponding laboratories with one quarter for exposure to introductory chemistry. Students can further increase their range using their free electives. Many of our students pursue a foreign language using this mechanism. Many use these credits for additional independent study in the department (6 credits of which can be counted in CSE electives, 9 credits if part of the University or College Honors program) or to take more electives within the department.

The heart of our curricular requirements is the departmental core requirements which are arranged in three tiers. First, and common for all our students in both Computer Engineering and Computer Science, are the common core requirements that ensure a solid foundation in the discipline. This component consists of 8 required courses. Students can choose between two options for outer core requirements: one focusing on hardware systems and the other on embedded software. Each option has three required senior level courses. In addition, a capstone course is required and students have a choice of one of two different capstones for each option. The capstone courses tie together many of the disparate elements of the curriculum and involve the students in a large implementation project where they work in teams and are exposed to complex engineering tradeoffs. The CSE electives serve to further ensure breadth in the discipline and can include credit for research participation.

We also use extra-curricular elements to further accentuate outcomes that are particularly important to engineers practicing in our field or continuing on to graduate school. These are:

- Industry internships
- Research participation
- Teaching assistantships

Although our students are not required to participate in these extra-curricular elements, a large number of them do participate and gain the most from what our type of department has to offer. For example, more than 50% of our students are recorded as having completed industry internships. Of course, this number is likely much higher as not all students inform us of their employment although our advisors to try to elicit this information when students complete their graduation application. Approximately 25% of our students participate in research with faculty and graduate students. They can often earn a salary for this work letting them spend more time on
campus while earning an income. Finally, approximately 10% of our graduating class each year has participated in teaching assistantships or teaching consultants not only for our introductory courses but for many of our capstone and other laboratory intensive courses as well. We find that our own undergraduates, who are familiar with our equipment and development environments, are often more effective than graduate students, who have not worked directly with the same tools.

We strive to increase participation in the these co-curricular activities via information sessions run by the ACM student chapters, orientation comments and explanations by the Undergraduate Faculty Advisor and Chair, through our Undergraduate Advising Unit, and through student mentoring. The participation rate has been steadily rising as students become aware of these options and talk to fellow students who have had the experiences.

Table III provides a summary of our curriculum requirements and the mapping to our outcomes.

Assessment

A summary of our assessment process can be found in Table IV. For each of our outcomes, we provide performance criteria, the implementation plan, evaluation methods, logistics involved in collecting the required information for evaluation, and who provides the feedback to the student. A meta-feedback loop on all of these is that the faculty are the first line for detecting issues that need attention. Their comments are provided to the advising staff and Undergraduate Faculty Advisor directly and indirectly through course grades and self-evaluations of teaching. Student feedback is provided through course evaluations, anonymous web forms, and public forums of discussion such as the undergraduate lunches held monthly by the Chair and attended by other faculty as well. Many issues can be handled directly by the advising staff. More systemic or problematic issues are forwarded to the Undergraduate Faculty Advisor in weekly meeting with the Undergraduate Advising Office staff. Next in line is the department’s curriculum committee and executive committee (for issues outside the curriculum). Recommendations from these two committees and the Chair may be brought to the entire faculty when bureaucratically necessary for a vote and, more importantly, as part of a continual faculty education process.

The department uses several forms of assessment:

- Course grades
- Course student evaluations
- Faculty self-evaluations
- Student surveys upon entry and exit from the department
- Surveys related to industry employment (pre/post and employer) through the College of Engineering’s Coop Office
- Capstone design projects and video productions
- Writing assessment through the Department of Technical Communications
- Presentations (posters, demos) at the annual affiliates meeting
- Alumni surveys

The combination of all of these metrics provides the leadership of the department (and the faculty) with a multi-faceted picture of the relative strength of the curriculum as seen from the point of view of most of our constituents.
Table II. Program outcomes and mapping to curricular and extra-curricular elements.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Core knowledge:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.a Mastered mathematics and science fundamentals.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.b Applied mathematics and science concepts.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.c Presented an effective oral presentation.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.d Prepared effective written materials.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Application:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.a Mastered core of the discipline of computer engineering.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.b Applied core concepts of the discipline to design/evaluation.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.c Constructs appropriate abstractions to manage complexity.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.d Able to evaluate design tradeoffs.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.e Completed a challenging design project.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. Educational renewal:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.a Facile in using tools of the discipline.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.b Worked on a multi-person design team.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.c Participated in diverse learning opportunities.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.d Assimilated knowledge of tools or concepts as needed.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Social context:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.a Interacted with industry practitioners.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.b Considered benefits of graduate study.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.c Understands principal concepts of professional ethics.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.d Evaluated implications of work in the discipline to society as a whole.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table III. Program outcomes and mapping to curriculum requirements.

<table>
<thead>
<tr>
<th>Core Courses</th>
<th>1.a</th>
<th>1.b</th>
<th>1.c</th>
<th>1.d</th>
<th>2.a</th>
<th>2.b</th>
<th>2.c</th>
<th>2.d</th>
<th>2.e</th>
<th>3.a</th>
<th>3.b</th>
<th>3.c</th>
<th>3.d</th>
<th>4.a</th>
<th>4.b</th>
<th>4.c</th>
<th>4.d</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mathematics</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Science</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General Education/VLPA-IS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Communication</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 142 Computer Programming I</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 143 Computer Programming II</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 321 Discrete Structures</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 322 Intro. to Formal Models</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 326 Data Structures</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 341 Programming Languages</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 370 Intro. to Digital Design</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 378 Machine Org. &amp; Assembly Lang.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 451 Operating Systems</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 461 Networking</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EE 215 Fund. of Elect. Engineering</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EE 233 Circuit Theory</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Hardware Option**

| CSE 467 Advanced Digital Design     |     |     | X   |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| CSE 471 Computer Design & Org.      |     |     |     |     |     |     |     | X   |     |     |     |     |     |     |     |     |     |
| EE 331 Devices & Circuits I         |     |     |     |     |     |     |     |     | X   |     |     |     |     |     |     |     |     |

**Capstone Design**

One of the following:

| CSE 468 Very Large Scale Integration| X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |

**Software Option**

| CSE 403 Software Engineering        |     |     | X   |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| CSE 466 Software for Embedded Sys.  |     |     |     |     |     | X   |     |     |     |     |     |     |     |     |     |     |     |
| CSE 401/457/471 Senior Project oriented course |     |     |     |     |     |     | X   |     |     |     |     |     |     |     |     |     |     |

**Capstone Design**

One of the following:

| CSE 476 Embedded Systems            | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |
| CSE 481 Software Design              | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |

**Senior Electives**

Also any courses listed above

| CSE 421 Intro. to Algorithms         |     |     | X   |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| CSE 431 Intro. to Theory of Computation |     |     |     |     |     |     | X   |     |     |     |     |     |     |     |     |     |     |
| CSE 444 Databases                    |     |     |     |     |     |     |     | X   |     |     |     |     |     |     |     |     |     |
| CSE 458 Computer Animation           |     |     |     |     |     |     |     |     | X   |     |     |     |     |     |     |     |     |
| CSE 473 Intro. to Artificial Intelligence |     |     |     |     |     |     |     |     |     |     | X   |     |     |     |     |     |     |
| CSE 498/499 Senior Project/Research  |     |     | X   |     |     |     |     |     |     |     |     |     | X   |     |     |     |     |

**Capstone Design**

One of the following:

| CSE 476 Embedded Systems            | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |
| CSE 481 Software Design              | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |

**Senior Electives**

Also any courses listed above
Table IV. Assessment guide for departmental outcomes.

**Goal 1. Core knowledge: To offer a challenging and broad-based education in the fundamental ideas, techniques, and tools underlying the discipline of computer engineering.**

<table>
<thead>
<tr>
<th>Program Outcomes</th>
<th>Performance Criteria</th>
<th>Implementation Plan</th>
<th>Evaluation Methods</th>
<th>Logistics</th>
<th>Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.a Mastered mathematics and science fundamentals.</td>
<td>Ability to solve problems that arise in core computer engineering courses.</td>
<td>Core courses in mathematics, statistics, physics, chemistry, and computing.</td>
<td>Course grades and sample assignments.</td>
<td>Grades reviewed by Undergraduate Advising Office.</td>
<td>Lead Advisor and Undergraduate Faculty Advisor raise issues of concern with Curriculum Committee.</td>
</tr>
<tr>
<td>1.b Applied mathematics and science concepts.</td>
<td>Ability to solve problems that arise in all computer engineering courses.</td>
<td>Senior-level courses as well as independent projects and research with faculty supervision.</td>
<td>Course grades, sample assignments, and project documentation.</td>
<td>Collected at the end of a course or project by the Advising Office. Each capstone course produces a video of a selection of projects.</td>
<td>Project complexity judged by faculty and industrial affiliates.</td>
</tr>
<tr>
<td>1.c Presented an effective oral presentation.</td>
<td>Ability to prepare and deliver an oral presentation on a capstone or independent project.</td>
<td>Capstone design courses and technical communication requirements.</td>
<td>Self-evaluation, peer-evaluation, and instructor evaluation of presentation.</td>
<td>Video-taping in technical communications courses.</td>
<td>Technical communication instructor provides direct feedback to student.</td>
</tr>
<tr>
<td>1.d Prepared effective written materials.</td>
<td>Project documentation that can be used by other students and faculty to evaluate and build on to project.</td>
<td>Term papers in senior courses. Capstone design course documentation and technical communication requirements.</td>
<td>Cross-presentations prepared using materials from other project group.</td>
<td>Web-based project documentation archived for review by faculty and future students.</td>
<td>Fellow students use the material to build on to projects in future courses.</td>
</tr>
</tbody>
</table>

22
Goal 2. Application: To impart the ability to apply the core knowledge of the discipline in making the many engineering tradeoffs that are central to solving problems and creating products.

<table>
<thead>
<tr>
<th>Program Outcomes</th>
<th>Performance Criteria</th>
<th>Implementation Plan</th>
<th>Evaluation Methods</th>
<th>Logistics</th>
<th>Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.a Mastered core of the discipline of computer engineering.</td>
<td>Ability to solve problems that arise in senior-level courses using concepts from core courses.</td>
<td>Core and senior-level courses in CSE.</td>
<td>Course grades and sample assignments.</td>
<td>Grades reviewed by Undergraduate Advising Office.</td>
<td>Lead Advisor and Undergraduate Faculty Advisor raise issues of concern with Curriculum Committee.</td>
</tr>
<tr>
<td>2.b Applied core concepts of the discipline to design/evaluation.</td>
<td>Completion of challenging projects in senior-level courses and in capstone design courses.</td>
<td>Core and senior-level courses in CSE.</td>
<td>Course grades and project assignments.</td>
<td>Faculty evaluation of courses.</td>
<td>Project complexity judged by faculty peers who teach the same courses.</td>
</tr>
<tr>
<td>2.c Constructs appropriate abstractions to manage complexity.</td>
<td>Developed abstractions as part of the process of solving problems and implementing ideas in all courses that emphasize design.</td>
<td>Capstone design courses and independent projects.</td>
<td>Course grades and sample assignments. Project documentation.</td>
<td>Grades reviewed by Undergraduate Advising Office. Documentation reviewed by faculty.</td>
<td>Project complexity judged by faculty peers who teach the same courses.</td>
</tr>
<tr>
<td>2.d Able to evaluate design tradeoffs.</td>
<td>Challenging projects in capstone design courses and ability to relate the thought process that went into the tradeoffs.</td>
<td>Capstone design courses.</td>
<td>Presentations and documentation in capstone design courses.</td>
<td>Presentation to peers with specific requirements to highlight tradeoffs and reasoning behind them.</td>
<td>Faculty provide feedback through evaluations in design courses.</td>
</tr>
<tr>
<td>2.e Completed a challenging design project.</td>
<td>Successful demonstration as part of capstone design course including video produced by UWTV.</td>
<td>Capstone design and research assistantship opportunities.</td>
<td>Demonstration at end of capstone design experience. Participation in undergraduate research seminar.</td>
<td>Presentation to entire department at graduation ceremony through video production. Poster session at industrial affiliates meeting.</td>
<td>Peers, faculty, and industry affiliates judge project relevance and comparison to state-of-the-art.</td>
</tr>
</tbody>
</table>
Goal 3. Educational renewal: To prepare our students to continually renew their education in a rapidly developing discipline so that they may realize their full potential throughout their career.

<table>
<thead>
<tr>
<th>Program Outcomes</th>
<th>Performance Criteria</th>
<th>Implementation Plan</th>
<th>Evaluation Methods</th>
<th>Logistics</th>
<th>Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.a Facile in using tools of the discipline.</td>
<td>Ability to solve problems that arise in senior-level courses using concepts from core courses.</td>
<td>Project-oriented courses, capstone design, independent projects.</td>
<td>Course grades and sample assignments.</td>
<td>Grades reviewed by Undergraduate Advising Office.</td>
<td>Lead Advisor and Undergraduate Faculty Advisor raise issues of concern with Curriculum Committee.</td>
</tr>
<tr>
<td>3.b Worked on a multi-person design team.</td>
<td>Capstone design experience guarantees all students participate in group work.</td>
<td>Capstone courses and independent projects.</td>
<td>Presentations and demonstrations of work.</td>
<td>Presentation to entire department at graduation ceremony through video production. Poster session at industrial affiliates meeting.</td>
<td>Directly from faculty and/or TAs managing and meeting with groups in capstone design or project courses.</td>
</tr>
<tr>
<td>3.c Participated in diverse learning opportunities.</td>
<td>Participation in industry internships, independent study/research with faculty, and/or teaching assistantship on campus.</td>
<td>Career development activities by Industrial Affiliates Program, seminars organized by ACM chapter, career preparation workshops.</td>
<td>Coop program evaluation, poster sessions at affiliates meeting, undergraduate research seminar, student evaluations of TA.</td>
<td>Keep track of participation in student database. Collect project reports for faculty review.</td>
<td>Directly from faculty involved in supervision (whether research or teaching). Post-internship surveys of students and employers.</td>
</tr>
<tr>
<td>3.d Assimilated knowledge of tools or concepts as needed.</td>
<td>Challenging capstone design experience.</td>
<td>Contact with member companies of Industrial Affiliates Program and seminars organized by ACM chapter.</td>
<td>Completion of design project with minimal guidance.</td>
<td>Successful demonstration at end of capstone design course.</td>
<td>Peers, faculty, and industry affiliates judge project relevance and comparison to state-of-the-art.</td>
</tr>
</tbody>
</table>
Goal 4. Social context: To foster personal development not only in the social aspects of the discipline itself but also in terms of understanding the impact of our field on society as a whole.

<table>
<thead>
<tr>
<th>Program Outcomes</th>
<th>Performance Criteria</th>
<th>Implementation Plan</th>
<th>Evaluation Methods</th>
<th>Logistics</th>
<th>Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.a Interacted with industry practitioners.</td>
<td>Completed industry internship. Poster at industrial affiliates meeting. Attendance at department colloquia and ACM-sponsored industry speaker events.</td>
<td>Chair, faculty, and advising staff leadership. Presentations at orientations.</td>
<td>Participation at annual affiliates meeting. Internship participation rates.</td>
<td>Coop and Placement office records, Advising Office records in CSE database, observation at affiliates meeting.</td>
<td>Through ACM sponsored events regarding job search and career choices.</td>
</tr>
<tr>
<td>4.b Considered benefits of graduate study.</td>
<td>Participation and/or inquiries about independent study/research.</td>
<td>Bi-annual research nights to expose students to research in department.</td>
<td>Participation in independent study/research. Involvement of our graduate students with undergraduates.</td>
<td>Advising Office records all participation in independent study/research and directs students to appropriate faculty for graduate school advice.</td>
<td>Through ACM sponsored events regarding graduate school options.</td>
</tr>
<tr>
<td>4.c Understands principal concepts of professional ethics.</td>
<td>Ability to discuss issues, if not solutions, specific to computing technologies.</td>
<td>Ethics and society course content, ethics and society seminar, written analysis of societal impact of capstone projects.</td>
<td>Critique in technical writing courses coupled with capstone courses.</td>
<td>Part of web-based project documentation.</td>
<td>Directly by technical communications and CSE faculty in coupled courses.</td>
</tr>
<tr>
<td>4.d Evaluated implications of work in the discipline to society as a whole.</td>
<td>Written analysis of capstone project with respect to societal impact.</td>
<td>VLPA requirements, ethics and society seminar, department colloquia, ACM-sponsored presentations.</td>
<td>Critique in technical writing courses coupled with capstone courses.</td>
<td>Part of web-based project documentation.</td>
<td>Directly by technical communications and CSE faculty in coupled courses.</td>
</tr>
</tbody>
</table>
Assessment (continued)

**Course grades:** The Undergraduate Advising Office reviews all grades each quarter and identifies students that are not making satisfactory progress towards the degree. These students are called in for an appointment with an advisor to discuss what measures they should take to remedy the situation. The department has a volunteer graduate student-run tutoring program that provides direct help to students or helps them form study groups. Course teaching assistant also perform this latter role in ensuring their students are working smart and benefiting from each other without over-stepping bounds on collaboration established for the course. The average student grade is approximately 3.4 (out of 4). Although this may seem relatively high, one must bear in mind that the average GPA of our incoming majors is 3.7.

**Course student evaluations:** Every course in the department is formally evaluated by students. Anonymized histograms of the results for faculty and teaching assistants are broadcast by the Chair to the entire department population. Results over the past several years, since this policy was instituted, have been steadily improving. An average evaluation five years ago was 3.7-3.8 (out of 5) and today is closer to 4.2.

**Faculty self-evaluations:** Every time a faculty member completes a course they must file a self-evaluation that discusses what worked well and what didn’t both for the instructor specifically as well as the material of the course. This reflective writing exercise provides invaluable material for future instructors.

**Student surveys upon entry and exit from the department:** These surveys include the names of students so that we can track the difference when they complete the program. Students are asked to rate themselves on the following 15 questions:

1. Ability to apply engineering theory and principles to the work environment.
2. Ability to design a system, component, or process to meet desired needs.
3. Ability to communicate effectively: on an interpersonal level; in formal presentations; in technical writing.
4. An understanding of professional and ethical responsibility.
5. Recognize the need for lifelong learning.
6. Ability to use techniques, skills and modern engineering tools necessary for engr practice.
7. Ability to design and conduct experiments.
8. Ability to analyze and interpret data.
9. Ability to function on multi-disciplinary teams.
10. A knowledge of contemporary issues in computing.
11. Ability to identify, formulate and solve engineering problems.
12. Recognize the value of hands-on experience.
13. Ability to think critically.
14. Ability to understand the implications of rapid technological change.
15. Understanding of the work place and work-related social dynamics.

We find that the most valuable aspects of this survey are not the specific answers students provide. As one can imagine, students are hardly prepared to answer the above questions accurately when they first enter the department as they are quite unsure as to their meaning. However, we do find that it sets student’s expectations for what their educational experience should provide and they begin to stop thinking of their courses as obstacles to surmount. It also provides a context in which they can make suggestions or lodge complaints about their experience in the department. As they leave the program, seeing the same questions again with their original answers (during their pre-graduation appointments) gives them an increased awareness of the issues they need to address in their comments (a listing of the last year’s is available in the appendix). We do not present the numerical values since they provide little, if any, information due to the variance in interpretation among students and even by the same student at entry and exit.
Surveys related to industry employment (pre/post and employer) through the College of Engineering’s Coop Office: CSE has worked with the Coop Office to develop surveys to be used by student before leaving for their internship/coop experience, upon their return, and by their employer after the student has returned to UW. The questions on this survey are listed here and in the information provided on our coop office in Appendix II where the relation of the questions to ABET a-k criteria is also discussed.

1. apply engineering theory & principles to work
2. design system, component or process to meet desired need
3. professional and ethical responsibility
4. interpersonal communication skills
5. formal presentation communication skills
6. technical writing skills
7. recognize need for lifelong learning
8. use techniques, skills, and modern tools necessary for practice
9. design and conduct experiments
10. analyze and interpret data
11. function on multi-disciplinary teams
12. knowledge of contemporary issues
13. identify, formulate, and solve engineering problems
14. recognize value of hands-on experience
15. critical thinking
16. managing change
17. understanding office dynamics

The responses for the past year are summarized in the graphic below. Note that the primary areas of improvement are in questions 17, 1, 2, 13, and 8. These are precisely what we would hope to see. Unfortunately, industry does not involve interns in much multi-disciplinary activity. Second internships with the same company tend to show stronger results in that area. Interestingly, the only area where student actually feel worse-off after completing an internship is 15. This may be an indication of the approach taken by many companies in not giving large responsibilities to short-term workers.

Capstone design projects and video productions: We capture most of the projects from our capstone projects in video productions that highlight the projects and their utility and/or cool-factor as well as the students’ own reflections on their projects. The department now has an extensive collection of these available on-line at: www.cs.washington.edu/info/videos/. The
videos are shown to our industrial affiliates, prospective students, parents at graduation ceremonies, and are broadcast on UWTV on a regular basis. They are professionally produced by UWTV and provide an important resource for the department’s outreach efforts. In addition, they provide an excellent record of what our students were able to accomplish so that we can verify that our capstone design experience are state-of-the-art or a bit ahead of industry practice. We target our capstone projects to be where commercial products will be in a year or two after the student graduates. Many of our graduates have sent us e-mail when they discover a commercial product similar to their capstone design project. This projects a positive image of the department as being cognizant of the cutting edge and able to bring their students to practice at that level.

Writing assessment through the Department of Technical Communications: As part of the college-wide writing assessment program, two of our faculty (Kautz and Seitz), together with a faculty member from technical communications (Plumb), evaluated a random sample of senior-level Computer Engineering student papers during the Spring of this year. In order to improve inter-rater reliability, the three raters first looked at three "calibration papers," then met to discuss their ratings. Discussion at the meeting led to a better understanding of the three categories of Not Acceptable, Competent, and Strong. After the meeting, the three raters evaluated 16 other papers. Plumb evaluated all 16 papers, whereas Kautz and Seitz evaluated 8 papers each for a total of 32 ratings. No papers produced unacceptable differences among the raters (i.e., Not Acceptable vs. Competent or Not Acceptable vs. Strong). The papers used for the process were from TC 333 (Advanced Technical Writing and Oral Presentation). The TC 333 papers were written by individual Computer Engineering students and were randomly selected from the TC 333 papers from spring quarter 2000 through winter quarter 2001. The original three categories were Not Acceptable, Competent, and Strong. The Competent category was subsequently subdivided to include three categories: Low Competent, Competent, and High Competent. In order to get a Not Acceptable rating, a paper needed to show evidence that the student was not ready to write on the job. Of the 32 ratings, 4 were Not Acceptable, 3 were Low Competent, 17 were Competent, 7 were High Competent, and 1 was Strong. Two papers were rated Not Acceptable by both evaluators. In addition to this coarse-grained evaluation, the evaluation team considered the specific Performance Outcomes for UW Engineering Writing and noted when an individual outcome was weak. The five most frequently checked outcomes (thus the weakest), in order of most weak to less weak are listed below, with the number of times the outcome was checked:

- Uses proper citation form. (14)
- Includes citations for other's ideas, including any information and non-textual material from sources outside the writer. (13)
- Uses correct punctuation, grammar, usage, and spelling. (13)
- Content supports the purpose thoroughly and concisely. (10) Note: The evaluators discussed the problem of the duality in this outcome and the need to revise it. Most of these checks dealt with a lack of thoroughness rather than a lack of conciseness.
- Exhibits a logical progression and structures the content to represent that logical progression. (8)

In conclusion, the fact that 2 of the 16 papers (or 12.5%) were deemed Not Acceptable by both raters shows room for program improvement. The weak areas in these student papers spanned nearly all aspects of writing, from grammar and spelling to content and organization. Across the board, papers were weak in using sources when appropriate and citing sources using a conventional format.

In the short term, the recommendation is that more emphasis should be given (both in the stand-alone writing courses and in the departmental courses) to assigning writing that involves using sources. If the papers had shown consistent weaknesses primarily in content areas, these weaknesses could have, perhaps, been ascribed to a weak assignment or lack of motivation. However, the weak performance on the broad range of desired outcomes may call for more writing practice of various types in varied settings. The Director of the College’s communication program
has agreed to work with CS&E to identify a senior-level CS&E course from which to sample student papers so that a wider variety of papers can be evaluated for the next assessment cycle. In the meantime, the department is also experimenting with linking capstone design courses with technical communication courses so that students can write more about their projects and be more motivated in their writing. It is too early to report on this experiment as it is currently in progress.

**Presentations (posters, demos) at the annual affiliates meeting:** Undergraduates participating in research are encouraged to prepare posters and demos for our annual industrial affiliates meeting. This gives them an opportunity to discuss their projects with leading industry researchers and receive valuable feedback. Participation rates at this event are steadily increasing among our undergraduates as they participate more heavily in research. Feedback from the affiliate members is usually provided directly to the faculty supervising the projects. Student in capstone design courses when the affiliate meeting takes place also prepare posters on their course projects.

**Alumni surveys:** Assessing how our students to when they leave our program is done through alumni feedback on our own alumni web pages as well as through surveys conducted by the Office of Educational Assessment. Graduates are asked to complete the survey 1 year after graduation. They do so with a close to 40% participation rate. Another survey is conducted after five years but this has too low a participation rate to be useful. Students are asked to rate themselves in the following 14 areas:

1. Writing
2. Speaking
3. Critically analyzing written info.
4. Learning independently
5. Understanding/appreciating the arts
6. Understanding/applying scientific principles/methods
7. Understanding/applying quantitative principles/methods
8. Defining/solving problems
9. Working cooperatively in a group
10. Readiness for a career
11. Readiness for advanced education
12. Understanding differing philosophies and cultures
13. Understanding the interaction of society and the environment
14. Recognizing your responsibilities, rights and privileges as a citizen

The survey is conducted every two years and the data for the past 4 surveyed classes is shown below.
The important messages from this summary is that our last ABET accreditation report was correct in its identification of ethics and societal impact as an area in which our curriculum is lacking. With the measures we are taking to remedy this situation (see section A.3), we hope to see these numbers change over coming years. Other low marks are for questions 5, which we do not directly address in our curriculum and do not plan to do so in the general sense of this question, and 2, which has led us to develop more opportunities for students to speak in front of their peers on technical concepts. Most of this is being done in the capstone design courses.

In addition to these OEA-sponsored surveys, the department conducts its own alumni survey, on a continuous basis, on our alumni web pages. We provide alumni with permanent e-mail addresses and a way to re-link with fellow students. This makes the pages highly visited. We collect broad comments using the following 3 questions as starting points:

1. Thinking back, is there a particular course(s) and/or instructor(s) who made a difference in your ability to succeed?
2. What did the department do well?
3. What could the department have done better?

The responses to these questions are available to all faculty to view and they are reminded of their existence on an annual basis. Comments are processed in the same way as our exit survey results and have lead to kudos to specific faculty as well as suggested changes to our curriculum committee.

*Process Results and Improvements*

The feedback loops described above are working. Section A.6 provides a three-page table outlining recent improvements made and how they were initiated. We do not reproduce it here. In addition, several improvements are suggested from the assessments described above in this section. In general, the ABET self-study process has proven an excellent experience for the department in spurring the development of additional assessment methods.
4. **Professional Component**

The Computer Engineering program curriculum is described in detail in the department undergraduate programs handbook which is appended to this self-study in its own binding. It can also be found on the web at: www.cs.washington.edu/education/ugrad/handbook/CSEhandbook.pdf. The curriculum is designed by faculty as a whole through the Curriculum Committee and the Undergraduate Faculty Coordinator. After changes are ratified by the faculty, they also must receive approval by the College’s Educational Policy Committee and the University’s Curriculum Committee. The handbook, the tables above listing our courses and their mapping to our objectives, and a detailed curriculum plan outlined in Appendix I provide all the required information. It is not repeated here in its entirety for the sake of brevity.

In summary, the program consists of the following requirements based on a 180 credit degree program under a quarter system. There are two options based on hardware or software emphasis in embedded computing systems. All ABET requirements are met as can be seen in Table A.1.

- **Core requirements (32 credits):** CSE 142, 143, 321, 322, 326, 341, 370, 378
- **Additional Computer Engineering requirements (17 credits):** CSE 451, 461, EE 215 and 233
- **Hardware Option (18 credits):** CSE 467, 471, EE 331; CSE 468 or 477 (capstone)
- **Software Option (16-17 credits):** CSE 403, 466; one of CSE 401, 457, or 471; CSE 476 or 481 (capstone)
- **CSE Electives (13 credits):** all 400-level CSE courses not listed above
- **Math (25 credits):** MATH 124, 125, 126, 307, 308, MATH/STAT 390
- **Science (20 credits):** PHYS 121/131, 122/132, 123/133; CHEM 142
- **VLPA and I&S (30 credits):** from UW provided list.
- **Written and Oral Communication (12 credits):** English Composition, TC 231, 333
- **Free Electives (13-15 credits; depending on option chosen)**
5. Faculty

The department operates with minimal hierarchy. Faculty of all ranks have similar teaching loads and are similarly evaluated including annual peer reviews of teaching and meetings with the Chair to discuss activity level and performance in teaching, research, and service. Annual evaluations include a reflective statement from each faculty member to the Chair outlining their assessment of the past year and goals for the future.

There are 42 faculty, 2 of these hold partial appointments (one with EE and one with the Business School). They are broken down as follows:

- 21 full professors
- 5 associate professors
- 9 assistant professors
- 2 research assistant professors
- 3 senior lecturers
- 2 lecturers

All are involved with both our undergraduate programs. Five faculty members in the department are recipients of University awards: 4 are winners of the Distinguished Teaching Award (Borriello, Diorio, Ebeling, and Salesin) and 1 is winner of the Distinguished Graduate Mentor Award (Notkin). Recently, one of our senior graduate student teaching assistants received a College award for teaching. The ACM student chapter honors a faculty member (at our annual end-of-year party) with an ACM Distinguished Teaching Award decided by a student vote. Finally, the Bob Bandes award honors the Department’s top TAs based on nominations from students.

The faculty is extremely active in research and our department is consistently ranked in the top 10 in the nation in our discipline (and usually in the top 5 or 6). There are strong ties with industry with many faculty receiving research and educational support from corporations, through consulting, and past experience in industry. National visibility is high with several faculty members serving on National Science Foundation advisory boards, National Research Council study panels, and Department of Defense research organizations. One of our faculty (Snyder) is the leader in a drive to develop university and high school courses in information technology literacy (see our own CSE 100 taught in conjunction with the Information School).

Our faculty continually strives to improve its teaching performance. The Chair broadcasts anonymous histogram results of student evaluations so that all faculty members know where they stand. All our courses are evaluated by students. Peer evaluations of teaching are taken very seriously. Graduate student instructors are provided with a seminar to develop their skills (CSE 590IT). The faculty takes full advantage of the resources the University and College have to offer. In particular, many of our faculty have had representatives from the Center for Instructional Development and Resources (CIDR – http://depts.washington.edu/cidrweb/ ) visit their classrooms and conduct student interviews and make suggestions for improving their lectures. Interactions with the College of Engineering’s Center for Engineering Learning and Teaching (CELT – www.engr.washington.edu/~celtweb/ ) have included: individual faculty consultations with instructional consultants; faculty participation in CELT workshops and attendance at CELT sponsored talks; collaborations in educational research proposals; evaluation of educational projects; CSE funding of a project with the Coop office to investigate learning outcomes resulting from co-op experiences; and classroom assessment.

The CSE faculty covers a wide range of the discipline and, collectively, has a wide range of experience in our discipline including: industry research labs, start-ups, and advanced development. Faculty members are highly visible in their respective communities with many serving as chairs of major conferences and enjoying extensive networks of colleagues on an international, as well as national, level.
6. Facilities

Building

CSE continues to occupy woefully inadequate space both in quality and quantity. We are housed in a 35+ year-old building with no classrooms and limited conference/meeting space. We have had to convert our hallways to breakout rooms with cubicles and whiteboards. The University administration claims that a new building with 2.5 times the assignable square feet for our department is the highest institutional priority. However, there has been no progress for several years. Given that we have converted all our available space to office or laboratory use, we are now severely limited in our ability to recruit faculty with laboratory needs and take on new initiatives. The department has traditionally assigned a high priority to undergraduate laboratory space. In fact, there are currently only 3 graduate laboratories as opposed to the 5 undergraduate laboratory spaces. The need for new facilities has been identified as our major need by both our External Advisory Committee and the University-appointed 10-year Review Committee.

Classrooms

The classrooms used by our students and faculty are scattered throughout campus with the majority being in the new Mary Gates Hall for Undergraduate Education and the new Electrical Engineering building. Classrooms in both buildings are state-of-the-art with networking taps and data projection equipment in each room. However, the inconsistency of room assignments (several of our courses each quarter are assigned classrooms 10-15 minutes across campus) makes it difficult to rely on a standard infrastructure. Many of our faculty have not taken steps to use educational technology because of this uncertainty.

Laboratories and Infrastructure

Laboratory and computing facilities available to computer engineering students include two general purpose modern computing laboratories, and three special purpose state-of-the-art laboratories geared to specific areas in the computer engineering curriculum.

The general purpose computing labs provide approximately 75 Pentium P-II computers running Windows 2000 and a number of laser printers, and are shared by all computer science and engineering students. The PCs have 550MHz & 700MHz CPUs, 128mb RAM and 19” monitors (with anticipated upgrades to 1GHz-class machines in the next academic year). These labs support general computer science and engineering courses, as well as certain computer engineering courses that do not require special hardware, and offer a very wide array of general purpose software development tools (e.g., MS Visual Studio, C, C++, Java), as well as specialized hardware design tools (e.g., DesignWorks, L-Edit, PSpice, and Xilinx Foundation Tools.)

The three special purpose labs that support computer engineering courses include: the VLSI Design Lab, the Integrated Systems Design Lab (ISDL) and the Embedded Systems Lab.

The ISDL and the Embedded Systems Lab share a common space, and alternate use of that space to match course offerings. The ISDL is populated with 12 Pentium P-III computers (800MHz CPU, 256mb RAM, 19” monitors) augmented by state-of-the-art digital and embedded computer hardware/software development equipment. A large-format (11”x17”) laser printer is available for producing high quality, easy-to-read design prints. The PC’s are loaded with contemporary development software, such as Xilinx Foundation Tools, Keil SDT, ARM SDT, and Palm CodeWarrior, as well as a wide variety of general-purpose software design tools. Development kits available to students contain hardware to facilitate development on popular microcontroller and programmable logic platforms. In addition, a complete inventory of currently used electronic components is maintained.
The VLSI Design Lab consists of 18 P-III PCs and a laser printer and is dedicated to the VLSI Design class (CSE 468). The PCs have a 800MHz processor, 256mb RAM, 19" monitors, and trackballs, and run the Windows2000 operating system. Software includes a wide variety of software design tools for general program development; as well as specialized VLSI design packages such as Capture and PSpice (from OrCAD EDA), and DesignPro (Tanner Research).

A full time lab manager supports the special purpose laboratories, and is available to advise, consult and problem solve with students in the use of the test equipment, project hardware components and software tools.

All laboratories are backed by a very robust server and network infrastructure, consisting of Windows and Unix servers (for file serving, compute cycles, printing, email, and web services), interconnected with a 100mb/1gb switched network. A building-wide (and its neighborhood) 802.11b wireless network is also available for students, faculty, and staff.

Students have the opportunity to use all of these facilities in learning the use of modern engineering tools. Facilities and infrastructure are constantly updated to maintain student access to the most modern tools. Access to the general purpose laboratories is available throughout each student's career in the department, while access to the specialized labs restricted to students actively enrolled in specific computer engineering courses, or doing senior project or undergraduate research work sponsored by a faculty member. Award-winning examples of engineering projects produced by students using these facilities can be viewed at www.cs.washington.edu/info/videos/.

Other Undergraduate Program Space

Undergraduate students have a small lounge (400 sq. ft.) which contains several smaller office populated with equipment that can be used for independent projects. The space is quite inadequate and does not provide facilities for hardware-oriented projects (prototyping benches, instrumentation, etc.).

Recently (Spring 2001), through industry gifts and some overhead recapture, we have leased 3500 sq. ft. of off-campus space that is being used primarily for inter-disciplinary projects involving students from CSE and other departments on campus at both the undergraduate and graduate levels. See http://point5.cs.washington.edu for more information.
7. Institutional Support and Financial Resources

The University of Washington operates on a biennial budget as does the entire state government. The department’s annual operating budget is currently at $1.2M and is further augmented by industry grant through our industrial affiliates program and from grants obtained directly by faculty. The department aggressively pursues corporate donations. Large donors include: Microsoft (software and cash), Intel (equipment and cash), Ford (cash), and Tektronix (equipment matching grants). Over 70 companies contribute fees to our industrial affiliates program providing much needed unrestricted funds.

The current funding level is adequate for the needs of our educational mission but relies on active faculty who are able to obtain large industry cash and equipment donations. Our state funding goes primarily to support personnel that keep our extensive infrastructure up and running. Students do not have to manage machines themselves and can rely on a highly available and robust infrastructure for all their work and communications.
8. Program Criteria

The department has a strong faculty base on which it has built excellent educational programs that have earned it the University of Washington’s inaugural Brotman Award for Educational Excellence. It is ranked among the top 10 in the nation in terms of both its research and educational reputation. It has five teaching award recipients among its faculty.

The curriculum for the Computer Engineering program meets all the ABET requirements for accreditation. Evidence of this has been provided above and in the attached appendices.
9. Cooperative Education Criteria

Not applicable.
10. General Advanced-Level Program

No accreditation is sought at the graduate level.
Appendix I - Additional Program Information

A. Tabular Data for Program

   Table A.1. Basic level Curriculum
   Table A.2. Course and Section Size Summary
   Table A.3. Faculty Workload Summary
   Table A.4. Faculty Analysis
   Table A.5. Support Expenditures

B. Course Syllabi

C. Faculty Curriculum Vitae

D. Student exit survey comments for the past two years
Table A.1. Basic-Level Curriculum
(Computer Engineering)

<table>
<thead>
<tr>
<th>Year/Quarter</th>
<th>Course (Department, Number, Title)</th>
<th>Category (Credit Hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Math &amp; Basic Sciences</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Check if Contains Design</td>
</tr>
<tr>
<td>1 Yr/ 1st Qtr</td>
<td>Math 124 Calc w/Anal Geometry</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Chem 142 General Chemistry</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>English Composition</td>
<td></td>
</tr>
<tr>
<td>1 Yr/ 2nd Qtr</td>
<td>Math 125 Calc w/Anal Geometry</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Phys 121/131 Mechanics</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>VLPA/I&amp;S Elective</td>
<td></td>
</tr>
<tr>
<td>1 Yr/ 3rd Qtr</td>
<td>Math 126 Calc w/Anal Geometry</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Physics 122/132 Electromag</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>VLPA/I&amp;S Elective</td>
<td></td>
</tr>
<tr>
<td>2 Yr/ 1st Qtr</td>
<td>Math 307 Differential Equations</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Phys 123/133 Waves</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>CSE 142 Computer Programming I</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>VLPA/I&amp;S Elective</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 Yr/ 2nd Qtr</td>
<td>Math 308 Linear Algebra</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>CSE 143 Computer Programming II</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>VLPA/I&amp;S Elective</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 Yr/ 3rd Qtr</td>
<td>Math/Stat 390 Prob &amp; Statistics</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>EE 215 Fundamentals of EE</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>TC 231 Intro. to Technical Writing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VLPA/I&amp;S Elective</td>
<td></td>
</tr>
<tr>
<td>3 Yr/ 1st Qtr</td>
<td>CSE 321 Discrete Structures</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>CSE 370 Intro to Digital Design</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>EE 233 Circuit Theory</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Free Electives</td>
<td></td>
</tr>
<tr>
<td>3 Yr/ 2nd Qtr</td>
<td>CSE 326 Data Structures</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>CSE 378 Mach Org/Assembly Lang</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>EE 331 (Hardware option) or CompE Elective (Software option)</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 Yr/ 3rd Qtr</td>
<td>CSE 341 Programming Languages</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>CSE 322 Formal Models</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>VLPA/I&amp;S Elective</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Free Elective</td>
<td></td>
</tr>
</tbody>
</table>

(continues on next page)
### Table A.1. Basic-Level Curriculum

*(Computer Engineering – cont’d)*

<table>
<thead>
<tr>
<th>Year/Quarter</th>
<th>Course (Department, Number, Title)</th>
<th>Category (Credit Hours)</th>
<th>Check if Contains Design</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Math &amp; Basic Sciences</td>
<td>Engineering Topics</td>
</tr>
<tr>
<td>4 Yr/1st Qtr</td>
<td>CSE 451 Operating Systems 4</td>
<td></td>
<td>(✓)</td>
</tr>
<tr>
<td>Hardware</td>
<td>CSE/EE 461 Networks 4</td>
<td></td>
<td>(✓)</td>
</tr>
<tr>
<td>Option</td>
<td>CompE Elective 4</td>
<td></td>
<td>( )</td>
</tr>
<tr>
<td></td>
<td>Free Elective 3</td>
<td></td>
<td>( )</td>
</tr>
<tr>
<td>4 Yr/2nd Qtr</td>
<td>CSE 467 Advanced Digital Design 4</td>
<td></td>
<td>(✓)</td>
</tr>
<tr>
<td></td>
<td>CSE 471 Design &amp; Organization 4</td>
<td></td>
<td>(✓)</td>
</tr>
<tr>
<td></td>
<td>CompE Elective 3</td>
<td></td>
<td>( )</td>
</tr>
<tr>
<td></td>
<td>CompE Elective 3</td>
<td></td>
<td>( )</td>
</tr>
<tr>
<td>4 Yr/3rd Qtr</td>
<td>CSE 468 VLSI or CSE 477 Digital</td>
<td>5</td>
<td>(✓)</td>
</tr>
<tr>
<td></td>
<td>Systems Design Capstone</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TC 333 Adv. Tech Writing &amp; Oral</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Presentation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CompE Elective 3</td>
<td></td>
<td>( )</td>
</tr>
<tr>
<td></td>
<td>Free Elective 4</td>
<td></td>
<td>( )</td>
</tr>
<tr>
<td>4 Yr/1st Qtr</td>
<td>CSE 403 Software Engineering 4</td>
<td></td>
<td>(✓)</td>
</tr>
<tr>
<td>Software</td>
<td>CSE 451 Operating Systems 4</td>
<td></td>
<td>(✓)</td>
</tr>
<tr>
<td>Option</td>
<td>CompE Elective 4</td>
<td></td>
<td>( )</td>
</tr>
<tr>
<td></td>
<td>Free Elective 3-4</td>
<td></td>
<td>( )</td>
</tr>
<tr>
<td>4 Yr/2nd Qtr</td>
<td>CSE/EE 461 Networks 4</td>
<td></td>
<td>(✓)</td>
</tr>
<tr>
<td></td>
<td>CSE 466 Software for Embedded</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Systems</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CSE 401 Compilers or CSE 457</td>
<td>3-4</td>
<td>(✓)</td>
</tr>
<tr>
<td></td>
<td>Graphics or CSE 471 CPU Design</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Free Electives 4</td>
<td></td>
<td>( )</td>
</tr>
<tr>
<td>4 Yr/3rd Qtr</td>
<td>CSE 476 Embedded Systems or</td>
<td>5</td>
<td>(✓)</td>
</tr>
<tr>
<td></td>
<td>CSE 481 Software Design Capstone</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TC 333 Adv. Tech Writing &amp; Oral</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Presentation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CompE Elective 4</td>
<td></td>
<td>( )</td>
</tr>
</tbody>
</table>

**TOTALS-ABET BASIC-LEVEL REQUIREMENTS** 45 79-81 (55-56) 42 12-14

**OVERALL TOTAL FOR DEGREE** 180

**PERCENT OF TOTAL**

<table>
<thead>
<tr>
<th></th>
<th>25%</th>
<th>44-45% (31%)</th>
<th>23%</th>
<th>7-8%</th>
</tr>
</thead>
</table>

**MINIMUM PERCENTAGE**

|                      | 25% | 37.5%        | 7-8% | 7-8% |

---

Page 41
<table>
<thead>
<tr>
<th>Course No.</th>
<th>Title</th>
<th>No. of Sections offered in Current Year</th>
<th>Avg. Section Enrollment</th>
<th>Type of Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Computer Fluency</td>
<td>14</td>
<td>10</td>
<td>60</td>
</tr>
<tr>
<td>142</td>
<td>Computer Programming I</td>
<td>66</td>
<td>26</td>
<td>75</td>
</tr>
<tr>
<td>143</td>
<td>Computer Programming II</td>
<td>41</td>
<td>25</td>
<td>60</td>
</tr>
<tr>
<td>321</td>
<td>Discrete Structures</td>
<td>6</td>
<td>25</td>
<td>75</td>
</tr>
<tr>
<td>322</td>
<td>Formal Models</td>
<td>3</td>
<td>51</td>
<td>100</td>
</tr>
<tr>
<td>326</td>
<td>Data Structures</td>
<td>7</td>
<td>22</td>
<td>75</td>
</tr>
<tr>
<td>341</td>
<td>Programming Languages</td>
<td>6</td>
<td>27</td>
<td>75</td>
</tr>
<tr>
<td>370</td>
<td>Introduction to Digital Design</td>
<td>6</td>
<td>28</td>
<td>75</td>
</tr>
<tr>
<td>373</td>
<td>Data Structures and Algorithms</td>
<td>3</td>
<td>65</td>
<td>75</td>
</tr>
<tr>
<td>378</td>
<td>Machine Org. and Assembly Lang</td>
<td>7</td>
<td>23</td>
<td>75</td>
</tr>
<tr>
<td>401</td>
<td>Introduction to Compiler Construction</td>
<td>2</td>
<td>40</td>
<td>100</td>
</tr>
<tr>
<td>403</td>
<td>Software Engineering</td>
<td>1</td>
<td>53</td>
<td>75</td>
</tr>
<tr>
<td>410</td>
<td>Computer Systems</td>
<td>1</td>
<td>59</td>
<td>100</td>
</tr>
<tr>
<td>413</td>
<td>Prog. Langs and Implementation</td>
<td>1</td>
<td>53</td>
<td>100</td>
</tr>
<tr>
<td>415</td>
<td>Introduction to Artificial Intelligel</td>
<td>1</td>
<td>59</td>
<td>100</td>
</tr>
<tr>
<td>417</td>
<td>Algorithms and Computational Complexity</td>
<td>1</td>
<td>49</td>
<td>100</td>
</tr>
</tbody>
</table>

(continues on next page)
<table>
<thead>
<tr>
<th>Course No.</th>
<th>Title</th>
<th>No. of Sections offered in Current Year</th>
<th>Avg. Section Enrollment</th>
<th>Lecture</th>
<th>Laboratory</th>
<th>Recitation</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>421</td>
<td>Introduction to Algorithms</td>
<td>1</td>
<td>45</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>431</td>
<td>Introduction to Theory of Computation</td>
<td>1</td>
<td>38</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>444</td>
<td>Introduction to Database Systems</td>
<td>2</td>
<td>52</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>451</td>
<td>Introduction to Operating Systems</td>
<td>6</td>
<td>25</td>
<td>75</td>
<td>25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>457</td>
<td>Computer Graphics</td>
<td>2</td>
<td>52</td>
<td>75</td>
<td></td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>458</td>
<td>Computer Animation</td>
<td>1</td>
<td>15</td>
<td>60</td>
<td></td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>461</td>
<td>Introd. to Computer Networks (CSE only)</td>
<td>5</td>
<td>17</td>
<td>75</td>
<td></td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>466</td>
<td>Software for Embedded Systems</td>
<td>2</td>
<td>7</td>
<td>75</td>
<td></td>
<td>25</td>
<td></td>
</tr>
<tr>
<td>467</td>
<td>Advanced Logic Design</td>
<td>2</td>
<td>17</td>
<td>75</td>
<td></td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>468</td>
<td>Very Large Scale Integration</td>
<td>1</td>
<td>20</td>
<td>60</td>
<td></td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>471</td>
<td>Computer Design and Org.</td>
<td>2</td>
<td>19</td>
<td>75</td>
<td></td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>473</td>
<td>Introduction to Artificial Intelligence</td>
<td>2</td>
<td>45</td>
<td>75</td>
<td></td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>476</td>
<td>Embedded System Design</td>
<td>2</td>
<td>7</td>
<td>60</td>
<td></td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>477</td>
<td>Digital System Design</td>
<td>2</td>
<td>15</td>
<td>60</td>
<td></td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>481</td>
<td>Capstone Software Design</td>
<td>1</td>
<td>25</td>
<td>60</td>
<td></td>
<td></td>
<td>40</td>
</tr>
</tbody>
</table>

(continues on next page)
Table A.2. Course and Section Size Summary

*(Computer Engineering – cont’d)*

<table>
<thead>
<tr>
<th>Course No.</th>
<th>Title</th>
<th>No. of Sections offered in Current Year</th>
<th>Avg. Section Enrollment</th>
<th>Type of Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>490ab</td>
<td>Computers, Ethics, and Society</td>
<td>1</td>
<td>8</td>
<td>100</td>
</tr>
<tr>
<td>490ca</td>
<td>Computer Animation</td>
<td>1</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>490i</td>
<td>Advanced Internet Systems</td>
<td>1</td>
<td>30</td>
<td>60</td>
</tr>
<tr>
<td>490kt</td>
<td>Computer Telephony</td>
<td>1</td>
<td>25</td>
<td>60</td>
</tr>
</tbody>
</table>
Table A.3. Faculty Workload Summary
(Computer Engineering)

<table>
<thead>
<tr>
<th>Faculty Member (Name)</th>
<th>FT or PT</th>
<th>Classes Taught (Course No./Credit Hrs.)</th>
<th>Term and Year</th>
<th>Teaching</th>
<th>Research</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anderson, Richard</td>
<td>FT</td>
<td>A00 – 142 (4); W01 – 590Z (1-3); S01 – 521 (3)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Anderson, Tom</td>
<td>FT</td>
<td>A99 – 451 (4)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Arnstein, Larry</td>
<td>FT</td>
<td>A00 – 466 (4); W01 – 467 (4)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Baer, Jean-Loup</td>
<td>FT</td>
<td>A99 - 378 (4); W00 – 471 (4)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Beame, Paul</td>
<td>FT</td>
<td>A00 – 533 (3); W01 – 417 (3); S01 – 321 (4)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Borning, Alan</td>
<td>FT</td>
<td>A00 – 100 (5); S01 – 510 (3)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Borriello, Gaetano</td>
<td>FT</td>
<td>A00 – 370 (4); W01 – 476 (5)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Chambers, Craig</td>
<td>FT</td>
<td>A00 – 505 (3); W01 – 501 (3); S01 – 341 (4)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Curless, Brian</td>
<td>FT</td>
<td>A00 – 457 (4); S01 – 558 (3)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Dickey, Martin</td>
<td>FT</td>
<td>A00 – 142 (4), 590IT (1); W01 – 143 (5); S01 – 370 (4)</td>
<td></td>
<td>80</td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>Diorio, Chris</td>
<td>FT</td>
<td>W01 – 468/568 (5); S01 – 477 (5)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Domingos, Pedro</td>
<td>FT</td>
<td>A00 – 473 (3); W01 – 590PD (1-3); S01 – 592 (4)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Dooley, Emer</td>
<td>PT</td>
<td>A00 – 590IP (4); S01 – 590YB (4)</td>
<td></td>
<td>80</td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>Dugan, Ben</td>
<td>FT</td>
<td>W01 – 378 (4); S01 – 142 (4)</td>
<td></td>
<td>80</td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>Ebeling, Carl</td>
<td>FT</td>
<td>W00 – 467 (3); S00 – 477 (5)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Eggers, Susan</td>
<td>FT</td>
<td>A00 – 471 (4); S01 – 378 (4)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Etzioni, Oren</td>
<td>FT</td>
<td>A00 – 573 (3); W01 – 4901 (5)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
<tr>
<td>Fox, Dieter</td>
<td>FT</td>
<td>A00 – 590DF (1); W01 – 321 (4); S01 – 473 (3)</td>
<td></td>
<td>33</td>
<td>50</td>
<td>17</td>
</tr>
</tbody>
</table>

(continues on next page)
Table A.3. Faculty Workload Summary  
(Computer Engineering – cont’d)

<table>
<thead>
<tr>
<th>Faculty Member (Name)</th>
<th>FT or Chair</th>
<th>Classes Taught (Course No./Credit Hrs.)</th>
<th>Total Activity Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Term and Year</td>
<td>Teaching</td>
</tr>
<tr>
<td>Gribble, Steve</td>
<td>FT</td>
<td>W01 – 451 (4); S01 – 551 (3)</td>
<td>33</td>
</tr>
<tr>
<td>Halevy, Alon</td>
<td>FT</td>
<td>A00 – 594 (4); S01 – 326 (4)</td>
<td>33</td>
</tr>
<tr>
<td>Karlin, Anna</td>
<td>FT</td>
<td>A00 – 589 (4); W01 – 321 (4), 522 (3); S01 – 590Z (1-3)</td>
<td>33</td>
</tr>
<tr>
<td>Kautz, Henry</td>
<td>FT</td>
<td>A00 – 326 (4); W01 – 574 (3); S01 – 590HK (1-3)</td>
<td>33</td>
</tr>
<tr>
<td>Ladner, Richard</td>
<td>FT</td>
<td>A00 – 531 (3); W01 – 322 (3); S01 – 431 (3)</td>
<td>33</td>
</tr>
<tr>
<td>Lazowska, Ed</td>
<td>FT</td>
<td>Chair</td>
<td>0</td>
</tr>
<tr>
<td>Levy, Henry</td>
<td>FT</td>
<td>W00 – 451 (4)</td>
<td>33</td>
</tr>
<tr>
<td>Mones, Barbara</td>
<td>FT</td>
<td>W01 – 458 (4), 490AP (1-3); S01 - 490 CA (5)</td>
<td>80</td>
</tr>
<tr>
<td>Notkin, David</td>
<td>FT</td>
<td>A00 – 401 (3); W01- 590N (1); S01 – 584 (4)</td>
<td>33</td>
</tr>
<tr>
<td>Perkins, Hal</td>
<td>FT</td>
<td>A00 – 143 (5); W01 – 142 (4), 413 (3); S01 – 143 (5)</td>
<td>80</td>
</tr>
<tr>
<td>Popovic, Zoran</td>
<td>FT</td>
<td>W01 – 557 (3); S01 – 457 (4)</td>
<td>33</td>
</tr>
<tr>
<td>Rao, Rajesh</td>
<td>FT</td>
<td>W01 – 599 (4); S01 – 373 (3)</td>
<td>33</td>
</tr>
<tr>
<td>Ruzzo, Larry</td>
<td>FT</td>
<td>A00 – 322 (3); W01 – 421 (3); S01 – 401 (3)</td>
<td>33</td>
</tr>
<tr>
<td>Salesin, David</td>
<td>FT</td>
<td>at Microsoft Research</td>
<td>33</td>
</tr>
<tr>
<td>Seitz, Steve</td>
<td>FT</td>
<td>W01 – 590SS (3); S01 – 591 (4)</td>
<td>33</td>
</tr>
<tr>
<td>Sengupta, Rimli</td>
<td>FT</td>
<td>A00 – 321 (4); S01 – 322 (3)</td>
<td>33</td>
</tr>
<tr>
<td>Shapiro, Linda</td>
<td>FT</td>
<td>A00 – 373 (3); W01 – EE562 (3); S01 – 576 (3)</td>
<td>33</td>
</tr>
</tbody>
</table>

(continues on next page)
Table A.3. Faculty Workload Summary
(Computer Engineering – cont’d)

<table>
<thead>
<tr>
<th>Faculty Member (Name)</th>
<th>FT or</th>
<th>Classes Taught (Course No./Credit Hrs.) Term and Year</th>
<th>Total Activity Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Snyder, Larry</td>
<td>FT</td>
<td>A00 – 378 (4); W01 – 548 (3); S01 – 100 (5)</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>Suciu, Dan</td>
<td>FT</td>
<td>A00 – 444 (3); W01 – 544 (3); S01 – 590DS (3)</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>Tanimoto, Steve</td>
<td>FT</td>
<td>A00 – 341 (4); W01 – 373 (3); S01 – 415 (5)</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>Tompa, Martin</td>
<td>FT</td>
<td>A99 – 321 (4); S00 – 431 (3)</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>Weld, Daniel</td>
<td>FT</td>
<td>W00 – 490I (5)</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>Wetherall, David</td>
<td>FT</td>
<td>W01 – 461 (4)</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>Zahorjan, John</td>
<td>FT</td>
<td>A00 – 451 (4); W01 – 142 (4); S01 – 481 (5)</td>
<td>33</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>Name</td>
<td>Age</td>
<td>Rank</td>
<td>FT or PT</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----</td>
<td>-----------------------</td>
<td>----------</td>
</tr>
<tr>
<td>Richard Anderson</td>
<td>42</td>
<td>Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Thomas Anderson</td>
<td>39</td>
<td>Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Larry Arnstein</td>
<td>37</td>
<td>Research Ass’t Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Jean-Loup Baer</td>
<td>64</td>
<td>Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Paul Beame</td>
<td>41</td>
<td>Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Alan Borning</td>
<td>51</td>
<td>Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Gaetano Borriello</td>
<td>42</td>
<td>Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Craig Chambers</td>
<td>36</td>
<td>Associate Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Brian Curless</td>
<td>34</td>
<td>Assistant Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Martin Dickey</td>
<td>53</td>
<td>Senior Lecturer</td>
<td>FT</td>
</tr>
<tr>
<td>Chris Diorio</td>
<td>39</td>
<td>Associate Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Pedro Domingos</td>
<td>35</td>
<td>Assistant Professor</td>
<td>FT</td>
</tr>
<tr>
<td>Emer Dooley</td>
<td>41</td>
<td>Lecturer</td>
<td>PT</td>
</tr>
</tbody>
</table>

(continues on next page)
## Table A.4. Faculty Analysis
(Computer Engineering – cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Rank</th>
<th>FT or PT</th>
<th>Highest Degree</th>
<th>Institution from which Highest Degree Earned &amp; Year</th>
<th>Years of Experience</th>
<th>Professional Registration (Indicate State)</th>
<th>Level of Activity (high, med, low, none) in:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Professional Society</td>
<td>Research Work in Industry</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Highest Degree</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ben Dugan</td>
<td>31</td>
<td>Lecturer</td>
<td>FT</td>
<td>MS</td>
<td>Washington, 1995</td>
<td>2</td>
<td>N</td>
<td>L</td>
</tr>
<tr>
<td>Carl Ebeling</td>
<td>51</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>CMU, 1986</td>
<td>0</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td>Susan Eggers</td>
<td>58</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Berkeley, 1989</td>
<td>4</td>
<td>L</td>
<td>H</td>
</tr>
<tr>
<td>Oren Etzioni</td>
<td>37</td>
<td>Associate Professor</td>
<td>FT</td>
<td>PhD</td>
<td>CMU, 1990</td>
<td>0</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>Dieter Fox</td>
<td>34</td>
<td>Assistant Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Bonn, 1998</td>
<td>0</td>
<td>L</td>
<td>H</td>
</tr>
<tr>
<td>Steve Gribble</td>
<td>28</td>
<td>Assistant Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Berkeley, 2000</td>
<td>0</td>
<td>L</td>
<td>H</td>
</tr>
<tr>
<td>Alon Halevy</td>
<td>37</td>
<td>Associate Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Stanford, 1993</td>
<td>4</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>Anna Karlin</td>
<td>41</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Stanford, 1987</td>
<td>6</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td>Henry Kautz</td>
<td>44</td>
<td>Associate Professor</td>
<td>FT</td>
<td>PhD</td>
<td>U-Rochester, 1988</td>
<td>13</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>Richard Ladner</td>
<td>57</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Berkeley, 1971</td>
<td>0</td>
<td>L</td>
<td>H</td>
</tr>
<tr>
<td>Edward Lazowska</td>
<td>50</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Toronto, 1977</td>
<td>0</td>
<td>H</td>
<td>L</td>
</tr>
<tr>
<td>Henry M. Levy</td>
<td>48</td>
<td>Professor</td>
<td>FT</td>
<td>MS</td>
<td>Washington, 1981</td>
<td>8</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>Barbara Mones-Hattal</td>
<td>49</td>
<td>Senior Lecturer</td>
<td>FT</td>
<td>MFA</td>
<td>Rhode Island School of Design, 1979</td>
<td>3</td>
<td>H</td>
<td>L</td>
</tr>
</tbody>
</table>

(continues on next page)
Table A.4. Faculty Analysis  
(Computer Engineering – cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Rank</th>
<th>FT or PT</th>
<th>Highest Degree</th>
<th>Institution from which Highest Degree Earned &amp; Year</th>
<th>Years of Experience</th>
<th>Professional Registration (Indicate State)</th>
<th>Level of Activity (high, med, low, none) in:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Total Faculty</td>
<td></td>
<td>Professional Society</td>
<td>Research</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>This Institution</td>
<td></td>
<td>Govt./Industry Practice</td>
<td></td>
</tr>
<tr>
<td>David Notkin</td>
<td>46</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>CMU, 1984</td>
<td>0</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>John Hal Perkins</td>
<td>49</td>
<td>Senior Lecturer</td>
<td>FT</td>
<td>MS</td>
<td>Cornell, 1982</td>
<td>2</td>
<td>18</td>
<td>3</td>
</tr>
<tr>
<td>Zoran Popovic</td>
<td>32</td>
<td>Assistant Professor</td>
<td>FT</td>
<td>PhD</td>
<td>CMU, 1999</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Rajesh Rao</td>
<td>31</td>
<td>Assistant Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Rochester, 1998</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Walter L. Ruzzo</td>
<td>54</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Berkeley, 1978</td>
<td>0</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>David Salesin</td>
<td>40</td>
<td>Professor</td>
<td>PT</td>
<td>PhD</td>
<td>Stanford, 1991</td>
<td>1</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Steven Seitz</td>
<td>32</td>
<td>Assistant Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Wisconsin, 1997</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Rimli Sengupta</td>
<td>35</td>
<td>Research Ass’t Prof.</td>
<td>FT</td>
<td>PhD</td>
<td>Georgia Tech, 1995</td>
<td>0</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>Linda Shapiro</td>
<td>51</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Iowa, 1974</td>
<td>0</td>
<td>27</td>
<td>15</td>
</tr>
<tr>
<td>Lawrence Snyder</td>
<td>54</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>CMU, 1973</td>
<td>0</td>
<td>28</td>
<td>18</td>
</tr>
<tr>
<td>Dan Suciu</td>
<td>42</td>
<td>Assistant Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Pennsylvania, 1995</td>
<td>5</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>Steve Tanimoto</td>
<td>51</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Princeton, 1975</td>
<td>0</td>
<td>26</td>
<td>24</td>
</tr>
<tr>
<td>Martin Tompa</td>
<td>49</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Toronto, 1978</td>
<td>0</td>
<td>23</td>
<td>23</td>
</tr>
</tbody>
</table>

(continues on next page)
Table A.4. Faculty Analysis
(Computer Engineering – cont’d)

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Rank</th>
<th>FT or PT</th>
<th>Highest Degree</th>
<th>Institution from which Highest Degree Earned &amp; Year</th>
<th>Years of Experience</th>
<th>Professional Registration (Indicate State)</th>
<th>Level of Activity (high, med, low, none) in:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Govt./Industry Practice</td>
<td>Total Faculty</td>
<td>This Institution</td>
</tr>
<tr>
<td>Dan Weld</td>
<td>40</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>MIT, 1988</td>
<td>0</td>
<td>13</td>
<td>13</td>
</tr>
<tr>
<td>David Wetherall</td>
<td>34</td>
<td>Assistant Professor</td>
<td>FT</td>
<td>PhD</td>
<td>MIT, 1998</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>John Zahorjan</td>
<td>46</td>
<td>Professor</td>
<td>FT</td>
<td>PhD</td>
<td>Toronto, 1980</td>
<td>0</td>
<td>21</td>
<td>21</td>
</tr>
</tbody>
</table>

Note that above lists all faculty in the department. It is not possible to separate faculty into Computer Science program and Computer Engineering program faculty. Approximately 50% of our current undergraduates are expected to graduate with a Computer Engineering degre
Table A.5. Support Expenditures  
(Computer Engineering)

<table>
<thead>
<tr>
<th>Fiscal year</th>
<th>1998-99 (prior to previous year)</th>
<th>1999-00 (previous year)</th>
<th>2000-01 (current year)</th>
<th>2001-02 (year of visit)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Expenditure Category</strong></td>
<td><strong>Operations (1)</strong> not including staff</td>
<td><strong>Travel (2)</strong></td>
<td><strong>Equipment (3)</strong></td>
<td><strong>Graduate Teaching Assistants</strong></td>
</tr>
<tr>
<td></td>
<td>675,384</td>
<td>78,204</td>
<td>130,921</td>
<td>365,275</td>
</tr>
<tr>
<td></td>
<td>1,114,148</td>
<td>107,877</td>
<td>788,023</td>
<td>518,549</td>
</tr>
<tr>
<td></td>
<td>1,206,383</td>
<td>100,136</td>
<td>1,018,620</td>
<td>385,003</td>
</tr>
<tr>
<td></td>
<td>1,254,638</td>
<td>104,141</td>
<td>1,059,365</td>
<td>400,403</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>1,487,993</td>
<td>2,866,340</td>
<td>2,941,940</td>
<td>3,059,618</td>
</tr>
</tbody>
</table>

All amounts are for the entire department. Computer Engineering students account for approximately 50% of our students in 2001-2 and 33% in 1998-1999 with linear scaling in between.

Comments:
Graduate Teaching Assistants and Part-time Assistance excludes fringe benefits.
Above amounts include Research Support allocations.

Sources:
Funding types include General Operating Funds and Designated operating funds, 74- and 75-; Actual exp operations, travel, equipment institutional funds, GTAs, part-time; Fray Database; Equipment (b): Office of Institutional Studies financial database; 2000-01 includes actuals thru month 8 and a 33% factor for the remainder of the year; 2001-02 data is projected @ 104% of 2000-01.
Appendix I

B. Course Syllabi

Core courses:
- CSE 142: Computer Programming for Engineers and Scientists I
- CSE 143: Computer Programming for Engineers and Scientists II
- CSE 321: Discrete Structures
- CSE 322: Introduction to Formal Models in Computer Science
- CSE 326: Data Structures
- CSE 341: Programming Languages
- CSE 370: Introduction to Digital Design
- CSE 378: Machine Organization and Assembly Language Programming

Senior courses:
- CSE 401: Introduction to Compiler Construction
- CSE 403: Software Engineering
- CSE 421: Introduction to Algorithms
- CSE 431: Introduction to Theory of Computation
- CSE 444: Introduction to Database Systems
- CSE 451: Introduction to Operating Systems
- CSE 457: Computer Graphics
- CSE 458: Computer Animation
- CSE 461: Introduction to Computer-Communication Networks
- CSE 466: Software for Embedded Systems
- CSE 467: Advanced Digital Design
- CSE 471: Computer Design and Organization
- CSE 473: Introduction to Artificial Intelligence
- CSE 490ab: Computers, Ethics, and Society
- CSE 490ca: Computer Animation
- CSE 490i: Advanced Internet Systems
- CSE 490kt: Computer Telephony

Capstone design courses:
- CSE 468: Very Large Scale Integration
- CSE 476: Embedded System Design
- CSE 477: Digital System Design
- CSE 481: Capstone Software Design

Independent study and/or research:
- CSE 498: Senior Project (Honors: 498H)
- CSE 499: Reading and Research

Non-majors courses (included only for completeness):
- CSE 100: Fluency in Information Technology
- CSE 373: Data Structures and Algorithms
- CSE 410: Computer Systems
- CSE 413: Programming Languages and Their Implementation
- CSE 415: Introduction to Artificial Intelligence
- CSE 417: Algorithms and Computational Complexity
<table>
<thead>
<tr>
<th>Course</th>
<th>1.a</th>
<th>1.b</th>
<th>1.c</th>
<th>1.d</th>
<th>2.a</th>
<th>2.b</th>
<th>2.c</th>
<th>2.d</th>
<th>3.a</th>
<th>3.b</th>
<th>3.c</th>
<th>3.d</th>
<th>4.a</th>
<th>4.b</th>
<th>4.c</th>
<th>4.d</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Core courses</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 142: Computer Programming for Engineers and Scientists I</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 143: Computer Programming for Engineers and Scientists II</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 321: Discrete Structures</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 322: Introduction to Formal Models in Computer Science</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 326: Data Structures</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 341: Programming Languages</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 370: Introduction to Digital Design</td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 378: Machine Organization and Assembly Language Programming</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Senior courses</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 401: Introduction to Compiler Construction</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 403: Software Engineering</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 421: Introduction to Algorithms</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 431: Introduction to Theory of Computation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 444: Introduction to Database Systems</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 451: Introduction to Operating Systems</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 457: Computer Graphics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 458: Computer Animation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>CSE 461: Introduction to Computer-Communication Networks</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 466: Software for Embedded Systems</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 467: Advanced Digital Design</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 471: Computer Design and Organization</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 473: Introduction to Artificial Intelligence</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 490: Computers, Ethics, and Society</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 490ca: Computer Animation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 490i: Advanced Internet Systems</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>CSE 490kt: Computer Telephony</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td><strong>Capstone design courses</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 468: Very Large Scale Integration</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 476: Embedded System Design</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 477: Digital System Design</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 481: Capstone Software Design</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Ind. Project/Research</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 498: Senior Project (Honors: 498H)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CSE 499: Reading and Research</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Core courses

- CSE 142: Computer Programming for Engineers and Scientists I
- CSE 143: Computer Programming for Engineers and Scientists II
- CSE 321: Discrete Structures
- CSE 322: Introduction to Formal Models in Computer Science
- CSE 326: Data Structures
- CSE 341: Programming Languages
- CSE 370: Introduction to Digital Design
- CSE 378: Machine Organization and Assembly Language Programming
CSE 142: Computer Programming for Engineers and Scientists I (4)

Catalog description:

Basic programming-in-the-small abilities and concepts. Highlights include procedural and functional abstraction with simple built-in data type manipulation. Basic abilities of writing, executing, and debugging programs. Not available for credit to students who have completed CSE 210 or ENGR 141.

Prerequisites: none

Textbook(s) and/or other required material:


Course objectives:

Develop basic programming and problem solving skills in a procedural language

Topics covered:

Basic concepts of computer structure and program execution
Variables, types, expressions, and assignment
Input/output: console and monitor, text files
Conditional execution (if, switch)
Iteration (while, for)
Loop development and design patterns
Functions, including value and reference (pointer) parameters
Program organization and libraries
Use of standard C and local libraries
Analyzing and depicting function relationships
Functional decomposition of problems
Arrays (1- and 2-D)
Sequential and binary search
Sorting (insertion or selection sort)
Structures
User-defined types combining arrays and structs
Strings
Introduction to recursion
Graphics concepts
Concepts of event-driven programming
Programming style
Experience with team or group work

Class/laboratory schedule:

3 50-minute lectures per week
1 50-minute quiz section per week
Sizable projects to be completed outside of class
CSE 143: Computer Programming for Engineers and Scientists II (5)

Catalog description:

Continuation of 142. Concepts of modularity and encapsulation, focusing on modules and abstract data types. Covers some basic data structures. Not available for credit to students who have completed CSE 211.

Prerequisites: none

Textbook(s) and/or other required material:


Course objectives:

Develop intermediate programming and problem-solving skills, including introduction to object-oriented programming; fundamental data structures and algorithms; and experience working in teams with programs of considerable complexity.

Topics covered:

Introduction to C++ (in contrast of C)
Abstract data types
Dynamic storage allocation and pointers
C++ classes: as a technique for encapsulation and abstraction
C++ classes: technical apparatus, including constructors, destructors, assignment, etc.
Classes and objects, inheritance, and dynamic dispatch
Basics of object-oriented design
Recursion
Linked lists
Basic container types: lists, stacks, and queues
Program efficiency, big-O notation
Searching and sorting (quicksort, mergesort)
Trees, particularly binary and binary search trees
Survey of hashing and other container types
Modeling and simulation concepts
Principles of engineering robust and extensible software (documentation, testing, modularity, reusability, etc.)
Experience working in a team or group

Class/laboratory schedule:

3 50-minute lectures per week
2 50-minute quiz sections per week
Sizable projects to be completed outside of class
CSE 321: Discrete Structures (4)

Catalog description:

Fundamentals of set theory, graph theory, enumeration, and algebraic structures, with applications in computing.

Prerequisites: CSE 143; either MATH 126, MATH 129, or MATH 136.

Textbook(s) and/or other required material:


Course objectives:

Provide students with the definitions and basic tools for reasoning about discrete mathematical objects useful for computer science and engineering.

Topics covered:

propositional logic and predicate logic
set theory, functions
the integers: divisibility, modular arithmetic, primality, etc.
methods of proof and formal reasoning
mathematical induction
recursive definitions
counting
discrete probability theory
binary relations
graph theory and graph algorithms
applications, including RSA

Class/laboratory schedule:

Meetings: 3 1-hour lectures per week, 1 recitation section hour per week
Assignments: weekly written assignments
Midterm plus final
CSE 322: Introduction to Formal Models in Computer Science (3)

Catalog description:

Finite automata and regular expressions; context-free grammars and pushdown automata; nondeterminism; Turing machines and the halting problem. Emphasis on understanding models and their applications and on rigorous use of basic techniques of analysis. Induction proofs, simulation, diagonalization, and reduction arguments.

Prerequisites: CSE 321.

Textbook(s) and/or other required material:


Course objectives:

Teach students the basics of three formal models of computation: finite automata, context-free languages, and Turing machines with special emphasis on methods for manipulating and reasoning about them.
Give students an idea of the value and applicability of the formal models.
Give sufficient grounding in automata and formal languages to prepare for parsing in the compilers class.

Topics covered:

- strings and languages, concatenation, powers, and reversal of strings
- operations on languages: union, concatenation and star
- regular languages, regular expressions
- countability of the regular languages, countability and uncountability, (Cantor’s proof)
- context-free grammars
- examples from real programming languages such as C++
- derivation trees, leftmost derivations
- ambiguous grammars, inherently ambiguous context-free grammars
- non-context-free languages w/o proof
- closure Properties - union, concatenation, star
- decision problems - emptiness, finiteness, equivalence
- parsing methods, top-down/bottom-up parsing, deterministic/non-deterministic parsing
- finite automata, DFAs, NFAs
- subset construction
- Kleene's theorem, closure properties - emptiness, finiteness, equivalence
- decision problems
- cross product construction
- nonregular languages

Class/laboratory schedule:

3 lecture hours per week. Weekly written assignments. Tests.
CSE 326: Data Structures (4)

Catalog description:


Prerequisites: CSE 321.

Textbook(s) and/or other required material:

Data Structures and Algorithm Analysis in C++, Mark A. Weiss, Benjamin Cummings Publishing Co., 1999

Course objectives:

The objective of this class is to study the fundamental data structures and algorithms used in computer science.

Topics covered:

1. Introduction: Data Structures, Abstraction, Induction, Recursion
2. Algorithm Analysis
3. Lists, Stacks, and Queues
4. Trees, Search Trees: Binary, AVL, Splay, B-tree
5. Hashing
6. Priority Queues: Heaps
7. Sorting: Mergesort, Quicksort, Decision
8. Disjoint Sets: Union-Find
9. Graph Algorithms

Class/laboratory schedule:

3 hours per week of lecture
1 hour per week of discussion section
CSE 341: Programming Languages (4)

Catalog description:

Basic concepts of programming languages, including abstraction mechanisms, types, and scoping. Detailed study of several different programming paradigms, such as functional, object-oriented, and logic programming. No credit if CSE 413 has been taken.

Prerequisites: CSE 143.

Textbook(s) and/or other required material:

Varies depending on instructor, but typically:

One of:
Abelson and Sussman, Structure and Interpretation of Computer Programs
Simon Thompson, Miranda: The Craft of Functional Programming
Jeff Ullman, Elements of ML Programming

And one of:
Timothy Budd, Understanding Object-Oriented Programming with Java
Mark Guzdial, Squeak: Object-Oriented Programming with Multimedia Applications

And sometimes:
Kim Marriott and Peter J. Stuckey, Programming with constraints: an introduction

Course objectives:

To learn fundamental programming language concepts and to gain practical experience with a diverse set of programming languages.

Topics covered:

Programming language design principles (readability, orthogonality, regularity, etc.)
Interpreters, browsers
Recursive functions over dynamic, recursive data structures (e.g. lists, trees)
Functional programming: side-effect-free programming; programming with first-class functions; lexical scoping of nested functions
Object-oriented programming: classes, objects, methods, instance variables; inheritance, overriding, overloading; object-oriented design principles
Type checking: static vs. dynamic typing; polymorphic static type systems; type inference; subtyping

Class/laboratory schedule:

3 lectures per week
1 section per week
CSE 370: Introduction to Digital Design (4)

Catalog description:
Introductory course in digital logic and its specification and simulation. Boolean algebra, combinational circuits including arithmetic circuits and regular structures, sequential circuits including finite-state-machines, use of programmable logic devices. Simulation and high-level specification techniques are emphasized.

Prerequisites: none

Textbook(s) and/or other required material:
Randy H. Katz, Contemporary Logic Design, Addison/Wesley, 1994

Course objectives:
Understanding of digital logic at the gate and switch level including both combinational and sequential logic elements.
Understanding of the clocking methodologies necessary to manage the flow of information and preservation of circuit state.
An appreciation for the specification methods used in designing digital logic and the basics of the compilation process that transforms these specifications into logic networks.
Facility with computer-aided design tools for digital logic design with programmable logic devices as the implementation technology.
To begin to appreciate the difference between hardware and software implementations of a function and the advantages and disadvantages of each.

Topics covered:
Introduction: 2 lectures
Combinational Logic: 5 lectures
Combinational Logic Implementation: 4 lectures
Combinational Logic Examples: 4 lectures
Hardware Description Languages: 2 lectures
Sequential Logic and Simple Registers: 4 lectures
Sequential Logic Implementation and Finite State Machines: 4 lectures
Sequential Logic Examples: 3 lectures
Computer Organization: 2 lectures

Class/laboratory schedule:
Meetings: 3 1-hour lectures per week, 1 recitation section hour per week
Assignments: weekly written assignments with a larger project as the last assignment
Laboratory: use of computer-aided design tools for aspects of almost every assignment
CSE 378: Machine Organization and Assembly Language Programming (4)

Catalog description:

Differences and similarities in machine organization; central processors; fundamentals of machine language and addressing; assembly language programming, including macros; operating system interfaces. No credit to students who have completed 410.

Prerequisites: CSE 143.

Textbook(s) and/or other required material:

D. Patterson and J. Hennessy:
Computer Organization & Design: The Hardware/Software Interface

Course objectives:

The purpose of this course is to give students a basic understanding of computer architecture and organization, and assembly language programming.

Topics covered:

machine organization, instruction sets, addressing modes, instruction encoding, subroutine handling, pipelining, microprogramming, memory systems, caches, I/O

Class/laboratory schedule:

3 lectures/week
1 discussion section/week
Senior courses

- CSE 401: Introduction to Compiler Construction
- CSE 403: Software Engineering
- CSE 421: Introduction to Algorithms
- CSE 431: Introduction to Theory of Computation
- CSE 444: Introduction to Database Systems
- CSE 451: Introduction to Operating Systems
- CSE 457: Computer Graphics
- CSE 458: Computer Animation
- CSE 461: Introduction to Computer-Communication Networks
- CSE 466: Software for Embedded Systems
- CSE 467: Advanced Digital Design
- CSE 471: Computer Design and Organization
- CSE 473: Introduction to Artificial Intelligence
- CSE 490ab: Computers, Ethics, and Society
- CSE 490ca: Computer Animation
- CSE 490i: Advanced Internet Systems
- CSE 490kt: Computer Telephony
CSE 401: Introduction to Compiler Construction (3)

Catalog description:

Fundamentals of compilers and interpreters: symbol tables; lexical analysis, syntax analysis, semantic analysis, code generation, and optimizations for general purpose programming languages. No credit to students who have taken 413.

Prerequisites: CSE 322; CSE 326; CSE 341; CSE 378.

Textbook(s) and/or other required material:

"Compilers: Principles, Techniques, and Tools" by Aho, Sethi, and Ullman

Course objectives:

Learn principles and practice of language implementations.
Understand tradeoffs between run-time and compile-time processing.
Understand tradeoffs between language features, run-time efficiency, and implementation difficulty.
Gain experience working with large systems software, object-oriented design, and C++.

Topics covered:

Organization of Compilers and Interpreters
Lexical Analysis
Syntactic Analysis
Semantic Analysis
Interpretation
Run-Time Storage Layout
Code Generation
Optimization

Class/laboratory schedule:

Lecture, 3 hours per week.
Laboratory, as needed to complete course project.
CSE 403: Software Engineering (4)

Catalog description:

Fundamentals of software engineering using a group project as the basic vehicle. Topics covered include the software crisis, managing complexity, requirements specification, architectural and detailed design, testing and analysis, software process, and tools and environments.

Prerequisites: CSE 321; CSE 341; CSE 378; recommended: CSE 401; CSE 451.

Textbook(s) and/or other required material:

FUNDAMENTALS OF SOFTWARE ENGINEERING
Carlo GHEZZI, Mehdi JAZAYERI, Dino MANDRIOLI
Publisher: Prentice Hall - Englewood Cliffs, NJ

Assorted papers from the literature

Course objectives:

A central objective of the course is to have students develop a deep understanding of the distinctions between software engineering and programming. In addition, the students understand the software lifecycle, increase their knowledge of classic and modern software engineering techniques, and develop concrete experience in turning ill-formed concepts into products working with a team.

Topics covered:

Topics covered include the software crisis, managing complexity, requirements specification, architectural and detailed design, testing and analysis, software process, and tools and environments.

Class/laboratory schedule:

There are three lectures each week (50 minutes each) and one section with the teaching assistant. The lectures cover general software engineering material, material pertinent specifically to the quarter-long project, and usually includes guest lectures from local industry during the last week or so of the quarter (when the students are completely focused on their project). The sections are used for reviews of the projects and for presentation by the TA of material on specific tools and technologies needed to develop the project.
CSE 421: Introduction to Algorithms (3)

Catalog description:

Techniques for design of efficient algorithms. Methods for showing lower bounds on computational complexity. Particular algorithms for sorting, searching, set manipulation, arithmetic, graph problems, pattern matching.

Prerequisites: CSE 322; CSE 326.

Textbook(s) and/or other required material:

Varies. Cormen, Leiserson, Rivest "Intro to Algorithms" or Manber, same title, are commonly used.

Course objectives:

Learn basic techniques for design and analysis of algorithms, including correctness proofs. Learn a number of important basic algorithms. NP-complete and other intractable problems.

Topics covered:

Main Techniques:
Design: Induction, Graph search, Divide and Conquer, Greedy, Dynamic Programming, Branch and Bound.
Analysis: Asymptotic Analysis, Recurrences.
Intractability: Reduction.

Typical Algorithm coverage: depth- and breadth-first search, bi- and/or strongly connected components, shortest paths, min spanning trees, transitive closure, flows and matchings, Strassen's method, FFT, knapsack, edit distance/string matching


Class/laboratory schedule:

3 50 minute lectures per week.
CSE 431: Introduction to Theory of Computation (3)

Catalog description:
Models of computation, computable and noncomputable functions, space and time complexity, tractable and intractable functions.

Prerequisites: CSE 322.

Textbook(s) and/or other required material:

Course objectives:
Develop the concepts and skills necessary to be able to evaluate the computability and complexity of practical computational problems.

Topics covered:
Turing machines (deterministic, nondeterministic, multitape)
Church-Turing Thesis
Decidability and undecidability, diagonalization, and reducibility
Halting problem, Post correspondence problem, Rice’s Theorem, and other undecidability results
Time and space complexity
P vs. NP, NP-completeness, Cook’s Theorem, and other NP-complete problems
L vs. NL, NL-completeness, Savitch’s Theorem, Immerman/Szelepcsényi Theorem

Class/laboratory schedule:
Three 50-minute lectures per week.
CSE 444: Introduction to Database Systems (3)

Catalog description:
Fundamental concepts, system organization, and implementation of database systems. Relational, hierarchical, and network data models; file organizations and data structures; query languages; query optimization; database design; concurrency control; security; issues involving distributed database systems.

Prerequisites: CSE 326.

Textbook(s) and/or other required material:
Database Management Systems (Ramakrishnan and Gehrke)

A first course on database systems (Ullman, Widom and Garcia Molina)

Course objectives:
Provide a basic introduction to the concepts of database system design and implementation, including: data models, conceptual design, query languages, system components, data storage, query optimization and transaction processing.

Topics covered:
data models, conceptual design, query languages, system components, data storage, query optimization and transaction processing.

Class/laboratory schedule:
3 lectures a week
CSE 451: Introduction to Operating Systems (4)

Catalog description:
Principles of operating systems. Process management, memory management, auxiliary storage management, resource allocation. No credit to students who have completed 410 or EE 474.

Prerequisites: CSE 326; CSE 378.

Textbook(s) and/or other required material:
Silberschatz and Galvin, Operating Systems Concepts

Course objectives:
Give students a working knowledge of operating systems principles, design issues, algorithms and data structures.
Build programming experience through a large operating systems project (using Nachos).

Topics covered:
operating system structure, processes, threads, synchronization, scheduling, deadlock, virtual memory, secondary storage management, distributed systems, file systems

Class/laboratory schedule:
Class: 3 days per week, 1 hour
Sections: 1 day per week, 1 hour
Labs: as needed for project and assignments
CSE 457: Computer Graphics (4)

Catalog description:

Introduction to computer image synthesis and interactive computer graphics applications. Topics include computer graphics hardware, color image display, event-driven programming, line drawing, polygon scan conversion, texture mapping, image morphing, image compositing, curves and surfaces, hidden surface algorithms, local illumination models, ray tracing, and photorealistic image synthesis.

Prerequisites: CSE 326.

Textbook(s) and/or other required material:


Course objectives:

Introduction to computer image synthesis and interactive computer graphics applications. Learn fundamentals of 2D and 3D computer graphics modeling, rendering, and animation through homeworks and projects.

Topics covered:

Topics include computer graphics hardware, color image display, event-driven programming, line drawing, polygon scan conversion, texture mapping, image morphing, image compositing, curves and surfaces, hidden surface algorithms, local illumination models, ray tracing, and photorealistic image synthesis.

Class/laboratory schedule:

Three hours of lecture
One hour of lab help session (4 times per quarter)
Projects to be completed, two weeks each, in lab or at home
CSE 458: Computer Animation (5)

Catalog description:

Introduction to basic principles of computer generated animation. Focus on the modeling and lighting of animated characters. Students from Art, CSE, and Music team up on projects to be built on commercially-available modeling and lighting packages.

Prerequisites: either CSE 457 or ART 380 or MUSIC 403.

Textbook(s) and/or other required material:

Michael O'Rourke, Three Dimensional Computer Animation

Course objectives:

To introduce students to the fundamentals of 3D modeling, shading, lighting, animating and rigging characters for three-dimensional computer generated environments.

To understand the complex technical and aesthetic components of the design of animation.

Topics covered:

Modeling digital objects that one can find reference for in the real world.
Modeling hard surface and characters for 3D animated digital environments
Shading objects.
Lighting concepts from the real world applied to digital 3D environments
Character Animation Principles
Character Animation Projects
Theory and fundamentals of character rigging for computer animation

Class/laboratory schedule:

Tuesday and Thursday 10:30 - 11:20 Lecture
Lab Hours in LA2
CSE 461: Introduction to Computer-Communication Networks (4)

Catalog description:


Prerequisites: CSE 143.

Textbook(s) and/or other required material:


Course objectives:

To provide students with an understanding of how to construct large-scale computer networks. This includes an appreciation of the fundamental problems that arise in building networks, the design principles that are of proven value, and the common implementation technologies that are in use today.

Topics covered:

This course introduces the basics of networking, ranging from transmitting bits over wires to the Web and distributed computing. We focus on the internetworking ground in-between these two extremes. We will cover framing, error correction, packet and circuit switching, multi-access protocols (Ethernet), queuing, addressing and forwarding (IP), distance vector and link state routing, reliable transport, congestion control (TCP), quality of service, multicast, and security.

Class/laboratory schedule:

Two 1 1/2 hour classes per week, laboratory available for projects.
CSE 466: Software for Embedded Systems (4)

Catalog description:

Software issues in the design of embedded systems. Microcontroller architectures and peripherals, embedded operating systems and device drivers, compilers and debuggers, timer and interrupt systems, interfacing of devices, communications and networking. Emphasis on practical application of development platforms.

Prerequisites: CSE 326; CSE 370; CSE 378.

Textbook(s) and/or other required material:

In inaugural course, we used An Embedded Software Primer, David E. Simon, Addison Wesley. But, will likely not require this book in the future. Extensive use of technical datasheets and on-line documentation of devices and tools.

Course objectives:

Hand's on experience in the design, development, and analysis of embedded software systems. Including basics of embedded hardware

Topics covered:

Embedded Hardware
Peripheral Interfaces (Embedded Hardware, I/O, power, clocks, etc.)
The 8051 Embedded Processor, and Development Tools
Timers and Interrupts
Embedded Software Architectures and Multi-tasking
Embedded Operating Systems
Device Drivers (Linux style)
PDA Programming and Interfacing
Timing analysis
Safety, and Society
Communications and Embedded Networking

Class/laboratory schedule:

3 hours lecture
3 hours lab
CSE 467: Advanced Digital Design (4)

Catalog description:

Advanced techniques in the design of digital systems. Hardware description languages, combinational and sequential logic synthesis and optimization methods, partitioning, mapping to regular structures. Emphasis on reconfigurable logic as an implementation medium. Memory system design. Digital communication including serial/parallel and synchronous/asynchronous methods.

Prerequisites: CSE 326; CSE 370.

Textbook(s) and/or other required material:


Course objectives:

1. To learn how to design digital systems, from specification and simulation to construction and debugging.
2. To learn techniques and tools for programmable logic design
3. To learn how to use modern laboratory test equipment, including logic analyzers and oscilloscopes
4. To understand the limitations and difficulties in modern digital design, including wiring constraints, high-speed, etc.
5. To design, construct, test, and debug a moderate-scale digital circuit.

Topics covered:

1. Review of basic digital-logic design; Combinational logic; Structured logic implementation; Sequential logic; Finite-state machines
2. Overview of digital technology; Logic families; Reading and understanding data books; Interfacing; Fixed-function devices; TTL/CMOS; glue logic; RAM/ROM; Programmable devices; PROMs; PALs and PLDs; FPGAs; Integrated circuits
3. Electrical realities; Resistance, capacitance and inductance; Time constants; Decoupling and ground; Power dissipation and drops; Wire delays; Fanout and loading; Ringing, reflections, and terminations; Clock
4. Computer-aided design; Hardware description languages; Logic compilation; Two-level and multi-level logic synthesis; Technology-independent optimization; Technology mapping; Sequential-logic synthesis; Tools for mapping to PLDs and FPGAs;
5. Laboratory realities; Logic analyzer and oscilloscope basics; Repetitive versus single-shot triggering; Timing, state, capture, bandwidth; Glitches and transient events
6. System-level components; Static, dynamic, and nonvolatile memories; Memory controllers and timing; Digital communication; Serial and parallel protocols; Synchronous vs. asynchronous data communication; Busses; Arbitration schemes
7. Technology; MOSFETs; FPGAs; Integrated circuits; Circuit boards; High-speed circuits; controlling impedances

Class/laboratory schedule:

3 hours lecture and 3 hours lab per week
CSE 471: Computer Design and Organization (4)

Catalog description:

CPU instruction addressing models, CPU structure and functions, computer arithmetic and logic unit, register transfer level design, hardware and microprogram control, memory hierarchy design and organization, I/O and system components interconnection. Laboratory project involves design and simulation of an instruction set processor.

Prerequisites: CSE 370; CSE 378.

Textbook(s) and/or other required material:


Course objectives:

Teach the design and architecture of major components of the structure of the central processing unit and memory hierarchy of modern microprocessor systems. Use a cycle by cycle simulator to illustrate logic complexities.

Topics covered:

Pipelining
Branch prediction
Exceptions
Examples of CPU's in modern microprocessors
Exploiting Instruction Level Parallelism
Scoreboard and Tomasulo's algorithm
Superscalars
Caches and cache assists
Hardware assists for paging systems
TLB's
Symmetric MultiProcessors
Cache coherence
Synchronization

Class/laboratory schedule:

3 lectures a week (50 mns)
CSE 490ab: Computers, Ethics, and Society (1)

Catalog description:

none (special topics course not yet listed)

Prerequisites: none

Course objectives:

To expose students to a wide range of topics in computers, ethics, and societal impacts, and to facilitate discussion, debate, and critical thinking on these topics.

Topics covered:

professional ethics
impacts of computing on the economy
computers and education
privacy and databases
universal and differential access
impacts on social capital
free speech online
the impact of the internet on the political process
computer crime; hacking; attacks
software patents and copyrights
commercialization within a university environment
antitrust and monopoly law and the software industry

Class/laboratory schedule:

1 hour seminar, once a week, often with an outside speaker
CSE 490ca: Computer Animation (5)

Catalog description:
none (special topics course not yet listed)

Prerequisites: none

Textbook(s) and/or other required material:
None

Course objectives:
To learn the production pipeline for short animated films
To learn how to design and implement a storyreel and animatic for a short film
To learn how to produce a brief professionally
produced collaborative work with the emphasis on applying the knowledge
gained to create a unique product.
To work with other students on a team.
The design and implement a final project that
incorporates and displays both aesthetic and technical excellence.

Topics covered:
Production Planning for Animated Short Films
Storyboard and Layout Techniques
Character Modeling
Character Setup
Character Animation
Acting and Performance for Animators
Designing shapes for Facial Expression
Designing and Building Surface Shaders
Compositing and Post Production
Animation and Technical Direction
Digital Special Effects

Class/laboratory schedule:
Tuesday and Thursday 10 - 11:50 Lecture
Lab in LA2
CSE 490i: Advanced Internet Systems (5)

Catalog description:

none (special topics course not yet listed)


Prerequisites: none

Textbook(s) and/or other required material:

None

Course objectives:

Introduction to Internet technologies
Design and implementation of a substantial application in a team environment.

Topics covered:

Introduction
Networking Fundamentals
HTTP and Server Basics
Spiders and Search Engine Architecture
Learning and Classification Algorithms
Search Engines and Information Retrieval
Clustering
Basics of Databases
Database Implementation
Information Aggregation and Its Applications
MetaCrawler
XML
Consistency and Availability for Large-scale Applications
E-Commerce
Cryptographic Primitives
Distributed Application Network: Where the Internet and Distributed Computing Converge
CSE 490kt: Computer Telephony (5)

Catalog description:

none (special topics course not yet listed)

Prerequisites: none

Textbook(s) and/or other required material:

None

Course objectives:

Introduction to MS Automation via Computer Telephony. ActiveX Components illustrate the use of databases in CT, audio recordings, internet transmission of sounds and images, voice recognition and text-to-speech, high-level CT protocols, low-level CT controls including call-analysis and routing via SC-bus.

Topics covered:

MS Automation; ADO database connections; audio recordings and playback; FSM implementation of home automation; WinSock internet connections; DirectX recording, mixing and playback; voice recognition and text-to-speech; flowchart control of telephony; low-level control of telephony. A substantial course project combining 2 or more of the above topics.
Capstone design courses:

- CSE 468: Very Large Scale Integration
- CSE 476: Embedded System Design
- CSE 477: Digital System Design
- CSE 481: Capstone Software Design
CSE 468: Very Large Scale Integration (5)

Catalog description:

Introduction to CMOS technology and circuit design; implementation of combinational and sequential logic; VLSI design methodologies; CAD tools for layout, simulation, and validation. Students design a VLSI chip using modern CAD tools.

Prerequisites: CSE 370.

Textbook(s) and/or other required material:


Course objectives:

1. To learn the static and dynamic behavior of physical (i.e. silicon) logic gates, including transistor sizing, time constants, loading and delays, fan-in and fan-out, interconnect, etc.
2. To learn the behavior of physical (i.e. silicon) sequential-logic circuits, including clocking (single and two-phase), clock skew, pipelining, memories and memory access, interconnect, etc.
3. To learn the tools of custom IC design, including schematic entry, simulation (analog, static timing, dynamic timing), layout, DRC, and LVS. Basics of IC fabrication.

Topics covered:

1. Introduction and overview of modern IC design
2. Transistor basics; Fermi levels and band diagrams; pn-junction diode; MOS transistor Sub- and above-threshold MOSFET operation; Static and dynamic MOSFET behavior
3. CAD tools for physical IC design; OrCAD Capture for schematic entry; Spice for simulation and analysis; Tanner Tools LEdit for physical layout; Tanner Tools DRC and LVS to verify design; UW Parasite to back-annotate simulation
4. Inverters and pass transistors; Static and dynamic behavior; Noise margin, fan-in and fan-out; Time constants, logic levels, drive
5. Static CMOS logic; Basic logic gates; Static and dynamic behavior; Time constants (RC), loading, fan-in and fan-out, delays; Transistor sizing
6. Dynamic CMOS logic; Domino, Zipper, CVSL logic; Dynamic & cascading behavior
7. Sequential circuits; Static latches and flip-flops; Dynamic latches and flip-flops; Registers; Merged logic; Clocking (1 and 2-phase); Clock skew, delays, race conditions
8. Memories; Static RAM; Sense amplifiers; Address decoders; Dynamic RAM; Nonvolatile memories
9. Arithmetic circuits; Adders; Multipliers
10. Chip design; Floorplanning; Power bussing, power and ground bounce; Interconnect and delays; Clocking and PLLs; Yield; Design margin; Pads
11. Advanced topics; Technology scaling; Low-power design; Bipolar transistors, differential CML; High-speed circuits, controlled impedance lines

Class/laboratory schedule:

3 lectures and 3 hour laboratory per week. Additonal laboratory time for projects.
CSE 476: Embedded System Design (5)

Catalog description:

System building course to provide students with a complete experience in embedded system design. Students will design, simulate, construct, debug, and document a substantial project of their choosing. Lectures will focus on case studies and emerging components and platforms.

Prerequisites: CSE 451; CSE 466.

Textbook(s) and/or other required material:

Required material, including datasheets and documentation of components and software, is handed out in class.

Course objectives:

To serve as a capstone design course to tie together the computer engineering curriculum via the design of a complete embedded system involving multiple communicating components.
To gain appreciation for the software issues in embedded system code.
Familiarity with basic communication methods.
To experience the development of a complete product from design to implementation and debugging.
To present design goals and decisions as well as implementation results in both verbal presentation and written documentation.
To work toward a common goal in a team environment.

Topics covered:

Introduction; Embedded Systems
Product design; Product development process
Team-based Design
Project definition; Decomposition into hardware and software components
Evaluation of needs; Assessment of team members’ skills
Experimental design to resolve unknowns; Design reuse; Project Evolution
Design for maintainability; Design for upgradability; Modularity in Design
Research Directions; Architecture innovations
Operating systems; Networking
Application domains; User interfaces
Standards: Physical layer; Protocol; Packaging; Standardization process; evolution
Case Studies; Past capstone design projects; Industry case studies

Class/laboratory schedule:

3 lectures and 3 hour laboratory per week. Lecture time for the first half of the quarter is spent covering the syllabus topics. In the last half of the quarter, lecture time is spent on design reviews, project presentations, special topics and guest lecturers. Students spend most of their out-of-class time in the lab designing, constructing and debugging their projects.
CSE 477: Digital System Design (5)

Catalog description:

Students use laboratory to design, simulate, construct, and debug a substantial project that includes hardware, software, and communication components. Lectures focus on use of embedded processors in digital system design and interfacing techniques. Writing and debugging of real-time reactive software emphasized.

Prerequisites: CSE 378; CSE 467.

Textbook(s) and/or other required material:

Required material, including datasheets and documentation of components and software, is handed out in class.

Course objectives:

To serve as a capstone design course to tie together the computer engineering curriculum via the design of a complete embedded system involving hardware, software, and communication components.
Understanding of basic microcontrollers and their use in embedded system design.
Appreciation for the software issues in embedded system code.
Familiarity with basic serial and parallel communication methods.
Experience the design and development of a complete product using hardware and software, from design to implementation and debugging.
Ability to present design goals and decisions as well as implementation results in both verbal presentation and written documentation.
Ability to work toward a common goal in a team environment.

Topics covered:

Introduction to embedded systems, Microprocessors and microcontrollers
Review of basic computer organization: Address/data bus; Memories; I/O ports
Timing subsystems
Interfacing techniques: Basic I/O ports; Interactions involving time
Polling, Interrupts and interrupt handling
Communication: Serial; Parallel; Basic wireless schemes; Error correction; Flow control
Hardware Design; Partitioning Software and Hardware
Interface design and implementation, Interface support devices
Design experiences: Case studies; Industry perspectives
Research directions

Class/laboratory schedule:

3 lectures and 3 hour laboratory per week. Lecture time for the first half of the quarter is spent covering the syllabus topics. In the last half of the quarter, lecture time is spent on design reviews, project presentations, special topics and guest lecturers. Students spend most of their out-of-class time in the lab designing, constructing and debugging their projects.
CSE 481: Capstone Software Design (5)

Catalog description:

Students work in teams to design and implement a software project involving multiple areas of the CSE curriculum. Emphasis is placed on the development process itself, rather than on the product.

Prerequisites: CSE 326; CSE 341; CSE 378 and substantial programming experience, such as in CSE 451 or 457.

Textbook(s) and/or other required material:

Use of development software, typically: Visual Studio, Visual Source Safe, a commercial game engine (e.g., from LithTech), 3D modelling and animation tools (e.g., 3D Studio Max).

Course objectives:

To learn about the software design process through hands-on development of a software product.
To experience working in larger teams than you have had to deal with previously in our curriculum.
To experience building sophisticated applications by making use of real-world tools (e.g., the DirectX SDK), rather than trying to build everything from scratch.
To gain experience dealing with the performance demands of high-performance, real-time, distributed applications.
To have some fun (by building a game).
To develop a portfolio documenting your efforts that could be useful in looking for a job.

Topics covered:

Software development in teams; basics of games development. Most student time is spent in the development process, and performing critiques of it.

Class/laboratory schedule:

3 hours of lectures per week by the instructor and guest lecturers from industry. However, lectures usually are given only during the first two to three weeks. After that, lectures are replaced with meetings with individual teams.
Independent study and/or research

- CSE 498: Senior Project (Honors: 498H)
- CSE 499: Reading and Research
CSE 498: Senior Project (1-6)  
CSE 498H Honors (9)

Catalog description:

A report (and perhaps demonstration) describing a development, survey, or small research project in computer science or an application to another field. Objectives are: (1) integrating material from several courses, (2) introducing the professional literature, (3) gaining experience in writing a technical document, and (4) showing evidence of independent work. Work normally extends over more than one quarter, for a maximum of 6 credits for 498; 9 credits are required for 498H.

Prerequisites: none
CSE 499: Reading and Research (1-24)

Catalog description:

Available in special situations for advanced computer science majors to do reading and research in field, subject to approval of undergraduate advisor and CSE faculty member. Free elective, but does not replace core course or computer science elective. Credit/no credit only.

Prerequisites: none
Non-majors courses
(included only for completeness)

- CSE 100: Fluency in Information Technology
- CSE 373: Data Structures and Algorithms
- CSE 410: Computer Systems
- CSE 413: Programming Languages and Their Implementation
- CSE 415: Introduction to Artificial Intelligence
- CSE 417: Algorithms and Computational Complexity
CSE 100: Fluency in Information Technology (5)

Catalog description:

Introduces the skills, concepts, and capabilities necessary to effectively use information technology. Includes logical reasoning, managing complexity through the operation of computers and networks, and contemporary applications such as email and word processing. Not available for credit to students who have completed CSE 142 or ENGR 142.

Prerequisites: none

Textbook(s) and/or other required material:

Fluency With Information Technology (Draft) by Snyder
Visual Basic 6, Eliason and Malarkey, Que, 1999

Course objectives:

To enable students to be "Fluent with Information Technology" as defined by the National Research Council's report, "Being Fluent with IT".

This is the first course of its kind.

Topics covered:

The NRC Report itemizes 10 skills, 10 concepts (including algorithmic thinking and programming) and 10 capabilities that would constitute FITness. This class covers 9 skills, 10 concepts and 8 capabilities.

Class/laboratory schedule:

MWF 1 hour lecture
TTh 1 hour Closed Lab
Catalog description:

Fundamental algorithms and data structures for implementation. Techniques for solving problems by programming. Linked lists, stacks, queues, directed graphs. Trees: representations, traversals. Searching (hashing, binary search trees, multiway trees). Garbage collection, memory management. Internal and external sorting. No credit to students who have completed 326, 374, or EE 374.

Prerequisites: CSE 143.

Textbook(s) and/or other required material:

Data Structures and Algorithm Analysis by Mark Allen Weiss

Course objectives:

This class covers a fairly broad sampling of fundamental data structures and algorithms used in everyday computer programming. The strengths, weaknesses, and practical uses of each new concept will be examined. The intent of the class is to equip students with a solid set of tools and skills necessary for choosing or designing appropriate data structures and algorithms used in programming.

Topics covered:

Algorithm Analysis
Linked Lists
Stacks & Queues
Trees
Balanced Trees
Hashing
Heaps
Sorting Algorithms
Graphs
Search Algorithms
Minimum Spanning Trees
Shortest Paths
Algorithm Classification

Class/laboratory schedule:

MWF 1 hour lecture
CSE 410: Computer Systems (3)

Catalog description:

Structure and components of hardware and software systems. Machine organization, including central processor and input-output architectures; assembly language programming; operating systems, including process, storage, and file management. No credit to students who have completed 378 or 451.

Prerequisites: CSE 373.

Textbook(s) and/or other required material:


Course objectives:

To learn the concepts of computer architecture and operating systems. The first half of the course focuses on architecture, using the MIPS R2000 as a specific example. We study MIPS assembly language, implementation schemes, and memory hierarchy. The second half of the course focuses on operating systems, covering process, memory, and file management techniques.

Topics covered:

- Data representations
- Instruction sets and assembly language
- Pipelining
- Memory hierarchies and caches
- Operating systems overview
- Processes and threads
- Synchronization and communications
- Deadlock
- Scheduling
- Memory management and virtual memories
- File systems

Class/laboratory schedule:

3 hours of lectures per week.
CSE 413: Programming Languages and Their Implementation (3)

Catalog description:
none

Prerequisites: none

Textbook(s) and/or other required material:
Revised Report on the Algorithmic Language Scheme (R5RS)
Course handouts and lecture notes

Course objectives:
Expose students to different ways of thinking about computation and programming, and gain insight into how programming languages are implemented.

Topics covered:
Functional programming using Scheme
Object-Oriented programming using Java
Implementation of programming languages, including introduction to formal grammars and parsing, introduction to machine organization, and implementation of a recursive-descent compiler for a small subset of C, generating x86 assembly language code (term project)

Class/laboratory schedule:
3 50-minute lectures per week
CSE 415: Introduction to Artificial Intelligence (5)

Catalog description:

Principles and programming techniques of artificial intelligence: LISP, symbol manipulation, knowledge representation, logical and probabilistic reasoning, learning, language understanding, vision, expert systems, and social issues. Not open for credit to students who have completed 473.

Prerequisites: CSE 373.

Textbook(s) and/or other required material:


Course objectives:

1. Survey fundamental techniques of artificial intelligence including knowledge representation, reasoning, and computer understanding of sensory and language information.
2. Provide hands-on experience designing and implementing programs that embody techniques of artificial intelligence.

Topics covered:

Definitions of intelligence, the Turing test, LISP, symbol manipulation, techniques for web-based intelligent agents, knowledge representation, logical and probabilistic reasoning, commonsense reasoning methods, learning, natural language understanding, computer vision, expert systems, and social issues.

Class/laboratory schedule:

MWF 12:30-1:20. Most classes are lecture format, but they occasionally include in-class exercises, quizzes, and 2-3 class meetings are held in a lab.
Catalog description:

Design and analysis of algorithms and data structures. Efficient algorithms for manipulating graphs and strings. Fast Fourier Transform. Models of computation, including Turing machines. Time and space complexity. NP-complete problems and undecidable problems

Prerequisites: CSE 373.

Textbook(s) and/or other required material:

Udi Manber, Introduction to Algorithms: A creative approach, Addison-Wesley.

Course objectives:

Teach the basics of algorithms, computability and complexity to students who are not CSE majors

Topics covered:

Review of Induction
Basic Algorithm Design Techniques
Graph Algorithms
Fast Fourier Transform
Turing machines and Computability
Basics of Complexity
NP-completeness & Reductions

Class/laboratory schedule:

3 lectures per week. Bi-weekly assignments. Tests.
Appendix I
(continued)

C. Faculty CVs

- Anderson, Richard
- Anderson, Thomas
- Arnstein, Lawrence
- Baer, Jean-Loup
- Beame, Paul
- Borning, Alan
- Borriello, Gaetano
- Chambers, Craig
- Curless, Brian
- Dickey, Martin
- Diorio, Chris
- Domingos, Pedro
- Dooley, Emer
- Dugan, Benedict
- Ebeling, Carl
- Eggers, Susan
- Etzioni, Oren
- Fox, Dieter
- Gribble, Steven
- Halevy, Alon
- Karlin, Anna
- Kautz, Henry
- Ladner, Richard
- Lazowska, Ed
- Levy, Henry
- Mones, Barbara
- Notkin, David
- Perkins, Hal
- Popovic, Zoran
- Rao, Rajesh
- Ruzzo, Walter L.
- Salesin, David
- Seitz, Steven
- Sengupta, Rimli
- Shapiro, Linda
- Snyder, Lawrence
- Suciu, Dan
- Tanimoto, Steven
- Tompa, Martin
- Weld, Daniel
- Wetherall, David
- Zahorjan, John
Anderson, Richard
Professor (1986 Ass't, 1991 Assoc, 1998 Full)

Education:
- B.A., Mathematics, Reed College, 1981
- Ph.D., Computer Science, Stanford, 1986

Other Experience:
- Visiting Professor, Indian Institute of Science, Bangalore, India, 1993-1994

Consulting, patents, etc.:
- Design Intelligence, 1997-1999

Principal publications of last five years:

Scientific and professional societies of which a member:
- ACM

Honors and awards:
- Best Paper Award, 1996 ACM Symp. on User Interface Software and Technology
- Fulbright Senior Scholar Award, 1993.
- Phi Beta Kappa, Reed College, 1981.
Institutional and professional service in last five years:

- 1999-2000 Department Executive Committee, Professional Master's Program Coordinator, Masters Admissions Committee (Chair), Tutored Video Instruction Coordinator, Faculty Recruiting Committee.
- 1998-1999 Professional Master's Program Coordinator, Masters Admissions Committee (Chair), Educational Technology and Distance Learning
- 1997-1998 Professional Masters Program Coordinator, Masters Admissions (Chair), Extension Liaison
- 1996-1997 Graduate Admissions (Chair), Department Executive Committee, Professional Masters Program Coordinator.
- 1995-1996 Graduate Admissions (Chair), Department Executive Committee, Professional Masters Program Coordinator, Departmental Retreat Organizer.

States in which registered: None

Level of activity: prof societies - low; research - medium; consulting/summer work in industry - none.
Anderson, Thomas  
*Professor (1997 Assoc, 2001 Full)*

**Education:**
- Ph.D., Computer Science, University of Washington, 1991
- M.S., Computer Science, University of Washington, 1989
- A.B., Philosophy, Harvard University, 1983

**Other Experience:**
- Computer Science Division, University of California, Berkeley.  
- GenRad Incorporated (manufacturer of automated test equipment)  

**Consulting, patents, etc.:**
- Digital Equipment Corporation Systems Research Center and Western Research Lab,  
  Palo Alto, CA
- NaviSoft Corporation, Santa Barbara, CA
- Xerox Palo Alto Research Center, Palo Alto, CA

**Principal publications of last five years:**
  "Thread Management for Shared-Memory Multiprocessors."  
  *Handbook for Computer Science*.
  Appeared as an appendix in,  
  and P. Galvin. Addison-Wesley (November 1993).
- Drew Roselli, Jay Lorch, and Tom Anderson.  
  "A Comparison of File System Workloads."  
- Neal Cardwell, Stefan Savage, and Tom Anderson.  
  "Modeling TCP Latency."  
  To appear in *Proceedings of the 2000 IEEE Infocom Conference*,  
  Tel-Aviv, Israel, March 2000.
- Amit Aggarwal, Stefan Savage, and Tom Anderson.  
  "Understanding the Performance of TCP Pacing."  
  To appear in *Proceedings of the 2000 IEEE Infocom Conference*,  
  Tel-Aviv, Israel, March 2000.
- Neil Spring, Maureen Chesire, Mark Berryman, Vivek Sahasranaman,  
  Thomas Anderson, and Brian Bershad.  
  "Receiver Based Management of Low Bandwidth Access Links."  
  To appear in *Proceedings of the 2000 IEEE Infocom Conference*,  
  Tel-Aviv, Israel, March 2000.
- Stefan Savage, Neal Cardwell, David Wetherall and Tom Anderson.  
  "TCP Congestion Control with a Misbehaving Receiver."  
- Vahdat, M. Dahlin, T. Anderson and A. Aggarwal.  
  "Active Names: Flexible Location and Transport of Wide-Area Resources."  
  *Proc. 1999 USENIX Symposium on Internet Technologies and Systems (USITS)*,  
  October 1999.
"The End-to-End Effects of Internet Path Selection."  

• Stefan Savage, Neal Cardwell and Tom Anderson.  
"The Case for Informed Transport Protocols.”  

"Detour: A Case for Intelligent Internet Routing and Transport."  

• R. Wang, T. Anderson, and D. Patterson.  
"Virtual Log-Based File Systems for a Programmable Disk."  

• N. McKeown and T. Anderson.  

• Dennis Lee, Patrick Crowley, Jean-Loup Baer, Tom Anderson, and Brian Bershad.  
"Execution Characteristics of Desktop Applications on Windows NT."  

• R. Wang, A. Krishnamurthy, R. Martin, T. Anderson, and D. Culler.  
"Modeling Communication Pipeline Latency."  

• Vahdat and T. Anderson.  
"Transparent Result Caching."  

"SLIC: An Extensibility System for Commodity Operating Systems."  

• R. Wang, T. Anderson, and M. Dahlin.  
"Experience with a Distributed File System Implementation."  

GLUnix: A Global Layer UNIX for a Network of Workstations.  

• Vahdat, T. Anderson, and M. Dahlin.  
WebOS: Operating System Services for Wide Area Applications.  

• Belani, A. Vahdat, T. Anderson, and M. Dahlin.  
CRISt: A Wide Area Security Architecture.  

• S. Chandra, M. Dahlin, B. Richards, R. Wang, T. Anderson, and J. Larus.  
"Experience with a Language for Writing Coherence Protocols."  
Proc. of the USENIX Conference on Domain-Specific Languages (October 1997).

Eraser: A Dynamic Race Detector for Multi-Threaded Programs."


Scientific and professional societies of which a member:
- Associate Editor, ACM Transactions on Computer Systems, 1998-pres.
- Program Committee Member, ACM SIGCOMM'00 Conference on Applications, Technologies, Architectures and Protocols for Computer Communications, 2000.
- Program Committee Member, Third Symposium on Operating System Design and Implementation, 1999.
- Member, NSF Panel, Research Directions in Computer Networking, November 1998.
- Program Committee Member, Fourth High Performance Computer Architecture Conference (HPCA-4), 1998.
- Program Committee Member, First USENIX NT Symposium, 1997.
- Program Committee Member, Seventh ACM Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-VII), October 1996.

Honors and awards:
- NSF Presidential Faculty Fellowship, 1994.
- NSF Young Investigator Award, 1992.
- IBM Graduate Ph.D. Fellowship, 1989.
Institutional and professional service in last five years:

- Nachos, a software project for teaching undergraduate operating systems.  
  http://www.cs.washington.edu/homes/tom/nachos
- Salsa, a software project for teaching undergraduate networking.  
  http://www.cs.washington.edu/education/courses/461/98wi/project
- Lecture notes and homework assignments for undergraduate operating systems, approx. 300 pages. Spanish translation available.  
  http://www.cs.berkeley.edu/homes/tom/cs162sp96

States in which registered: none
Level of activity: prof societies - medium; research - medium; consulting/summer work in industry - low.
Arnstein, Lawrence  
Research assistant professor (1999 Ass't)

Education:
- B.S., Computer Engineering, Case Western Reserve University, 1985  
- M.S., Computer Engineering, Carnegie Mellon University, 1989  
- Ph.D., Computer Engineering, Carnegie Mellon University, 1993

Other Experience:
- Stratos Product Development Group, Seattle WA, 7/98-present  
  Business Development and Electrical Engineering  
- Cadence Design Systems, Inc. Yokohama, Japan  
  Director of Program Management Department 7/94-7/98  
  Major role in establishing Cadence’s consulting services business unit in Japan.  
  Highlights include: Managing successful multi-million dollar consulting projects to  
  improve design processes for designing large scale integrated circuits. Selling and  
  managing product design services for Cadence clients in Japan. Leading the  
  development and adoption of new division level business processes for the quality  
  and financial control of our growing consulting services business. Guiding culturally  
  diverse and geographically distributed organizations. Developing new design  
  automation methods and technologies through client funded consulting projects.  
  Some of these results have been published in commercial conferences, and can now  
  be found in standard products from Cadence.

Consulting, patents, etc.:
- Quicksilver Technology, Inc. January 2001 to Present,  
  Consulting on the design of compiler and synthesis technology for highly  
  parallel/configurable architectures.

Principal publications of last five years:
- Ubicomp 2000 Submission: Labscape, Experiment Capture in the Biology  
  Laboratory, Submitted April 2001.  
- Arnstein, L., Sigurdsson, S., Franzia, R., “Ubiquitous Computing in the Biology  
  Laboratory,” Journal of Laboratory Automation (JALA), Vol 6, no 1, March 2001.  
- SuperComputing 2000 Panel, Convergence of the Extremes, with David Culler  
- The Sixth Annual Workshop on Frontiers in Distributed Information Systems June  
- CHI 2001 Workshop "Building the Ubiquitous Computing User Experience".  

Scientific and professional societies of which a member:
- IEEE  
- ACM

Honors and awards:
- National Science Foundation Graduate Research Fellow  
- Reinberger Memorial Case Alumni Association Merit Scholar

Institutional and professional service in last five years:
- Article Reviewer for Transactions on VLSI  
- NSF Panel for CADRE'00  
- Reviewer for Special Issue of IEEE Computer Magazine
States in which registered: none
Level of activity: prof societies - none; research - high; consulting/summer work in industry - medium.
Baer, Jean-Loup
Professor (1969 Ass’t, 1974 Assoc, 1979 Full)

Education:
- Diplome d'Ingenieur, Electrical Engineering, Universite de Grenoble, France, 1960
- Doctorat 3e Cycle, Computer Science, Universite de Grenoble, France, 1963
- Ph.D., Engineering, UCLA, 1968

Other Experience:
- University of Minnesota Oct. 1984 - Nov 1984 Control Data Corporation Visiting Chair
- University of Paris VI Sept. 1979 - July 1980 Visiting Professor
- UCLA June 1965-Sept. 1969 Acting Assistant Professor of Computer Science (1968 - 1969)

Consulting, patents, etc.:
- ATT, ETS

Principal publications of last five years:

Scientific and professional societies of which a member:

• IEEE Computer Society
• ACM (SIGARCH, SIGOPS)

Honors and awards:

• Guggenheim Fellowship, 1979-1980
• IEEE Fellow 1992
• ACM Fellow 1997

Institutional and professional service in last five years:

• ACM SIGARCH Chair 1995-99
• ACM SGB (Sec-Large SIG Advisor) 1998-2000
• Area Editor Journal of Parallel and Distributed Computing 1987-present
• Associate Editor Computer Languages 1975-1999
• Program Committee Chair HPCA-4, 1998 + numerous PC memberships
• External Review Committee, Dept of CS Univ. of Minnesota 1998
• Chair Search Committee, Dean of Engineering 1995-1996
• ABET Review Coordinator for CSE 1996
• CSE Executive Committee 1996-98
• College of Engineering Faculty Governance team (Chair) 1998-1999
• UIF2 Review Committee 1999
• Faculty Senate 1999-2000

States in which registered: none

Level of activity: prof societies - high; research - high; consulting/summer work in industry - none.
Beame, Paul
Professor (1987 Ass’t, 1993 Assoc, 1999 Full)

Education:
• B.Sc., Mathematics, University of Toronto, 1981
• M.Sc., Computer Science, University of Toronto, 1982
• Ph.D., computer Science, University of Toronto, 1987

Other Experience:
• Postdoctoral Research Associate, Laboratory for Computer Science, M.I.T., 1986-7
• Lecturer for course, Discrete Mathematics for Computer Science, University of Toronto, Summer 1985

Consulting, patents, etc.:
• IBM Almaden Research, Summer 1997

Principal publications of last five years:


Scientific and professional societies of which a member:
• Association for Computing Machinery

Honors and awards:
• Presidential Young Investigator Award, National Science Foundation, 1988
• University of Toronto Open Fellowship, 1985-6
• Natural Sciences and Engineering Research Council Postgraduate Scholarship, 1981-5

Institutional and professional service in last five years:
• Co-chair, DIMACS Workshop on Inherent Complexity of Problems, April 2000
• Associate Editor, Computational Complexity
• Program Committee Chair, IEEE Symposium on Foundations of Computer Science, 1999
• Program Committee Member, Computational Complexity Conference, 1999
• NSF CCR-TOC CAREER Awards Panel, 1998
• Program Advisory Committee, Fields Institute special half-year on computational complexity, 1998.
• Program Committee Member, ACM Symposium on Theory of Computing, 1997
• Program Committee Member, International Computing and Combinatorics Conference, 1997
• Co-chair, DIMACS Workshop on Feasible Arithmetic and Proof Complexity, April 1996
• Steering committee, DIMACS special year on Logic and Algorithms 1995-6.
• University of Washington Faculty Senate, 1995-8

States in which registered: none
Level of activity: prof societies - medium; research - high; consulting/summer work in industry - low.
Education:

- B.A., Mathematics, Reed College, 1971
- M.Sc., Computer Science, Stanford University, 1973
- Ph.D., Computer Science, Stanford University, 1979

Other Experience:

- Visiting Professor at Monash University, Australia, January-February 2001.
- Visiting Professor at University of Melbourne and Monash University, Australia (on sabbatical leave from UW) January--July 1997
- Visiting Scientist at Rank Xerox EuroPARC, Cambridge, England (on sabbatical leave from UW), Sept 1989 - Sept 1990
- Postdoctoral Fellow, Department of Artificial Intelligence, University of Edinburgh, 1979 – 1980

Consulting, patents, etc.:

- Aldus Corporation, Seattle, Washington
- American Bell, Indianapolis, Indiana
- Apple Computer, Cupertino, California
- Atari Sunnyvale Research Center, Sunnyvale, California
- Axon Corporation, Seattle, Washington
- Corgraphics, Inc., Seattle, Washington
- Data I/O, Redmond, Washington
- Intel Corporation, Aloha, Oregon
- Object Technology International, Ottawa, Canada
- Rank Xerox EuroPARC, Cambridge, England
- Teklicon, Mountain View, California
- Tektronix Computer Research Laboratory, Beaverton, Oregon
- Xerox Palo Alto Research Center, Palo Alto, California

Principal publications of last five years:


Scientific and professional societies of which a member:

• Association for Computing Machinery
• Computer Professionals for Social Responsibility

Honors and awards:

• Forsythe Memorial Award, March 1976. (This is a Stanford Computer Science Department award for exceptional service in teaching by a graduate student.)
• Faculty Recognition Award, Minority Science and Engineering Program, University of Washington, February 1996 (for work in setting up and maintaining a tutoring program for women and minority students in computer science and engineering).
• Best Paper Award, 1996 ACM Symposium on User Interface Software and Technology. (The paper is Alan Borning, Richard Anderson, and Bjorn Freeman-Benson, "Indigo: A Local Propagation Algorithm for Inequality Constraints.")

Institutional and professional service in last five years:

• Organizing Committee Member, International Conferences on Constraint Programming, 1993--present.
• Program Committee, Constraints Stream, First International Conference on Computational Logic (conference held in London, July 2000).
• Program Committee, ACM Symposium on User Interface Software Technology (symposium held in San Diego, November 2000).
• Doctoral Program Committee, Seventh International Conference on Principles and Practice of Constraint Programming (CP 2001). Conference to be held November 2001, Paphos, Cyprus.
• Dept of Computer Science & Engineering, Colloquia and Distinguished Lecturers Chair, 1997--2000.
• Dept of Computer Science & Engineering, Departmental Retreat organizer, 1999.
• Dept of Computer Science & Engineering, Graduate admissions committee, 1997-1998.
• Dept of Computer Science & Engineering, Faculty teaching evaluation coordinator, 1997-1998.
• Dept of Computer Science & Engineering, Curriculum Committee Chair, 2000-2001.
• College of Engineering Educational Policy Committee, Chair, 1995--96; member, 1999-2002.
• College of Engineering Physics Course Review Committee, Chair, 2000.
• Reviewing for National Science Foundation, Australian Research Council, numerous journals and conferences

**States in which registered:** none

**Level of activity:** prof societies - medium; research - high; consulting/summer work in industry - low.
Borriello, Gaetano
Professor (1988 Ass't, 1993 Assoc, 1998 Full)

Education:
- B.S., Electrical Engineering, Polytechnic Institute of New York, 1979
- M.S., Electrical Engineering, Stanford University, 1981
- Ph.D., Computer Science, University of California at Berkeley, 1988

Other Experience:
- Xerox Corporation Palo Alto Research Center, Member of Research Staff, 1981-1984

Consulting, patents, etc.:
- Cadence Design Systems, 1998
- Georgia Board of Regents, 1999-2000
- Consystant Design Technologies (Chair, Technical Advisory Board), 1999-
- Intel, 2000
- Microsoft Research, 2000

Principal publications of last five years:


Scientific and professional societies of which a member:

• IEEE/Computer Society
• ACM/SIGDA
• Computer Professionals for Social Responsibility

Honors and awards:

• Fulbright Pisa Chair Award - Scuola Superiore Sant'Anna, Pisa, Italia, 1995-1996
• Distinguished Teaching Award - University of Washington, 1995

Institutional and professional service in last five years:

• Member, NSF/CISE Advisory Board, 2001-
• Program chair, Ubiquitous Computing Conference, 2002
• Program co-chair, Ubiquitous Computing Dagstuhl Workshop (with H. Gellersen, F. Mattern), 2001
• Associate editor, IEEE Transactions on CAD of Integrated Circuits and Systems, 1997-1999
• Program chair, 1st Workshop on Software Engineering for Wearable and Pervasive Computing, 2000
• Program co-chair, 4th University of Washington/Microsoft Research Summer Institute on the Technologies of Invisible Computing, 1999
• Program chair, 5th ACM/IEEE International Workshop on Hardware/Software Co-design, 1997
• General chair, 6th ACM/IEEE International Workshop on Hardware/Software Co-design, 1998
• Steering committee, Intel Computing Continuum Conference, 2000
• Program committee member, International Symposium on Wearable Computing (ISWC), 1999-2000
• Program committee member, International Symposium on System Synthesis (ISSS), 1997-1999
• Program committee member, Architectural Support for Programming Languages and Operating Systems, 2000
• Provost's ad Hoc Committee on Faculty Responsibilities & Rewards, 1998
• Associate Chair for Educational Programs, 1998-2000

States in which registered: none
Level of activity: prof societies - low; research - high; consulting/summer work in industry - medium.
Chambers, Craig
Associate professor (1991 Ass't, 1997 Assoc)

Education:
- S.B., Computer Science, MIT, 1986
- Ph.D., Computer Science, Stanford University, 1992

Principal publications of last five years:
- David Grove and Craig Chambers. An Assessment of Call Graph Construction Algorithms. Accepted to ACM Transactions on Programming Languages and Systems (TOPLAS).
• Brian Grant, Markus Mock, Matthai Philipose, Craig Chambers, and Susan Eggers. Annotation-Directed Run-Time Specialization in C. In Proceedings of the ACM SIGPLAN Symposium on Partial Evaluation and Semantics-Based Program Manipulation (PEPM’97), Amsterdam, the Netherlands, June 1997.

Scientific and professional societies of which a member:
• Association of Computing Machinery (ACM)
• ACM Special Interest Group on Programming Languages (SIGPLAN)

Institutional and professional service in last five years:
• Program committee, 2001 ACM Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA ’01)
• General chair, International Symposium on Memory Management (ISMM ’00)
• Program committee, International Conference on Functional Programming (ICFP ’00)
• Program committee, Seventh International Workshop on Foundations of Object-Oriented Languages (FOOL ’00)
• Program committee, 1999 ACM Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA ’99)
• Program committee, International Symposium on Static Analysis (SAS ’99)
• Chair, program committee, 1998 ACM Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA ’98)
• Program committee, Workshop on Profile and Feedback-Directed Compilation. At the 1998 International Conference on Parallel Architectures and Compilation Techniques (PACT ’98)
• Program committee, 1998 European Conference on Object-Oriented Programming (ECOOP ’98)
• Organizing and program committee, Second International Workshop on Types in Compilation (TiC ’98)
• Program committee, 1997 ACM Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA ’97)
• Program committee, Seventh International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS ’96)
• Program committee, 2nd International Symposium on Object Technologies for Advanced Software (ISOTAS ’96)

States in which registered: none
Level of activity: prof societies - medium; research - high; consulting/summer work in industry - none.
Curless, Brian  
Assistant professor (1998 Ass't)

Education:
- Ph.D., Electrical Engineering, Stanford University, 1997
- M.S., Electrical Engineering, Stanford University, 1991
- B.S., Electrical Engineering, University of Texas at Austin, 1988

Other Experience:
- Visiting Scientist, Digital Michelangelo Project, Florence, Italy (Winter 1999).
- Scientific Advisory Board Member, Paraform, Inc., Mountain View, CA (1998 - present).

Principal publications of last five years:
- "Image Analogies," Aaron Hertzmann, Charles Jacobs, Nuria Oliver, Brian Curless, and David Salesin. To appear in SIGGRAPH’01.
- "A volumetric method for building complex models from range images," Brian Curless and Marc Levoy, SIGGRAPH ’96, New Orleans, LA, 4-9 August 1996.

Scientific and professional societies of which a member:
- Association for Computing Machinery
- Institute of Electrical and Electronics Engineers, Inc.
- Eurographics
Honors and awards:

- Nominee for UW Distinguished Teaching Award (2001)
- ACM student nomination for best instructor (2000)
- CSE Department Packard Fellowship nominee (2000)
- Sloan Fellowship for Computer Science, University of Washington (2000)
- NSF CAREER Award, University of Washington (1999)
- Achievement Rewards for College Scientists (ARCS) fellowship (1993)
- Gores Award for Teaching Excellence, Stanford (1992)
- Graduated Summa Cum Laude, University of Texas (1988)

Institutional and professional service in last five years:

- Co-organized a “3D Photography” course for CVPR’99, SIGGRAPH’99, SIGGRAPH ’00
- Presenter for “Digital geometry processing” course submitted (and accepted) for SIGGRAPH ’01
- Reviewer for SIGGRAPH ’00
- Reviewer for IEEE Transactions on Visualization and Computer Graphics
- Reviewer for Computers and Graphics

States in which registered: none

Level of activity: prof societies - low; research - high; consulting/summer work in industry - low.
Martin, Dickey  
*Senior lecturer (1996 Lecturer, 2000 Senior Lecturer)*

**Education:**
- B.A., Mathematics, Kent State University, 1969
- M.S., Mathematics, University of Kentucky, 1971
- Ph.D., Computer Science, Arizona State University, 1992

**Other Experience:**
- 1978-1986 Honeywell Information Systems, Phoenix, AZ
- 1989 (summer) Siemens (Munich)
- 1990 (spring) Grand Canyon University, Phoenix, AZ
- 1991-1996 Illinois College, Jacksonville, IL

**Scientific and professional societies of which a member:**
- ACM
- ACL
- MAA

**Institutional and professional service in last five years:**

**States in which registered:** none

**Level of activity:** prof societies - medium; research - low; consulting/summer work in industry - none.
Diorio, Chris
Associate professor (1997 Ass’t, 2001 Assoc)

Education:
- B.A., Physics, Occidental College, 1993
- M.S., Electrical Engineering, California Institute of Technology, 1984
- Ph.D., Electrical Engineering, California Institute of Technology, 1997

Other Experience:
- Senior Staff Engineer, TRW, Inc., 1991-1997
- Senior Staff Scientist, American Systems Corporation, 1989-1991
- Member of the Technical Staff, TRW, Inc., 1984-1986

Consulting, patents, etc.:

Principal publications of last five years:


Scientific and professional societies of which a member:
• Member, IEEE

Honors and awards:
• A UW Distinguished Teaching Award on June 7, 2001.
• An ONR Young Investigator Award on February 6, 2001.
• An Alfred P. Sloan Foundation Research Fellowship on Feb. 4, 2000.
• Distinguished Lecture Series invitee, the University of Virginia, Department of Computer Science, Jan. 24, 2000. Talk title: "Biologically Inspired Computation".
• An NSF CAREER Award on May 15, 1998.

Institutional and professional service in last five years:
• Panel member, NRC Committee on Frontiers at the Interface Between Computing and Biology. Sponsors: (1) NAS Computer Science and Telecommunications Board, and (2) DARPA Information Technology Office (2000-2001)
• Floating-gate workshop organizer and instructor at the 1999 NSF Telluride Workshop on Neuromorphic Engineering, held in Telluride, CO, from June 27 to July 17, 1999.
• Co-organizer, 1998 UW/MSR Summer Workshop on Intelligent Systems, held at the University of Washington from August 18-23, 1998.
• Floating-gate workshop organizer and instructor at the 1998 NSF Telluride Workshop on Neuromorphic Engineering, held in Telluride, CO, from June 28 to July 13, 1998.
• Floating-gate workshop organizer and instructor at the 1998 NSF Telluride Workshop on Neuromorphic Engineering, held in Telluride, CO, from June 24 to July 14, 1996.
• Floating-gate workshop organizer and instructor at the 1998 NSF Telluride Workshop on Neuromorphic Engineering, held in Telluride, CO, from June 26 to July 9, 1995.
• Departmental committees: 2000-2001 Faculty recruiting, staff ombudsperson, 1999-2000 Faculty recruiting, staff ombudsperson, Endowed Chair selection, undergraduate program expansion, 1998-1999 Adjunct to the EE faculty recruiting committee, 1997-1998 CSE co-op liaison

States in which registered: none
Level of activity: prof soc - medium; research - high; consulting/summer work in industry - medium.
Domingos, Pedro
Assistant professor (1999 Ass't)

Education:
- Licenciatura, Electrical Engineering and Computer Science, Instituto Superior Tecnico, Lisbon, Portugal, 1988
- M.S., Electrical Engineering and Computer Science, Instituto Superior Tecnico, Lisbon, Portugal, 1992
- M.S., Information and Computer Science, University of California at Irvine, 1994
- Ph.D., Information and Computer Science, University of California at Irvine, 1997

Other Experience:
- Assistant professor, Instituto Superior Tecnico, Portugal (1997-99)
- Research and teaching assistant, Instituto Superior Tecnico, Portugal (1987-92)
- Intern, INESC, Portugal (1986-88)
- Researcher, INESC, Portugal (1988-89)

Consulting, patents, etc.:
- Irvine Research Corporation (1994)

Principal publications of last five years:

• "A Unified Bias-Variance Decomposition for Zero-One and Squared Loss", in Proceedings of the Seventeenth National Conference on Artificial Intelligence, Austin, TX, 2000.


• "Mining High-Speed Data Streams", first author, with Geoff Hulten, in Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining, Boston, MA, 2000.


• "Occam's Two Razors: The Sharp and the Blunt", in Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, New York, 1998.


• "Why Does Bagging Work? A Bayesian Account and its Implications", in Proceedings of the Third International Conference on Knowledge Discovery and Data Mining, Newport Beach, CA, 1997.


• "Linear-Time Rule Induction", in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, OR, 1996.

• "Efficient Specific-to-General Rule Induction", in Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, OR, 1996.


• "Rule Induction and Instance-Based Learning: A Unified Approach", in Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, Montreal, Canada, 1995.

Scientific and professional societies of which a member:

• ACM / SIGKDD, SIGART, SIGMOD

• IEEE / Computer Society

• AAAI

Honors and awards:

• NSF CAREER Award (2000)
• IBM Faculty Partnership Award (2000)
• KDD-99 Best Paper Award
• KDD-98 Best Paper Award
- Fulbright Scholarship (1992-97)
- University of California Regents' Dissertation Fellowship (1996)

**Institutional and professional service in last five years:**

- Editorial board member, Machine Learning (current)
- Editorial board member, Journal of Artificial Intelligence Research (current)
- Editorial board member, Intelligent Data Analysis (current)
- Editorial board member, Applied Intelligence (current)
- Editorial board member, Evaluation of Intelligent Systems (current)
- Area chair (supervised learning), ICML-2001
- Panels chair, KDD-2001
- Program committee member, AAAI-97, AAAI-98, AAAI-99, AAAI-2000
- Program committee member, ICML-98, ICML-99, ICML-2000
- Program committee member, KDD-98, KDD-99
- Program committee member, IJCAI-99
- Program committee member, ECML-2000
- NSF panel member (2000)
- Faculty mentor of students from African countries (1998-99)

**States in which registered:** none

**Level of activity:** prof societies - low; research - high; consulting/summer work in industry - low.
Education:
- Ph.D., Business, University of Washington, 2000
- MBA, Strategic Management, University of Washington, 1992
- M. Eng, Computer Engineering, University of Limerick, Ireland, 1986
- B.S., Electronic Engineering, University of Limerick, Ireland, 1982

Other Experience:

Honors and awards:
- 1997 University of Washington Excellence in Teaching Award

States in which registered: none

Level of activity: prof societies - low; research - low; consulting/summer work in industry - none.
Dugan, Benedict
Lecturer (Lecturer 1995)

Education:
- B.S., Science & Technology Studies, Stanford University, 1992
- M.S., Computer Science, University of Washington, 1995

Other Experience:
- 2 years as TA/RA at Stanford and/or UW
- 3 years as Lecturer at UW
- 2 years as Chief Architect at VocalPoint Technologies Inc, San Francisco, CA

States in which registered: none

Level of activity: prof societies - none; research - low; consulting/summer work in industry - medium.
Ebeling, Carl  
Professor (1986 Ass't, 1992 Assoc, 1997 Full)

Education:
- B.S., Physics, Wheaton College, 1971
- M.S., Computer Science, Southern Illinois University - Carbondale, 1976
- Ph.D., Computer Science, Carnegie-Mellon University, 1986

Consulting, patents, etc.:

Principal publications of last five years:

Scientific and professional societies of which a member:
- Association for Computing Machinery
- IEEE
- Sigma Xi

Honors and awards:
• Association for Computing Machinery Distinguished Dissertation Award 1986
• National Science Foundation Presidential Young Investigator 1987
• American Association for Artificial Intelligence Pioneer in Computer Chess 1989
• Burlington Resources Foundation Faculty Achievement Award for Teaching, College of Engineering 1992
• Fulbright Fellowship, University of Mauritius 1993-94
• University of Washington Distinguished Teaching Award 1995
• Allen Newell Award for Research Excellence 1997

**Institutional and professional service in last five years:**

• Chair, Program Committee, Fourth ACM Symposium on Field-Programmable Gate Arrays, February 1996.
• Member, Program Committee, International Conference on Computer Design, November, 1996.
• General Chair and Member of Program Committee, Fifth ACM Symposium on Field-Programmable Gate Arrays, February 1997.
• Member, Program Committee, 7th International Workshop on Field Programmable Logic and Applications, September 1997
• Member, Program Committee, 17th Conference on Advanced Research in VLSI, September 1997.
• Member, Program Committee, Sixth ACM Symposium on Field-Programmable Gate Arrays, February 1998.
• Guest editor, special issue on FPGAs, IEEE Transactions on VLSI Systems, April 1998
• Member, Program Committee, 8th International Workshop on Field Programmable Logic and Applications, September 1998
• Member, Program Committee, Seventh ACM Symposium on Field-Programmable Gate Arrays, February 1999
• Member, Program Committee, Reconfigurable Architectures Workshop, March 1999.
• Member, Program Committee, 9th International Workshop on Field Programmable Logic and Applications, September 1999.
• Member, Program Committee, Eighth ACM Symposium on Field-Programmable Gate Arrays, February 2000
• Associate Editor, IEEE Transactions on VLSI Systems, 1999-
• Graduate Program Coordinator, 1994-1998
• Faculty Senate, 1998-2000

**States in which registered:** none

**Level of activity:** prof societies - medium; research - high; consulting/summer work in industry - low.
Eggers, Susan
Professor (1989 Ass't, 1994 Assoc, 1999 Full)

Education:
- B.A., Economics, Connecticut College, 1965
- M.S., Computer Science, University of California, Berkeley, 1984
- Ph.D., Computer Science, University of California, Berkeley, 1989

Other Experience:
- 1979 - 1983: Computer Scientist, Lawrence Berkeley Laboratory, Department of Computer Science and Mathematics.

Consulting, patents, etc.:
- 1997: Consultant, Compaq (then Digital) Western Research Laboratory
- Thread Properties Attribute Vector Based Thread Selection in Multithreading Processor, CPQ Reference No.: PD96-0116

Principal publications of last five years:
- J. Aldrich, C. Chambers, E.G. Sirer and S. Eggers, Static Analyses for Eliminating Unnecessary Synchronization from Java Programs, Sixth International Static Analysis Symposium (September 1999), pp. 19-38.

Scientific and professional societies of which a member:
• ACM
• IEEE
• AAUW
• AWIS

Honors and awards:
• 1993: Finalist, NSF Presidential Faculty Fellowship
• 1991 - 1994: AT&T Bell Laboratories, PYI Matching Program
• 1991: Finalist, David and Lucile Packard Foundation Fellowship
• 1990 - 1995: NSF Presidential Young Investigator Award
• 1989 - 1991: IBM Faculty Development Award
• 1986 - 1988: IBM Graduate Student Fellowship
• 1985 - 1986: California Fellowship in Microelectronics

Institutional and professional service in last five years:
• Member, College Associate Dean Search: 1999/00
• Chair, Faculty Recruiting Committee: 1999/00
• University Disciplinary Committee, 1998/99
• Graduate Student Recruiting: 1994/95, 1995/96.
• Graduate Student Evaluation Committee: 1993/94, 1994/95.

States in which registered: none
Level of activity: prof societies - low; research - high; consulting/summer work in industry - low.
Etzioni, Oren
Associate professor (1991 Ass't, 1996 Assoc)

Education:
- Ph.D., Computer Science, Carnegie Mellon University, 1991
- M.S., Computer Science, Carnegie Mellon University, 1988
- B.A., Computer Science, Harvard University, 1986

Other Experience:
- Chief Technology Officer, Go2Net, Inc., May 1999 - April 2000
- Founder and Chief Scientist, Netbot, Inc., May 1996 - Present

Consulting, patents, etc.:
- Chief Technology Officer, Go2Net, Inc., May 1999 - April 2000
- The MetaCrawler service at www.metacrawler.com was sublicensed by Netbot to Go2net, Inc.
- The BRUTE data mining software has been licensed to:
  - Boeing Computer Services, December 1994.
  - Over 20 licenses have been granted to universities including Vanderbilt University, the California Institute of Technology, the University of California at Irvine.
- The LCW closed-world reasoning software package was licensed on a non-exclusive basis to Apple, Inc. in 1996.

Principal publications of last five years:


Scientific and professional societies of which a member:
• American Association for Artificial Intelligence
• Association for Computing Machinery

Honors and awards:
• Edge Award for the best use of intelligent technology (in Jango) at the 1997 WebINNOVATION show.
• MetaCrawler selected for Editor's Choice Award for Web Search Engines by PC Magazine (December 1997).
• MetaCrawler selected as Best Web Search Solution by Infoworld (May 1997).
• MetaCrawler chosen as one of 3 finalists in the 1996 C|NET Awards for Best Internet Search Engine.
• Internet Softbot chosen as one of 5 finalists in the 1995 National DISCOVER Awards for Technological Innovation in Computer Software.
• NSF Young Investigator Award 1993.
• NSF Research Initiation Award 1992.
• AT&T Bell Laboratories Fellow 1987-90.

Institutional and professional service in last five years:

• President of AI Access Foundation
• Member of Advisory Board for Journal of Artificial Intelligence Research
• Guest editor, AI Magazine special issue on Intelligence on the Web (June 1997)
• Editorial Board: Journal of Artificial Intelligence Research, 1994-1996
• Editorial Board: Journal of Data Mining and Knowledge Discovery
• Editorial Board: for Springer Series in Agent Technology 1997
• International Conference on Autonomous Agents, 1999 (Program Chair)
• WWW8, 1998 (vice-chair for the "Searching, querying and indexing" track)
• First International Conference on Autonomous Agents, 1997 (Area Chair for Software Agents)
• International Conference on User Modeling, 1997 (program committee)
• American Association for Artificial Intelligence, 1996 (Senior member)
• Machine Learning Conference, 1995 (program committee)
• Second International Workshop on Next Generation Information Technologies and Systems 1995 (program committee)
• Lifelike Computer Characters 1994 and 1995 (program committee)
• AAAI Spring Symposium on Software Agents, 1994 (Chair)
• American Association for Artificial Intelligence, 1994 (program committee)

States in which registered: none

Level of activity: prof soc - medium; research - medium; consulting/summer work in industry - medium.
Education:

- B.S., Computer Science, University of Bonn, Germany, 1990
- M.S., Computer Science, University of Bonn, Germany, 1993
- Ph.D., Computer Science, University of Bonn, Germany, 1998

Other Experience:

- 8/98 - 8/00 Postdoctoral research associate, Department of Computer Science, Carnegie Mellon University

Principal publications of last five years:

The mobile robot Rhino.
AI Magazine, 16(2), Summer 1995.
• D. Fox, S. Thrun, F. Dellaert, and W. Burgard.
Particle filters for mobile robot localization.
• D. Fox.
Markov Lokalisierung für mobile Roboter.
• D. Fox, W. Burgard, H. Kruppa, and S. Thrun.
Efficient Multi-Robot Localization Based on Monte Carlo Approximation.
Map learning and high-speed navigation in RHINO.
• D. Schulz, W. Burgard, D. Fox, and A. B. Cremers.
Tracking Multiple Moving Targets with a Mobile Robot using Particle Filters and Statistical Data Association.
• S. Enderle, M. Ritter, D. Fox, S. Sablatnög, G. Kraetzschmar and G. Palm.
Soccer-Robot Localization Using Sporadic Visual Features.
• R. Simmons, D. Apfelbaum, D. Fox, R. P. Goldman, K. Zita Haigh, D. J. Musliner, M. Pelican, and S. Thrun.
Coordinated Deployment of Multiple, Heterogeneous Robots.
• S. Thrun, D. Fox, and W. Burgard.
Monte carlo localization with mixture proposal distribution.
• S. Thrun, W. Burgard, and D. Fox.
A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping.
• W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun.
Collaborative Multi-Robot Exploration.
• D. Fox, W. Burgard, F. Dellaert, and S. Thrun.
Monte carlo localization: Efficient position estimation for mobile robots.
In Proc. of the National Conference on Artificial Intelligence (AAAI), 1999.
• D. Fox, W. Burgard, H. Kruppa, and S. Thrun.
Collaborative Multi-Robot Localization.
In Proc. of the German Conference on Artificial Intelligence (KI) and the 21st Symposium on Pattern Recognition (DAGM), Germany, 1999. DAGM outstanding paper award.


- W. Burgard, D. Fox, H. Jans, C. Matenar, and S. Thrun. Sonar-based mapping of large-scale mobile robot environments using EM.


• D. Fox, W. Burgard, S. Thrun, and A.B. Cremers.
A hybrid collision avoidance method for mobile robots.
• S. Thrun, D. Fox, and W. Burgard.
Probabilistic mapping of an environment by a mobile robot.

Scientific and professional societies of which a member:
• Institute of Electrical and Electronics Engineers
• American Association for Artificial Intelligence
• Gesellschaft fuer Informatik

Honors and awards:
• ECAI Artificial Intelligence Dissertation Award: European Coordinating Committee for Artificial Intelligence, 2000.
• AKI Dissertation Award: Arbeitsgemeinschaft der deutschen KI-Institute (German AI institutes), 1999.
• Outstanding paper award: 21st Symposium on Pattern Recognition (DAGM), 1999.
• Outstanding paper award: National Conference on Artificial Intelligence (AAAI), 1998.
• First place award: Clean-up tennis court event at the AAAI autonomous mobile robot competition, 1996.
• Second place award: Clean-up an office event at the AAAI autonomous mobile robot competition, 1994.

Institutional and professional service in last five years:
• Workshop on Interactive Robotics and Entertainment (WIRE), 2000.
• IEEE International Symposium on Computational Intelligence in Robotics & Automation (CIRA), 1999.
• Third European Workshop on Advanced Mobile Robots (EUROBOT), 1999.
• Co-chair of IJCAI-01 Workshop on Reasoning with Uncertainty in Robotics, 2001.
• Editorial Board of Autonomous Robots (Special Issue on Integrating Robotics Research).

States in which registered: none
Level of activity: prof societies - low; research - high; consulting/summer work in industry - low.
**Gribble, Steven**  
*Assistant professor (2000 Ass't)*

**Education:**
- Ph.D., Computer Science, University of California at Berkeley, 2000
- M.S., Computer Science, University of California at Berkeley, 1997
- B.S., Combined Computer Science and Physics, University of British Columbia, 1995

**Principal publications of last five years:**
- Systems Directions for Pervasive Computing, by Robert Grimm, Janet Davis, Ben Hendrickson, Eric Lemar, Adam MacBeth, Steven Swanson, Tom Anderson, Brian Bershad, Gaetano Borriello, Steven Gribble, and David Wetherall. The 8th Workshop on Hot Topics in Operating Systems (HotOS-VIII).
- Robustness in Complex Systems, by Steven D. Gribble. The 8th Workshop on Hot Topics in Operating Systems (HotOS-VIII).
- A Network Architecture for Heterogeneous Mobile Computing, by Eric A. Brewer, Randy H. Katz, Elan Amir, Hari Balakrishnan, Yatin Chawathe, Armando Fox, Steven D. Gribble, Todd Hodes, Giao Nguyen, Venkata N.
Padmanabhan, Mark Stemm, Srinivasan Seshan, and Tom Henderson
IEEE Personal Communications.

- Adapting to Network and Client Variation Using Active Proxies: Lessons and Perspectives,
  by Armando Fox, Steven D. Gribble, Yatin Chawathe, and Eric A. Brewer
  Special issue of IEEE Personal Communications on Adapation, August 1998.

- System Design Issues for Internet Middleware Services: Deductions from a Large Client Trace,
  by Steven D. Gribble and Eric A. Brewer
  Proceedings of the 1997 Usenix Symposium on Internet Technologies and Systems,
  Monterey, California, December 1997.

- Scalable Network Services,
  by Armando Fox, Steven D. Gribble, Yatin Chawathe, and Eric A. Brewer

- Orthogonal Extensions to the WWW User Interface Using Client-Side Technologies,
  by Armando Fox, Steven D. Gribble, Yatin Chawathe, Anthony S. Polito, Andrew Huang, Benjamin Ling, and Eric A. Brewer
  Demo at the 10th Annual Symposium on User Interface Software and Technology (UIST '97), Banff, Canada, October 1997.

- Adapting to Client Variability via On-Demand Dynamic Distillation,
  by Armando Fox, Steven D. Gribble, Eric A. Brewer, and Elan Amir
  Proceedings of the ACM Seventh International Conference on Architectural support for Programming Languages and Operating Systems, Cambridge, Massachusetts, October 1-5, 1996.

- Security on the Move: Indirect Authentication using Kerberos,
  by Armando Fox and Steven D. Gribble.

- The Bay Area Research Wireless Access Network (BARWAN),
  by Randy H. Katz, Eric A. Brewer, Elan Amir, Hari Balakrishnan, Armando Fox, Steve Gribble, Todd Hodes, Daniel Jiang, Giao Thanh Nguyen, Venkata Padmanabhan, and Mark Stemm.

- Give and Take: Children Collaborating on One Computer,
  by Kori Inkpen, Kellogg S. Booth, Steven D. Gribble and Maria Klawe.
  Chi '95 Proceedings, Short Papers.

Scientific and professional societies of which a member:

- ACM
- IEEE
- USENIX

Honors and awards:

- ACM Teaching Award (May 2001)
- Cal @ Silicon Valley Fellowship, UC Berkeley CS, 1999.
- Regent’s Fellowship, UC Berkeley, 1995 (graduate program entrance award).
• Banks Scholarship, UBC, 1995.
• Rick Semple Memorial Award, UBC CS, 1995.
• McDonald Memorial Scholarship, UCB Physics, 1995.
• NSERC Undergraduate Summer Research Grant, UBC, 1993 and 1994.
• Science Scholar, UBC, 1993-1995, awarded to the top 20 students in the faculty of science.
• Canadian Information Processing Society Award, 1993.
• National Entrance Scholarship, UBC, 1990-1995, the top UBC entrance scholarship.

Institutional and professional service in last five years:

• Program Committee Member and WIP Organizer, The Third USENIX Symposium on Internet Technologies and Systems (USITS '01), March 2001.
• Publications and Publicity Chair, The Second IEEE WOrkshop on Internet Applications (WIAPP '01), July 2001

States in which registered: none
Level of activity: prof societies - low; research - high; consulting/summer work in industry - none.
Halevy, Alon
Associate professor (1998 Ass't, 2001 Assoc)

Education:
- B.S., Computer Science and Mathematics, Hebrew University, Jerusalem, 1988
- Ph.D., Computer Science, Stanford University, 1993

Other Experience:
- 4 and a half years at AT&T (Bell) Laboratories as a Principal Member of Technical Staff

Consulting, patents, etc.:
- Founder of Nimble Technologies, a Seattle based company for providing advanced XML query processing tools.
- Author of 10 patents.

Principal publications of last five years:

Scientific and professional societies of which a member:
- ACM
- IEEE

Honors and awards:
- NSF Career Award – 2000
- Sloan Fellowship – 1999
- AAAI Best Paper Award - 1996
Institutional and professional service in last five years:

- Program committees:
  SIGMOD, VLDB -- 2001
  AAAI (senior PC member) 2000
  SIGMOD Panel Chair -- 2000
  PODS -1999
  SIGMOD - 1998
  IEEE Data Engineering - 97,98,99
- Phd Admissions committee -- 2000
- Faculty recruiting committee - 1999

States in which registered: none
Level of activity: prof societies - high; research - high; consulting/summer work in industry - high.
Karlin, Anna  
Professor (1996 Ass't, 1996 Assoc, 1998 Full)

Education:
- Sc.B., Mathematical Sciences, Stanford University, 1981
- Ph.D., Computer Science, Stanford University, 1987

Other Experience:

Consulting, patents, etc.:
- Patent No. 5032987, "Dynamic Hashing with Multiple Tables" (with A. Broder), 1990.

Principal publications of last five years:
- Balanced Allocations, (with Y. Azar, A. Broder and E. Upfal).
- Implementation and Performance of Integrated Application-Controlled Caching, Prefetching and Disk Scheduling, (with P. Cao, E. Felten and K. Li).
- Near-Optimal Parallel Prefetching and and Caching, with T. Kimbrel.
- A Trace-Driven Comparison of Algorithms for Parallel Prefetching and Caching, (with T. Kimbrel, A. Tomkins, R.H. Patterson, B. Bershad, P. Cao, E. Felten, G. Gibson, K. Li)

Scientific and professional societies of which a member:

- ACM
- SIGACT

Honors and awards:


Institutional and professional service in last five years:

- Member of Editorial Board, SIAM Journal on Computing
- Member, NRC Computer Science and Telecommunications Board

States in which registered: none
Level of activity: prof societies - medium; research - high; consulting/summer work in industry - none.
Kautz, Henry
Associate professor (2000 Assoc)

Education:
- B.A., Mathematics, Cornell University, 1978
- M.A., Creating Writing, Johns Hopkins University, 1980
- M.S., Computer Science, University of Toronto, 1982
- Ph.D., Computer Science, University of Rochester, 1987

Other Experience:
- Department Head, AT&T Laboratories, 1994-1997.

Consulting, patents, etc.:

Principal publications of last five years:

Scientific and professional societies of which a member:

• American Association for Artificial Intelligence (AAAI).
• Association for Computing Machinery (ACM).

Honors and awards:

• 1998 Elected to the Executive Council of the American Association for Artificial Intelligence
• 1997 Elected Fellow of the National Association for Artificial Intelligence
• 1996 Best Paper, 13th National Conference on Artificial Intelligence
• 1993 Best Paper, 11th National Conference on Artificial Intelligence
• 1989 Computers and Thought Award, by the International Joint Conference on Artificial Intelligence
• 1989 Best Paper, 1st International Conference on Knowledge Representation and Reasoning
• 1988 Best Paper, 7th Biennial Conference of the Canadian Society for the Computational Studies of Intelligence

Institutional and professional service in last five years:

• 2001 Advisory Committee, International Joint Conference on Artificial Intelligence
• 2001 Chair, LICS Workshop on Theory and Applications of Satisfiability Testing
• 2000 Program Co-Chair of the National Conference on Artificial Intelligence
• 2000 Reviewer for NASA Advanced Cross-Enterprise Technology Development
• 2000 Program Committee International Conference on Constraint Programming (CP2000)
• 1999 Associate Program Chair for the National Conference on Artificial Intelligence
• 1998 Chair AIPS-2000 Working on Planning as Combinatorial Search
• 1996-1998 Associate Editor of the Journal of Artificial Intelligence Research
• 1996-2001 Reviewer for the National Conference on Artificial Intelligence (AAAI) and the International Conference on Artificial Intelligence (IJCAI)

States in which registered: none
Level of activity: prof societies - high; research - high; consulting/summer work in industry - high.
Ladner, Richard
Professor (1971 Ass't, 1976 Assoc, 1981 Full)

Education:
- Ph.D., Mathematics, University of California, Berkeley, 1971
- B.S., Mathematics, St. Mary's College of California, 1965

Other Experience:
- Visiting Positions:
  - University of Toronto, 1977
  - Yale University, 1978
  - Gallaudet University, 1985
  - Mathematical Sciences Institute, Berkeley, 1986
  - Victoria University, Wellington, New Zealand, 1993
  - AT&T Labs, Florham Park, NJ, 1999-2000

Consulting, patents, etc.:
- Microsoft
- IBM
- AT&T

Principal publications of last five years:
- J. D. Fix and R.E. Ladner
- A. LaMarca and R.E. Ladner.
- B.S. Srinivas, R. Ladner, E.A. Riskin, and M. Aziz\~o\{glu.
- J.D. Fix and R.E. Ladner.
- R.E. Ladner, A. LaMarca, and E. Tempero.
- A. LaMarca and R.E. Ladner.
• R.-Y. Wang, E.A. Riskin, and R. Ladner.
  Codebook Organization to Enhance Maximum A Posteriori Detection of Progressive
  Transmission of
  Vector Quantized Images over Noisy Channels.
  Complexity of Sub-Bus Mesh Computations.
• E.S. Hong, R.E. Ladner, and E.A Riskin.
  Group Testing for Wavelet Packet Image Compression.
  IEEE Data Compression Conference (DCC 2001), 73-82.
• A. Bar-Noy and R.E. Ladner.
  Competitive On-Line Stream Merging Algorithms for Media-on-Demand.
  Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms
  (SODA 2001), 364-373. Revised and submitted for publication.
• A.E. Mohr, E.A. Riskin, and R.E. Ladner, Approximately Optimal
• E.S. Hong and R.E. Ladner.
  Group Testing for Image Compression.
  Proceedings IEEE Data Compression Conference (DCC 2000), pp. 3-12, March
  2000.
  Revised and submitted for publication.
• A. E. Mohr, E. A. Riskin, and R. Ladner,
  Generalized Multiple Description Coding through Unequal Error Correction.
• A.E. Mohr, E.A. Riskin, and R.E. Ladner.
  Graceful Degradation over Packet Erasure Channels through Forward Error
  Correction.
  Revised and published.
• R.E. Ladner, J.D. Fix, and A. LaMarca.
  Cache Performance Analysis of Traversals and Random Accesses.
  Proceeding of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms.
  pp. 613-622.
• K. Zatloukal, R.E. Ladner and M.H. Johnson.
  Nearest Neighbor Search in Vector Quantization.
  Workshop on Algorithm Engineering and
  Experimentation, January 1999. Revised and submitted to for AMS Series
  publication.
• J.D. Fix and R.E. Ladner.
  Multiresolution Banded Refinement to Accelerate Surface
  Reconstruction from Polygons.
• A. E. Mohr, E. A. Riskin, and R. Ladner. Bit Allocation for
  Wavelet Image Compression and Uniform Bit Loss.
  Proceedings of the 32th Annual Conference on Information Sciences and Systems,
• A.E. Mohr, E. A. Riskin, and R. Ladner.
  Recovering from Bit Errors in Scalar-Quantized Discrete Wavelet
  Transformed Images .
• A. LaMarca and R.E. Ladner.  
  The Influence of Caches on the Performance of Sorting.  
  Revised and published.
• M.H. Johnson, R. Ladner, and E.A. Riskin.  
  Fast Nearest Neighbor Search for ECVQ and other Modified Distorsion  
  Measures by Comparing Boundary Distance.  
  September, 1996. Revised and published.

Scientific and professional societies of which a member:

• ACM and SIGACT
• IEEE Computer Society
• European Association for Theoretical Computer Science
• American Mathematical Society
• Association for Symbolic Logic

Honors and awards:

• Fullbright, 1992-93
• Guggenheim, 1985-86
• Fellow of ACM, 1994-

Institutional and professional service in last five years:

• External:
  Editor, Journal of Computer and System Sciences  
  NSF Panel for Program in Theoretical Computer Science  
  External Review Committee for Computer Sciences at Purdue University, 1999  
  External Review Committee for Computer Science at CUNY, 1998  
  Referee for several journals
• University:
  Member of Senate Planning and Budget Committee, 1996-1998
• College:
  Member of College of Engineering Promotion and Tenure Committee, 1995-1998
• Department:
  Chair of Faculty Recruiting, 1999  
  Program Coordinator for Professional Masters Program, 2000  
  Chair of Affiliates Program, 2000

States in which registered: none

Level of activity: prof societies - low; research - high; consulting/summer work in industry - medium.
Lazowska, Ed
Professor (1977 Ass't, 1982 Assoc, 1986 Full)

Education:

- A.B., Computer Science, Brown University, 1972
- M.Sc., Computer Science, University of Toronto, 1974
- Ph.D., Computer Science, University of Toronto, 1977

Other Experience:

- Visiting Scientist, DEC Systems Research Center, 1984-85
- Visiting Scholar, Computer Science Department, Stanford University, 1984-85

Consulting, patents, etc.:

- Corporate Technical Advisory Boards:
  - Microsoft Research, 1991-;
  - Voyager Capital, 1996-;
  - E-Quill, 2000-;
  - Ignition, 2000-;
  - Frazier Technology Ventures, 2000-;
  - Madrona Venture Group, 2000-
- Corporate Boards of Directors:
  - Data I/O Corp., 1996-;

Principal publications of last five years:


Scientific and professional societies of which a member:

- National Academy of Engineering
- Association for Computing Machinery (Fellow)
- Institute of Electrical and Electronics Engineers (Fellow)
- IFIP Working Group 7.3 on Computer System Modeling

Honors and awards:

- National Academy of Engineering, 2001
- Bill & Melinda Gates Endowed Chair in Computer Science, 2000
- BAM 100 (one hundred Brown University alumni who had the greatest impact on the twentieth century), 2000
- University of Washington Brotman Award for Instructional Excellence (Departmental), 1999
University of Washington Outstanding Public Service Award, 1998
Alliance for Education "A+ Partnership Award" for Outstanding Contributions to the Seattle Public Schools, 1997
University of Washington Annual Faculty Lecturer, 1996
Fellow of the Institute of Electrical and Electronics Engineers, 1996
Fellow of the Association for Computing Machinery, 1995
Award paper, ACM SIGCOMM '93 Symposium
Award paper, 1993 Machnix Workshop
Award paper, 1989 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems
Award Paper, 1985 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems
Award paper, 7th ACM Symposium on Operating Systems Principles (1977)

Institutional and professional service in last five years:

Professional:
Information Science and Technology (ISAT) Study Group of the Defense Advanced Research Projects Agency, 1998-
Computer Science and Telecommunications Board of the National Research Council, 1996-
Chair, Board of Directors, Computing Research Association, 1997-2001
Chair, ACM A.M. Turing Award Committee, 1999; Member, 1996-2001
National Science Foundation 50th Anniversary Public Advisory Committee, 1998-99
Networking Research Liaison Council, University Corporation for Advanced Internet Development, 1998-
Washington Software Alliance Board of Directors, 1996-
State of Washington Information Services Board, 1995-
Technology Alliance of Washington Council, 1996-; Board of Directors, 2000-; Vice President, 2001-
Chair, University-level review committees for Princeton Univ. Department of Computer Science (1999), Univ. of Virginia Department of Computer Science (1999), Rice Univ. Department of Computer Science (1998)
Standing Advisory Committees for UC Berkeley Department of EECS, Georgia Institute of Technology College of Computing, Hong Kong Univ. of Science and Technology Dep’t of Computer Science, National College of Ireland Program in Informatics

Institutional:
Chair, Department of Computer Science & Engineering, 1993-
Chair, University Advisory Committee on Academic Technology, 1990-
Search Committee for the Dean of the College of Education, 1999-00
Search Committee for Educational Technology Faculty, Coll. of Education, 1998-99
Chair, Committee to Review the Department of Statistics, 1998-99
Search Committee for the Head of the Graduate School of Library and Information Sciences, 1997-98
Committee on the Future of the Graduate School of Library and Information Science, 1996-97
Presidential Task Force on Enrollment Planning, 1996-97

States in which registered: none
Level of activity: prof societies - high; research - low; consulting/summer work in industry - high.
Levy, Henry
Professor (1983 Ass’t, 1988 Assoc, 1994 Full)

Education:
- B.S., Math/Computer Science, Carnegie-Mellon University, 1974
- M.S., Computer Science, University of Washington, 1981

Other Experience:
- Digital Equipment Corporation, 1974 - 1983, Consulting Engineer, member of design team for VAX/VMS operating system, system architect for VAXclusters distributed system.

Consulting, patents, etc.:

Principal publications of last five years:

Scientific and professional societies of which a member:
• Fellow, Association for Computing Machinery
• Senior Member, IEEE

Honors and awards:
• Microsoft Professor (endowed professorship), 2000
• Fellow of ACM, 1995
• Fulbright Research Scholar Award, 1992
• Outstanding Paper Awards:
  3rd European Conf. on Object-Oriented Programming (1989)

Institutional and professional service in last five years:
• Program Chair:
  16th ACM Symp. on Operating Systems Principles
  4th IEEE Int. Workshop on Hot Topics in Operating Systems
• Program Committee:
  6th Int. Symp. on High Performance Computer Architecture
  15th ACM Symp. on Operating Systems Principles
• Associate Editor, ACM Trans. on Computer Systems
• Chairman, ACM SIGOPS (Special Interest Group in Operating Systems)
• Chairman, research evaluation review committee, INRIA France

States in which registered: none
Level of activity: prof societies - high; research - high; consulting/summer work in industry - low.
Mones, Barbara  
*Senior Lecturer (Senior Lecturer 1999)*

**Education:**
- BFA, Art Studio and Art History, University of Michigan, 1974
- MFA, Sculpture and Mixed Media, Rhode Island School of Design, 1979
- Post Graduate Certificate, Animation, Sheridan College, 1986

**Other Experience:**
- Associate Professor (tenured), George Mason University  
  Visiting Scholar -- George Washington University  
  Animation curriculum design and training, technical writing for research and development area.
- Pacific Data Images  
  Animation and technical training and curriculum and professional development
- Industrial Light and Magic  
  Web and graphic design
- GLOBE Program, NASA Goddard
- Professor of Computer Animation And 3D Modeling  
  Lugano Academy Of The Electronic Arts, Switzerland

**Consulting, patents, etc.:**
- Curriculum consulting including Sheridan Animation Program redesign, Marymount College, and several K-12 programs.

**Principal publications of last five years:**
- Artists And Visualization In Education  
  Eurographics '94
- Is Visualization Struggling under the Myth of Objectivity?, IEEE Visualization '95
- Barbara Mones-Hattal, "The Ambiguity Of Dimension," Third International Symposium On The Electronic Arts (Tisea), Sydney, Australia

**Scientific and professional societies of which a member:**
- ACM SIGGRAPH
- College Art Association
- ASIFA

**Honors and awards:**
- Group Achievement Award, Globe Visualization Team, National Aeronautics and Space Administration

**Institutional and professional service in last five years:**
- SIGGRAPH 97 Panels Chair
- SIGGRAPH Art Show Jury Member
- SIGGRAPH Animation Competition Coordinator
- SIGGRAPH Education Committee Art Representative
- Member, Visualization And Presentation Committee Globe Program, White House, Washington D.C.
- Created and edited animation production video for 1999 CSE Graduation
- Contributed as needed for several outreach activities related to Animation on campus -- including this year's Open House, 10 Yr review and others.
- Recommended and helped implement the redesign of LA2 and equipment procurement for the animation area.

**States in which registered:** none

**Level of activity:** prof societies - high; research - low; consulting/summer work in industry - medium.
Notkin, David
Professor (1984 Ass't, 1989 Assoc, 1994 Full)

Education:
- Sc.B., Computer Science, Brown University, 1977
- Ph.D., Computer Science, Carnegie Mellon University, 1984

Other Experience:
- IBM Haifa Research Laboratory, Visiting Researcher, 1997-1998.
- Tokyo Institute of Technology, Department of Computer Science, Visiting Associate Professor, Toshiba Endowed Chair of Intellectual Information Systems, 1990.
- Osaka University, Department of Information and Computer Sciences, Visiting Associate Professor, Endowed Chair of Intelligent Information Processing Systems, 1990-1991.

Consulting, patents, etc.:
- TRW; Siemens; US Army; Bell Laboratories; Tartan Laboratories; IBM; Xerox PARC; US West/AT; Stoel Rives Boley Jones & Grey; Preston Thorgrimson Shidler Gates & Ellis; MacDonald, Hoague & Bayless; Perkins Coie; i5digital.com (principal scientist, member technical advisory board)

Principal publications of last five years:
- Michael Ernst, Adam Czeisler, William G. Griswold, and David Notkin. Quickly Detecting Relevant Program Invariants. 22nd International Conference on Software Engineering (June 2000).


• Michael VanHilst and David Notkin. Using Role Components to Implement Collaboration-Based Designs. OOPSLA-96 (October 1996).


**Scientific and professional societies of which a member:**

- Association for Computing Machinery, Special Interest Groups: Programming Languages, Software Engineering.
- Institute of Electrical and Electronic Engineers, IEEE Computer Society. Society of the
- Sigma Xi, Carnegie-Mellon University Chapter, Full Member (1983), Brown University Chapter, Associate Member (1977).

**Honors and awards:**

- University of Washington Distinguished Graduate Mentor Award (2000).
- IBM Faculty Development Award (1985).
- NSF Presidential Young Investigator Award (1988).

**Institutional and professional service in last five years:**

- Associate Chair (2000-2001).
- CSE142/143 Stewardship and Community College Liaison (1999-2000).
- University of Washington Tacoma, Computing and Software Systems, Faculty Recruiting Committee.
- University of Washington Bothell, Computing and Software Systems, Faculty Recruiting Committee.
- UW Educational Outreach Review Committee (1999).
- Board of Advisors, i5 Digital LLC (2000-).
- Chair, Organizing Committee for the Workshop on Software Engineering and Programming Languages, Cambridge MA (June 1996).
- Program Committee Chair, Co-chair, ICS2000: Software - Theory and Practice (part of
• Program Committee Chair, ACM SIGSOFT ’93 Symposium on Foundations of Software Engineering (December 1993).
• ACM SIGSOFT Chair (1997-2001).
• ACM SIGSOFT Secretary/Treasurer (1992-1997).

States in which registered: none
Level of activity: prof societies - high; research - high; consulting/summer work in industry - medium.
Perkins, Hal
Senior lecturer (1998 Lecturer, 2001 Senior Lecturer)

Education:
- B.S., Mathematics, Arizona State University, 1975
- M.S., Computer Science, Cornell University, 1982

Other Experience:

Scientific and professional societies of which a member:
- ACM (SIGCSE, SIGPLAN, others)
- IEEE & IEEE Computer Society

Honors and awards:
- 1998 Faculty of the Year Award, Cornell University Association of Computer Science Undergraduates

Institutional and professional service in last five years:
- August, 1999, Instructor, Computer Science Track, Microsoft Washington IT Educators Summer Institute, Bellevue Community College
- Panelist, Microsoft Washington IT Educators Summer Institute, Bellevue Community College, August, 2000
- Invited talk, WCERTE spring conference, 2001, WSU Pullman
- Department service: Undergraduate admissions (2000-present), CSE142/143 stewardship (1998-present), FIG Liason (1999-present), Faculty lunch coordination (2000-present)

States in which registered: none
Level of activity: prof societies - low; research - none; consulting/summer work in industry - none.
Popovic, Zoran
Assistant professor (1999 Ass't)

Education:
- B.S., Computer Science, Brown University, 1991
- M.S., Computer Science, Carnegie Mellon University, 1993
- Ph.D., Computer Science, Carnegie Mellon University, 1999

Other Experience:
- Computer Graphics Researcher
  Justsystem Pittsburgh Research Center (summer 1997)
- Research Associate
  University of California at Berkeley, Department of Integrative Biology (summer 1996)
- Computer Graphics Software Engineer

Principal publications of last five years:

Scientific and professional societies of which a member:
- ACM
- IEEE

Honors and awards:
- Schlumberger Research Fellowship, 1995-99
- Baxter-Travenol Scholarship 1990-91 (merit-based full-tuition scholarship)

Institutional and professional service in last five years:
- 2000 Graduate Student Admissions Committee
- 2000 Washington State Advanced Technology Initiative Center for Animation organizing committee
- 2000 SIGGRAPH'00 Course Organizer, Animating Humans by Combining Simulation and Motion Capture Organized a full day advanced level course on combining dynamics and motion capture for character animation
- 2000 SIGGRAPH'00 Course Speaker, Motion Editing: Principles, Practice and Promise Full day advanced level course on the current state of motion editing algorithms and how they are used in production
- 1999 SIGGRAPH'99 Course Speaker, Motion Editing: Principles, Practice and Promise Full day advanced level course on the current state of motion editing algorithms and how they are used in production.
States in which registered: none
Level of activity: prof societies - none; research - high; consulting/summer work in industry - none.
Rao, Rajesh
Assistant professor (2000 Ass't)

Education:
- Ph.D., Computer Science, University of Rochester, 1998
- M.S., Computer Science, University of Rochester, 1994
- B.S., Computer Science and Mathematics, Angelo State University, 1992

Other Experience:
- Teaching Assistant, Mathematics Department, Angelo State University, 1989-1992. Undergraduate courses on calculus and analytical geometry.
- Teaching Assistant, Physics Department, Angelo State University, 1989-1990. Undergraduate courses on fundamentals of physics.

Principal publications of last five years:

Scientific and professional societies of which a member:

• Society for Neuroscience

Honors and awards:

• Alfred P. Sloan Research Fellow, 2001-2003.
• Alfred P. Sloan Postdoctoral Fellow, Salk Institute for Biological Studies, 1997-2000.
• Computer Science Research and Teaching Assistantship, University of Rochester, 1992-1997.
• Presidential Fellowship for Graduate Studies, State University of New York, Buffalo, 1992 (declined in favor of Rochester).
• Robert and Nona Carr Academic Scholarship for undergraduate study, Angelo State University, 1988-1992.
• 1991 Who's Who among students in American Colleges and Universities.
• Alpha Chi (National Honor Scholarship), Epsilon Delta Pi (Computer Science), and Pi Mu Epsilon (Mathematics) 1991-1992.

States in which registered: none

Level of activity: prof societies - medium; research - high; consulting/summer work in industry - low.
Ruzzo, Walter L.
Professor (1977 Ass't, 1982 Assoc, 1990 Full)

Education:
- B.S., Mathematics, Caltech, 1968
- Ph.D., Computer Science, UC Berkeley, 1978

Other Experience:
- 5 years industrial experience, 1968-1973

Principal publications of last five years:

Institutional and professional service in last five years:
- 95--96: Executive Committee, Lab Policy, Publications, Development (ch), UW Bothell Curric & planning
• 96--97: UW Bothell CSS Search Committee, TA Assignments, Conflict of Interest Oversite (ch), COE Educational Policy,
• 97--98: UW Bothell Liason, Community College Liason, 142/143 Coordination (ch), Conflict of Interest Oversite (ch), COE Educational Policy (ch), COE ad hoc ENGR Restructuring, University Grant and Contract Process Enhancement Team, University (Search) Advisory Committee on the Dean and Vice Provost of UW Bothell, UW Bothell CSS Search committee (twice).
• 98--99: Lab Policy (ch), UW Bothell Liason, COE Educational Policy (ch), University Grant and Contract Process Enhancement Team, and chair of the associated Project Implementation Team, University (Search) Advisory Committee on the Dean and Vice Provost of UW Bothell, UW Bothell CSS Search committee.
• 00--01: Curriculum, 142-143 Stewardship, ACMS Program Liason (ch), CMB Program Liason (ch)

States in which registered: none
Level of activity: prof societies - none; research - high; consulting/summer work in industry - none.
Salesin, David
Professor (1992 Ass’t, 1996 Assoc, 2001 Full)

Education:
- Ph.D., Computer Science, Stanford University, 1991
- S.B., Computer Science, Brown University, 1983

Other Experience:

Consulting, patents, etc.:
- Method and system for virtual cinematography, with M. Cohen and L. He. Filed 1996.

Principal publications of last five years:
• Synthesizing realistic facial expressions from photographs, with F. Pighin, J. Hecker, D. Lischinski, and R. Szeliski. Proceedings of SIGGRAPH 98, in Computer Graphics Proceedings, Annual Conference Series, 75-84, July 1998. A video accompanying the paper was screened at the SIGGRAPH 98 Computer Animation Festival, Animation Theater program, July 1998; appears in the SIGGRAPH 98 Video Review; and was selected as one of 25 works for Art Futura 98 in Seville and Madrid, Spain, October 1998.
• Multiresolution video, with A. Finkelstein and C. Jacobs. Proceedings of SIGGRAPH 96, in Computer Graphics Proceedings, Annual Conference Series, 281-


Scientific and professional societies of which a member:

• Association for Computing Machinery
• Institute of Electrical and Electronics Engineers
• Eurographics

Honors and awards:

• The Carnegie Foundation for the Advancement of Teaching and the Council for the Advancement and Support of Education 1998-1999
• Washington Professor of the Year Award.
• University of Washington Distinguished Teaching Award, 1997.
• University of Washington Award for Outstanding Faculty Achievement in the College of Engineering, 1996.
• National Science Foundation Presidential Faculty Fellow Award, 1995-98.
• Alfred P. Sloan Research Fellowship, 1995-97.
• Office of Naval Research Young Investigator Award, 1995-98.
• National Science Foundation Young Investigator Award, 1993-95.
• Stanford University Nominee for ACM Dissertation Award, 1991.
• AT&T Graduate Fellowship, 1988-91.
• Stanford University School of Engineering Fellowship, 1986-87.
• Winston Churchill Fellowship Winner (declined, in order to work at Lucasfilm), 1983.
• Brown University Prize for Outstanding Undergraduate Research, 1983.
• Elected to Sigma Xi, 1983.
• Elected to Phi Beta Kappa, junior year, 1982.
• National Merit Scholarship, 1979-83.
• New York State Regents Scholarship, 1979.
• Rensselaer Polytechnic Institute Mathematics and Science Award, 1978.
• Bausch and Lomb Science Award, 1978.

Institutional and professional service in last five years:

• National Science Foundation, CAREER Awards Review Panel, 1999.
• Raster Imaging & Digital Typography Program Committee, 1998.
• Computers and Graphics editorial advisory board member, 1992-98.
• SIGGRAPH Technical Sketches Committee, 1996.
• Eurographics Technical Programme Committee, 1996.
• National Science Foundation Institutional Infrastructure Site Visit Committee, 1995.

States in which registered: none

Level of activity: prof societies - low; research - high; consulting/summer work in industry - high.
Seitz, Steven
Assistant professor (2000 Ass't)

Education:
- Ph.D., Computer Science, University of Wisconsin, 1997
- B.A., Computer Science, Mathematics, University of California, Berkeley, 1991

Other Experience:
- Assistant Professor of Robotics and Computer Science, Carnegie Mellon University, October 1997 - July 2000
- Adjunct Assistant Professor, Carnegie Mellon University, July 2000 – present
- Postdoctoral Researcher, Microsoft Research, October 1997 - August 1998.

Consulting, patents, etc.:

Principal publications of last five years:
- F. Dellaert, S. M. Seitz, C. E. Thorpe, and S. Thrun, EM, MCMC, and chain flipping for structure from motion with unknown correspondence, Machine Learning, to appear
- S. M. Seitz and C. R. Dyer, Cyclic motion analysis using the period trace, Motion-Based Recognition, M. Shah and R. Jain, eds., Kluwer, Boston, 1997
• S. M. Seitz and K. N. Kutulakos, Plenoptic image editing, Sixth International Conference on Computer Vision, 1998, pp. 17-24

Scientific and professional societies of which a member:
• IEEE

Honors and awards:
• NSF CAREER AWARD, May 2000
• DAVID MARR PRIZE, for the best paper at the International Conference on Computer Vision, September 1999
• BEST GRADUATE STUDENT RESEARCHER AWARD, U.W. Madison, May 1998

Institutional and professional service in last five years:
• University committees:
  Engineering Writing Assessment (University of Washington)
  Robotics Minor Committee (Carnegie Mellon University)
  Faculty Recruiting Committee (Carnegie Mellon University)
• CO-ORGANIZER, ACM SIGGRAPH Course on 3D Photography, 1999 and 2000
• CO-ORGANIZER, IEEE CVPR Course on 3D Photography, 1999
• PANELIST, NSF Panel on Symbolic, Numeric, and Geometric Computing; NSF Panel on Robotics and Human Augmentation

States in which registered: none
Level of activity: prof societies - low; research - high; consulting/summer work in industry - low.
Sengupta, Rimli
Research assistant professor (1999 Ass't)

Education:
- Ph.D., Computer Science, Georgia Tech, 1995
- M.S., Computer Science, Georgia Tech, 1993
- M. Tech., Electrical Engg., Indian Institute of Technology, Kanpur, India, 1989
- B. Engg., Electronics and Telecommunication Engg., Jadavpur University, Calcutta, India, 1987

Other Experience:
- 8/95 - 8/99: Assistant Professor, Computer Science, Rose-Hulman Institute of Technology, Terre Haute, Indiana.
- 6/97 - 8/97: Visiting Researcher, Computer Science, University of Texas, San Antonio.
- 9/94 - 3/95: Instructor, Georgia Tech.

Principal publications of last five years:
- R. Sengupta and H. Venkateswaran
- D. Mutchler, C. Anderson, C. Laxer, R. Sengupta, F. Young
- S. R. Das, R. Castaneda, R. Sengupta, J. Yan
- R. Sengupta and H. Venkateswaran
- R. Sengupta
- R. Sengupta and D. Mutchler
  "Design vs. Use: Do We Have to Choose?" Position paper for NSF Workshop on Future Directions of CS2, Duke University, North Carolina, 1997.
- R. Sengupta and H. Venkateswaran

Honors and awards:
- Volunteer of the month (for voluntary services to the adult literacy program at the county public library in Terre Haute, Indiana) March 1998.
Institutional and professional service in last five years:

- Institutional and professional service while at Rose-Hulman (1995 - 1999):
  Member, Commission on Institutional Assessment (1996 - 1998)
  Technical Communication faculty advisor
  Participant, Women in Math and Engg. Program (1996) designed to attract
  high school girls to Math, Science and Engg.
  President, Rose-Hulman chapter of Phi-Beta-Delta (1997 - 1998)

States in which registered: none
Level of activity: prof societies - low; research - high; consulting/summer work in industry - none.
Shapiro, Linda
Professor (1986 Ass't, 1981 Assoc, 1989 Full)

Education:
- Ph.D., Computer Science, University of Iowa, 1974
- M.S., Computer Science, University of Iowa, 1972
- B.S., Mathematics, University of Illinois, 1970

Other Experience:
- Director of Intelligent Systems, Machine Vision International,

Consulting, patents, etc.:
- Consulting for U.S. Dept of Education
- Consulting for CIA Office of Research and Development

Scientific and professional societies of which a member:
- Institute of Electrical and Electronics Engineers
- American Association for Artificial Intelligence
- Pattern Recognition Society

Honors and awards:
- Phi Beta Kappa, 1969
- Pattern Recognition Society Honorable Mention, 1985, 1987, and 1999
- Fellow of the IEEE, 1996
- Fellow of the IAPR, 2000

Institutional and professional service in last five years:
- Editorial Board Member, Pattern Recognition, beginning March 1986 and ongoing.
- Editorial Board Member, Computer Vision and Image Understanding, ongoing.
- Area Chair, Program Committee, Computer Vision and Pattern Recognition, 1998
- Director, Computer Systems Group, Washington Technology Center, 1992-98
  Co-Chair, ICPR2002 Track on Biomedical and Multimedia Applications

States in which registered: none
Level of activity: prof societies - medium; research - high; consulting/summer work in industry - low.
Snyder, Lawrence
Professor (1983 Ass't)

Education:
- B.A., Mathematics and Economics, University of Iowa, 1968
- Ph.D., Computer Science, Carnegie Mellon University, 1973

Other Experience:
- Assistant/Associate Professor, Yale University, 1973-1980
- Associate/Professor, Purdue, 1980-1983
- Visiting Scholar, University of Washington, 1979-80
- Visiting Scholar, MIT and Harvard, 1987-88
- Visiting Professor, University of Sydney, 1994-95

Consulting, patents, etc.:

Principal publications of last five years:
- Being Fluent With Information Technology, National Academy Press, 1999
- Integrated Multi-class Routing, with M. Fulgham, 2nd Parallel Computer Routing and Communication Workshop, Springer Verlag, 1997
- Quantifying the Effect5s of Communicatoin Optimizations, with S-E Choi, Int'l Conference on Parallel Processing, pp. 218-222, 1997
• ZPL's WYSIWYG Performance Model, with B. Chamberlain, S-E Choi, E Lewis, C. Lin
  and W. Weathersby, IEEE Workshop on High Level Parallel Programming Models,
  pp. 50-61, 1998
• The Implementation and Evaluation of Fusion and Contraction in Array Languages,
  with E. Lewis and C. Lin, ACM Programming Languages and Implementation
  Symposium
  pp. 50-59, 1998
• The Case for High-Level Parallel Programming in ZPL, with B. Chamberlain, S-E
  Choi, E. Lewis, C. Lin and W. Weathersby, IEEE Computational Science and
• Problem Space Promotion and Its Evaluation as a Technique for Efficient Parallel
  Computation, with B. Chamberlain and E. Lewis, 13th Int'l Conference on
  Supercomputing, pp. 311-318, 1999
• Regions: An Abstraction for Expressing Array Computation, with B. Chamberlain,
  E. Lewis and C. Lin, ACM International Conference on Array Languages,
  pp. 41-49, 1999
• Array Language Support for Wavefront and Pipelined Computations, with B.
  Chamberlain and E. Lewis, Workshop on Languages and Compilers for Parallel
  Computing, 1999
• ZPL: A Machine Independent Language for Parallel Computers, with B.
  Chamberlain,
  S-E Choi, E. Lewis, C. Lin and W. Weathersby, IEEE Transactions on Software
  Engineering, 2000 (to appear)
• Pipelining Wavefront Computations: Experiences and Performance, with E. Lewis,
  IEEE Workshop on High Level Parallel Programming Models, pp. 261-268, 2000

Scientific and professional societies of which a member:
• ACM -- Association for Computing Machinery
• IEEE -- Institute for Electrical and Electronic Engineers
• SIAM -- Society of Industrial and Applied Mathematicians

Honors and awards:
• Fellow of the ACM
• Fellow of the IEEE

Institutional and professional service in last five years:
• Board Member, National Research Council's Army Research Laboratory Technical
  Assessment Board, 1996 -- Chair of Digitization Panel, 1999 –
• Organizer, NSF Workshop on Experimental Research in Computer Science, 1996
  Board Member, Computer Research Association, 1996 –
• Chair, CSTB Committee on Computer Literacy, National Research Council,
  1997 – 1999
• Member, CISE Panel on Future Directions in Experimental Computer Science, 1998

States in which registered: none
Level of activity: prof societies - medium; research - high; consulting/summer work in industry - low.
Suciu, Dan
Assistant professor (2000 Ass't)

Education:
• M.S., Computer Science, Politechnic of Bucharest, Romania, 1982
• M.S., Mathematics, University of Bucharest, 1991
• Ph.D., University of Pennsylvania, 1995

Other Experience:
• AT&T Labs, 1995-2000

Consulting, patents, etc.:
• US Patents:
  6,134,553 Method for using region-sets to focus searches in hierarchical structures
  6,076,087 Query evaluation on distributed semi-structured data
  6,052,686 Database processing using schemas
  5,999,926 View maintenance for unstructured databases
  5,987,449 Queries on distributed unstructured databases
  5,978,790 Method and apparatus for restructuring data in semi-structured databases
  5,970,489 Method for using region-sets to focus searches in hierarchical structures
  5,960,425 Database access system with optimizable expressions
  5,956,720 Method and apparatus for web site management

Principal publications of last five years:
• Serge Abiteboul, Peter Buneman, Dan Suciu, "Data on the Web: from Relations to Semistructured Data and XML", Morgan Kaufmann, 1999, ISBN 1-55860-622-X.
• Alin Deutsch, Mary Fernandez, Dan Suciu, "Storing semistructured data with STORED", ACM SIGMOD Conference on Managment of Data (SIGMOD), 1999, pages 431-442.
• Tova Milo, Dan Suciu, "Index structures for path expressions", International Conference on Database Theory (ICDT), 1999, pages 277-295.

Scientific and professional societies of which a member:
• Member of ACM

Honors and awards:
• Best paper award at the ACM/SIGMOD International Conference on Management of Data, 2000
• Sloan Fellow, 2001

Institutional and professional service in last five years:
• IEEE International Conference on Data Engineering (ICDE), 2001. (vice chair)
• International World Wide Web Conference (WWW10), 2001 (deputy vice chair)
• ACM SIGMOD International Conference on Management of Data (SIGMOD), 2001
• International Conference on Database Theory (ICDT), 1999 and 2001
• ACM International Conference on Information and Knowledge Management (CIKM), 2000
• Deductive and Object-oriented Databases (DOOD), 2000
• International Conference on Very Large Data Bases (VLDB), 2000
• Conference on Extending Database Technology (EDBT), 1999
• ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), 1998
• Dagstuhl Seminar on Semistructured Data, 2001 (Co-organizer)
• WebDB 2000 (Program Co-chair)
• DIMACS Workshop on Data Processing on the Web: A Look into the Future, 2000 (Co-organizer)
• Query Processing for Semistructured Data and Non-Standard Data Formats, 1999 (Program Co-chair)
• Workshop on Management of Semistructured Data (Program Co-chair)

States in which registered: none
Level of activity: prof societies - low; research - high; consulting/summer work in industry - medium.
Tanimoto, Steven
Professor (1977 Ass't, 1981 Assoc, 1987 Full)

Education:
- Ph.D., Electrical Engineering, Princeton University, 1975
- A.B., Visual and Environmental Studies, Harvard University, 1971
- M.A., Electrical Engineering, Princeton University, 1974
- M.S.E.E., Electrical Engineering, Princeton University, 1973

Other Experience:
- Assistant Professor, Univ. of Connecticut, 1975-1977

Consulting, patents, etc.:

Principal publications of last five years:
- Tanimoto, S.~L. and Bernardelli, C.~E. 1998. The design and implementation of Vedo-Vedi, a visual language for human communication in the Internet.
  Proc. Int'l Conf. on Pattern Recognition, Brisbane, Australia.
  Technical report 97-04-03, Dept. of Computer Science and Engineering, Univ. of Washington, April.

Scientific and professional societies of which a member:
• IEEE Computer Society
• Association for Computing Machinery
• National Council of Teachers of Mathematics.

Honors and awards:
• IEEE Fellow.
• IEEE Computer Society Outstanding Contribution Award.

Institutional and professional service in last five years:
• Faculty Council on University Libraries, 1999-present.
• Search committee for Director of Libraries, 2000.
• International Association for Pattern Recognition, Education Committee Chair, 1998-2000.
• International Symposium on Visual Languages, Co-General Chair, 2000; Steering Committee, Program Committee.
• NSF IGERT program at the University at Buffalo on Geographical Information Science: Advisory Board member.

States in which registered: none

Level of activity: prof societies - medium; research - high; consulting/summer work in industry - high.
Tompa, Martin  
Professor (1978 Ass't, 1984 Assoc, 1989 Full)

Education:
- A.B., Applied Mathematics, Harvard University, 1974
- M.S., Computer Science, University of Toronto, 1975
- Ph.D., Computer Science, University of Toronto, 1978

Other Experience:
- 1985-87: Research Staff Member, Theory of Computation, IBM Research Division, Thomas J. Watson Research Center.

Consulting, patents, etc.:
- 1980: Consultant to Boeing Aerospace Company on VLSI design.
- 1999: Consultant to Rosetta Inpharmatics.

Principal publications of last five years:
- "A Linear Time Algorithm for Finding All Maximal Scoring Subsequences" (with W. L. Ruzzo), Seventh International Conference on Intelligent Systems for Molecular Biology, Heidelberg, Germany, August 1999, 234-241.
- "An Exact Algorithm to Identify Motifs in Orthologous Sequences from Multiple Species" (with M. Blanchette and B. Schwikowski), Eighth International Conference on Intelligent Systems for Molecular Biology, San Diego, CA, August 2000, 37-45.


Honors and awards:

- 1984-86: Presidential Young Investigator Award
- 1998: ACM Undergraduate Teaching Award
- 1999: ACM Undergraduate Teaching Award

Institutional and professional service in last five years:

- Program Committee chair, 37th Annual Symposium on Foundations of Computer Science, Burlington, Vermont, October 1996.
- Site visit committee member, National Institutes of Health proposed Center for Computational Biology and Bioinformatics, University of Pittsburgh, July 1999.
- University of Washington Computational and Mathematical Biology steering committee, 1999-present.

States in which registered: none
Level of activity: prof societies - low; research - high; consulting/summer work in industry - high.
Weld, Daniel  
Professor (1988 Ass't, 1993 Assoc, 1997 Full)

Education:

- Ph.D., Artificial Intelligence, Massachusetts Institute of Technology, 1988
- M.S., Computer Science, Massachusetts Institute of Technology, 1984
- B.S., Computer Science, Yale University, 1982
- B.A., Molecular Biophysics & Biochemistry, Yale University, 1982

Other Experience:

- Founder, Nimble.com (1999)
- Founder, AdRelevance Inc. (1998)
- Consultant, Excite Inc. (1998)
- Founder, Netbot Inc. (1996) Creator of Jango shopping search

Consulting, patents, etc.:

- Patents Pending:
  Method and System for Accessing an On-Line Store, Application 09/008,413
  Doorenbos, Etzioni, Weld
  Method and System for Network Information Access, Application 60/035,844
  Friedman, Weld, Kwok
  Method and System for Network Information Access, Application 9032-003 Selberg et al.
  Method and Apparatus for use in Accessing On-line Semi-structured Information,
  Application 08/982,857 Kushmerick, Doorenbos, Weld - filed 2 Dec 97

Principal publications of last five years:


Scientific and professional societies of which a member:
• American Association for Artificial Intelligence
• Association for Computing Machinery
• Institute of Electrical and Electronics Engineers

Honors and awards:
• Fellow, American Association of Artificial Intelligence, 1999
• Edge Award for best use of intelligent technology (in Jango), WebINNOVATION Show, 1997
• New Innovator's Award (for Jango), CommerceNet, 1997
• Award for Technological Innovation in Computer Software (Internet Softbot one of five finalists), Discover Magazine, 1995
• Young Investigator Award, Office of Naval Research, 1990
• Presidential Young Investigator Award, National Science Foundation, 1989
• John E. Bierwirth Scholarship, Yale University, 1979—82
• Wilfred Freeman Fellowship, Phillips Academy, 1978

**Institutional and professional service in last five years:**

- **Advisory & Review Boards:**
  - Member of Advisory Board, Journal of Artificial Intelligence Research (1992--)
  - Member and Editor, AAAI/NSF Committee on Intelligence in the NII (1994)
  - Member, AAAI/ARPA Committee on Twenty-First Century Intelligent Systems (1994)

- **Editorial Activities:**
  - Editorial Board Member, Artificial Intelligence (1999--)
  - Guest Editor, Artificial Intelligence special issue on Intelligent Internet Systems (1998)
  - Associate Editor, Journal of Artificial Intelligence Research (1993--1996)
  - Guest Editor, Computational Intelligence special issue on qualitative reasoning (May 1992)
  - Editorial Board of AI Communications (AICOM) declined

- **Program Committee Chair:**
  - AAAI (1996)
  - International Workshop on Qualitative Reasoning (1993)

- **Program Committee Member**
  - AGENTS (1998, Area Chair for Software Agents)
  - KR (1998 (declined))
  - International Workshop on Principles of Diagnosis (1998, declined)
  - Pacific Rim International Conference of Artificial Intelligence, (1998, declined)
  - AAAI Workshop on Model-Based Reasoning (1990, 1991)
  - Conference on Artificial Intelligence and Symbolic Mathematical Computation, Karlsruhe (1992)

**States in which registered:** none

**Level of activity:** prof societies - high; research - high; consulting/summer work in industry - high.
Wetherall, David
Assistant professor (1999 Ass't)

Education:
- B.E., Electrical Engineering, University of Western Australia, 1989
- M.S., Computer Science, Massachusetts Institute of Technology, 1994
- E.E., Computer Science, Massachusetts Institute of Technology, 1995
- Ph.D., Computer Science, Massachusetts Institute of Technology, 1998

Principal publications of last five years:

Scientific and professional societies of which a member:
- IEEE
- ACM

Institutional and professional service in last five years:
- Program Co-Chair of IEEE OpenArch, 2001.
- Graduate Admissions Committee, 2000.
- Member of the SIGCOMM Program Committee, 2000.
- Member of the SIGCOMM Program Committee, 1999.
- Member of the 1st IEEE Workshop on Internet Applications Program Committee, 1999.

States in which registered: none
Level of activity: prof societies - none; research - high; consulting/summer work in industry - none.
Zahorjan, John
Professor (1980 Ass’t, 1985 Assoc, 1989 Full)

Education:

- Ph.D., Computer Science, University of Toronto, 1980
- M.Sc., Computer Science, University of Toronto, 1976
- Sc.B., Applied Mathematics, Brown University, 1975

Principal publications of last five years:

- Derek Eager, Mary Vernon, and John Zahorjan, "Minimizing Bandwidth Requirements for On-Demand Data Delivery", to appear in IEEE Trans. on Knowledge and Data Engineering, Special Section of invited papers from MIS'99, Sep. 2001.
Institutional and professional service in last five years:

- Program Committee Member, 18th ACM Symposium on Operating Systems Principles (2001).
- NSF Grant Review Panels, 2000
- Member of Board, Cleveland InfoTech Infusion Academy, 2000-.
- Webmaster, ACM SIGMETRICS, 1999-
- Program Committee Member, 2nd Symposium on Operating Systems Design and Implementation, October 1996.
- Program Committee Member, Workshop on Scheduling Strategies for Parallel Computing, May 1996.
- General Chair, 8th IEEE Symposium on Parallel and Distributed Processing, October 1996.

States in which registered: none

Level of activity: prof societies - low; research - medium; consulting/summer work in industry - low.
D. Student exit survey comments for the past two years

Below are collected comments from our exit survey questions: **Do you have comments or suggestions to the department to better serve our students?** and **What courses would you like to see offered in the CSE department?**

All comments are carefully reviewed each quarter by the Undergraduate Advising Unit and the Undergraduate Faculty Advisor. Exiting students have a particularly valuable perspective on the program because they have been through it all and have had the chance to evaluate for themselves how it fits together. In addition, a large percentage of these students have also worked in industry (approx. 50% and rising) and have gained an even larger point of view. Suggestions are evaluated and either acted upon directly or forwarded to the Curriculum Committee for further discussion and resolution. Of course, not all suggestions are implementable or may run counter to the Department’s goals. However, these issues are brought up at subsequent student lunches so that all aware of the rationale for implementing or not implementing them.

- The quarter system sucks.
- I might suggest to add more flexibility offered to students in the computer engineering program, but I feel like the new program offers a lot more of that. Overall I've been extremely impressed with the program: great professors, interesting classes, smart & nice students, and lots of opportunities to get involved in research!
- **Unix course of how to operate and manage the unix system for new entries should be helpful** (ACM Unix tutorial) 1999 graduate
- More hardware design courses. (1999 grad)
- yes. Make students who wish to take cse457, take a computational geometry course first, so that one can better handle the math in graphics. If possible, increase the amount of disk space for CS accounts. Do not change the number of credits of a course during the middle of the quarter. This causes a myriad of annoying events, which will easily be overlooked by the busy CS student. Raise the mathematical standards of the incoming CS students. Perhaps, less chemistry prequisites as a result. (1999)
- I would like to see animation offered more often. (1999 grad, so since have added more courses)
- EE 332 should not be a mandatory class for Computer Engineering students. EE 332 is an analog class. There should be a few more hardware design classes and a few less theory/programming classes for computer engineers. (1999)
- Department should try to keep the planning schedule as best as it can so students can easily plan for their schedule every quarter. 2. Core classes should be offered at least twice a year. This helps students don't have to wait a year or more to take one class for graduation. This problem will delay students' graduation. For instance, CSE471 is offered only in Autumn and is delayed to Winter this year. Many students have to take substitute classes for graduation, and they will miss the chance to learn about computer design. (We do need better course planning …) 1999
• Keep CSE451 using Unix. Have more X terminals. Keep using SGI machines for at least some Graphics course to get students familiar with more than one OS. Organize some sort of Alumni CS Dept club so that students can network with one another. Also perhaps connect these Alumni groups with Business-related entrepreneurial groups to encourage commercial tech-related ventures post-graduation. This would elevate the University’s profile and allow a synergy similar to that of Stanford and Silicon Valley. 1999

• Lab space is tight and not all that well maintained. Information isn't distributed well... i.e. cs-ugrads does NOT get to ce students (mailing lists corrected to include both student groups in cs-ugrads) 1999

• Hmm. The department at a whole is excellent. I really like the push to diversify it in the recent years. I would just plan more social activities for undergrads to get together.

• More space. I know you all know this, but... Dedicated labs for classes would be fabulous. In 468 last quarter having a dedicated lab was great! The software never got screwed up!

• I wanted to take a language, but because I was in the engineering school, and the first year doesn't count, I really couldn't. That was a bit disappointing.

• I think some more classes should be made prerequisites for others. 370 should be required before taking 378, and 421 a prerequisite for 431. I think this sequence makes sense and would keep the instructor from spending time bringing part of the class up to speed on topics other students have already covered. I'm sure there are some other courses that would benefit from sequencing. I think the length of a quarter is a real disadvantage, especially for outer core courses with a large project component. If some of these courses could run over two quarters, then more significant projects could be tackled. I found it a little frustrating that once a project really got going the quarter would be over. Perhaps some of the capstone courses could be extended, if not outer core courses. I wish some of the outer core and capstone courses would be offered more frequently. There were some courses I wanted to take, but couldn't because they just didn't fit into my schedule or sometimes a course would get bumped to a different quarter. I think most cs students spend more time in the department than I have, so this may not be an issue with the majority of students. The past two quarters Prof. Borning has taught the Computers, Ethics, and Society course. While I did not take this course, I think it should be made a requirement. It was my impression that many of my peers do not give these important issues enough thought and they very rarely came up in cse courses I took. (2000 graduate)

• An honors introductory programming course that fulfills a university science requirement. You could encapsulate 142,143,326, and 341 into a three quarter long sequence. Not only would this free up a quarter for accelerated students, but attract top university honor students to the CSE department. (2000)

• When I arrived at the University in Fall 1995, I hadn't used a computer for anything more than word processing and gaming (Atari and Nintendo). Now, five years later, I have a firm grasp on what computers are, what they can do, and the major issues in efficient development and improvement of them. Truthfully, I cannot think of a computer science project where, maybe after some necessary introduction, I could not impressively contribute. Thanks. (2000)

• Without this research (Graphics), I wouldn't have gotten into grad school. I seriously urge every student who's considering grad school to do what I did as soon as possible. (2000)
- The classes provide pretty good coverage so far. Maybe a computer system's design course (eg, motherboards). 468 and 471 provided a good background on how a processor works, but there was no class that showed how the processor interfaces with the rest of the motherboard. (2000)

- It is a fundamental error on the part of the CSE department to not offer a course in the Ethics of Research and Development in Computing Technologies. A course should also be offered at the undergraduate level for HCI.

- Career counselor would be nice. Current academic counselors are awesome.

- It would be better if the CSE dept. would teach their own EE courses. Especially now that they are essentially only 2 required. They could gear them better toward computer applications.

- I was a little disappointed that my distribution of classes did not cover the 8 credits of Engr., Math, Science Electives. I think that one needs to be explained a little bit better; and also set apart from the Comp E. Electives better. 451, 467, and 477 were INCREDIBLE classes. I learned so incredibly much from them, and like my earlier response, I loved the hands-on nature of the courses. CSE471, while informative, only held my interest on the project half of class. The Entire final was not based at all on the project, but instead on only the lecture material f the last few weeks. So I was a little disappointed. The EE half of things was also very informative. I really loved to learn everything that 215, 233, 321, and 322 had to offer. However, the excessive lab-work, and the concentration of accuracy over learning really distracted me, since I feel that my University experience should be their for the learning, not as much for the applied. (The industry is so diferent, regardless of all the "group activities" that classes attempt.)

- 2 Quarter Course -- Software engineering joint some other department (eg. Business). 10 weeks just doesnt seem like enough time to get through the complete development cycle.

- There should be at least one class which focuses on real world programming. IE - how things work in the real world -- and I don't mean cse 403.

- security; a __REAL__ computation linguistics class, not Ling 472.

- more web/internet courses (have since added Internet capstone)

- I think the courses currently offered are fine. However, I would like to see less cancelations...they are really annoying.

- More classes that invovle interaction between CS and other fields - such as a biology or human research program or class.

- Oh, I 'unno. Being third of seven co-authors on a paper published at this year's SIGGraph is kinda sweet. :) Now if I could only get the UW to pay for me to go... ;) Seriously, research is interesting. I learned I'm a better software engineer than researcher, but it's been very educational all the same. Being on a very loose schedule without any sort of "credits" or formal plan has been a boon as well, since I've tended to get lost in a lot of other projects as well.

- Don't expand. ^_^;
  Expansion baaaad. Staying small lets one stay elite. Expanding large means less quality education for people, and lower caliber students to get that education.
• Java programming

• I would like to see more networking classes. I would also like to see more in depth studies on the I/O subsystem of an Operating System.

• Provide more computers. Expanding the department has put a crunch on the number of computers available in the prime time hours. It is not always possible to rework your schedule to avoid the busy hours.

• Better courses on Networks, especially as it relates to the Internet.

• Get rid of NACHOS for CSE 451-- I would have much rather started at the ground and learned the basics in a coding environment, while learning about advanced topics in a higher level.

• A course on Windows Programming / MFC. It seems like a lot of courses tried to stay away from Windows-centric programming, which is a real shame considering the majority of companies out there are Windows shops.

• I really wanted to take a robotics class, but the EE class always conflicted with my other (required) CS classes. I think there is enough of an interest to cover robotics in a CS/CE class (or have better synchronization between time schedules).

• Break up the senior design capstone class (cse 477) so that there aren't any labs in 477- they are covered in the pre-req classes. 10 weeks isn't enough time to do a project, let alone doing labs as well.

• I think this has gotten better, but when I took cse 341, we studied languages that I will probably never see again. Now they are talking about other languages that I have used in my internship (Perl, XML) that would have been great to have exposure to in 341. So, I guess the lesson is to make sure that the content of the classes is up to date.

• I was a comp-E major for 2 years. I would only recommend the program to somebody who really, REALLY, likes hardware and bad EE professors. I would almost suggest working a deal with the various departments to have people only apply for CS, and, if they enjoy 370/378 and EE 215 then to apply for the CE degree program.

• How about courses that are 5 credits instead of 3? These courses and the amount of homework for even the basic 3 credit courses puts the average "hard" 5 credit course to shame from most other departments in this school. People basically kill themselves in this department taking 4 or 5 classes just to make full time.

• More crossover classes with different fields (or at least advertise them more)

• I wish I would have known EE332 would not be required back in September of 1999. I believe, better communication with students is needed.

• I really didn't enjoy many of the inner core computer science classes. They were too focused on theory and mathmatics. Also the pace was way faster than I felt was reasonable for a solid understanding of all of the interesting concepts.

• I was denied admission to both of the upper level CS classes that were the pinnacle of my focuses. I was really excited about these two capstone classes, and had worked up all of the pre-recs for. I would suggest that you send people interested in graphics or games to another university.
• Natural Language Processing - a practical rather than the survey course currently offered through the Ling Department.
System Management - a practical course that would allow students to experiment with setting up networks, user profiles, system settings, scripting etc.

• I really feel like I did not get enough practical and coding experience while in the department. The theoretical foundation was very good but the projects were too "well" guided to be long-term beneficial. I guess I would like to see more hands-off projects that allow students to experiment more

• a more advanced graphics course

• My research experience was much better than the internship one.
I worked on two truly exciting projects, and had a chance to actually design specific components of these projects. Professors' help and support was extraordinary.

• Career counselor would be helpful. Other than that you guys KICK ASS!!!

• Embedded systems, but it looks like that's what's happening now.

• Could have used some web programming in inner core, in preparation for outer core.
Software Engineering with David Notkin was a super class.

• Add some web programming to inner core

• I would like to see greater coordination between the mathematics and computer science departments. The ACMS program is not yet an acceptable substitute, as there are a great many repetitive and wasted classes when taking in conjunction with a CS degree

• Move away from drill-oriented curriculum. I dare say that those for whom Computer Science is new are unprepared to promote the proper goals of computing.

• a second course in data structures. (at the 400 level)

• I really *really* like the attempts made here to be as language and platform neutral as possible. Instead of teaching a technology that is current now and will be out of date in three years, we learn the theories and ideas that are timeless. This more than anything is what is valuable about the cse degree. I can pick up a book and learn a new language or technology, but the skills I have gained in the cse dept here are much deeper and more fundamental than that.

• Classes inside the department were noticeably better than those outside. (with the exception of VLPA courses perhaps.) Physics would get a 5 if it weren't for the first year series I took. (Honors Physics - normal physics would have been better if it meant getting a better professor!) Math would get a 5 too if it weren't for Math 324 which I just didn't get along with. (Why wasn't hw collected again?) CSE 341 is a weak point IMHO.

• Everything here is great expect the EE classes. Especially EE 215. The professor involved had no business teaching that course. EE 233 was noticeably better (at least the professor had a clue) but still could have benefited from significant reorganization.

• The biggest problem with the current curriculum is a lack of structure. What I mean here is that there isn't enough of a progression in classes to allow for good buildup of the material. I can't tell
you how many times I've seen a professor look around and say "How many people have taken xxx" and half the class raises their hands. In that case, the prof usually caters to the most general audience, which means that fewer topics can be discussed. I think some of this could be alleviated if CSE took some ideas from other departments, which offer series of courses. (see Math, Physics, Music...) That way, at the end of the series, the professor can rely on what was taught all year long. One idea would be to join up 378, Compilers, and OS together into a year-long series. I think the progression from assembly language to compilers is very natural, and the concepts learned in the first quarter of 378 would come back in a very useful way to make for a solid OS foundation in the third quarter. At the end of that series, students would know how the whole thing works from the ground up. I think that would be a very satisfying experience. Personally, I feel those three should be required classes anyway...

Another series could be theoretical. 322, 326, 421. Again, my personal belief is that these should be mandatory. In my case, I took 326 from Professor Tampa. Needless to say, we covered about half the topics of 421! That's really another part of the problem- professors teach the classes differently. That's nice because it gives variety, and lets the professor tailor the class to his/her strong points, but it makes for class interoperability problems. A case in point is Dijkstra's Algorithm. I think I got a thorough introduction to this in no less than 4 classes! It's a nice algo and all, but it got old after a while. That's mostly because professors thought it was cool, so if they had some topic that was close-by, they'd take a little detour. And I'm sure that some people in each one of those four classes were seeing it for the first time! If there were a series, it would only be shown in detail once. Again, I think the similarities in material would allow for some creative teaching opportunities in the series, allowing more topics to be covered, or in more detail, in the same amount of time.

Series only really make sense for these sorts of core classes. I don't see Networks fitting into a series too well, but then it's also relatively well contained. Perhaps two series would be enough. The other thing to think about would be the hardware side. 370/467 really should go together. My 467 professor spent a couple weeks just brushing up on 370 stuff! Also note that the series doesn't HAVE to be taught by the same professor. Someone else could take over in the middle, as long as there was a clear understanding of what was covered in the previous lectures. It may be beneficial to have the later professors attend the earlier classes. (this not only lets the professor become acquainted with the class, but also track their progress very closely.)

My other rant is that some classes really should be given a few more credits. Most notably CSE 490ca. In reality, this class is like a very intense full-time job. It's hard to sign up and only take 5 credits!

The largest waste of my time at the U was the technical communications courses. I honestly believe they have reduced my ability to write and speak effectively. EE was painful, but at least in the end it helped me understand something new. I think the idea to have a technical communications course is a great one- those are very important skills which most people could use a little help with (myself included.) I'd recommend having the class moved into the department, though, as the teachers who teach it now have absolutely no clue what they are doing.

Anyway, take everything I've just said with a grain of salt. I had a great time in this department, and I learned so very much from many wonderful professors. I really feel we have a solid program, and hope to see it continue to improve!

- I think that getting more exposure to the students about the research available in the department will be very helpful because they can apply what they get in the classes to what technology exists in the industry.
- Drop more EE requirements for Comp. E. Allow current Comp E students to switch tracks so they
don't have to suffer what I suffered.

- 143 did not prepare me for the programming I needed to do in the department! I had a terrible time in 326 because of this. There should be an additional possibly optional course for people who don't have a programming background to take after 143 that would bridge the gap between 143 and the departmental classes. I know I didn't get enough from 143 because I got a 3.9 in it, but then couldn't do the programming required for the homework in 326. Also, people should be encouraged to do a co-op earlier (maybe right after 143) so they have some more software development experience before taking departmental classes.

- Have faculty advisors and possibly graduate student advisors. I needed more advice from faculty that the advisors couldn't really provide since they aren't professors! The advisors were great for what they provide, I just needed more mentoring from someone in the field. Also, career advising just for our field! We are in a unique situation. It is frustrating not to have anyone to talk to about our job market, interview styles, and job offers. The people in career services are of only limited help because they don't specialize in our field.

- Computer Networks, Software Engineering, Embedded Systems (which are already offered at the undergraduate level), but I'm not sure at the graduate level. When I do my master's I would like to specialize in these areas.

- I LOVED the CSE department -- all courses and professors included. I like it just the way it is. Perhaps the only improvement would be to allow more research involvement opportunities at the undergraduate level.

- Human-Computer Interaction
  Windows programming (other schools do it, like Cornell)
  Advanced C++ and Java programming (Stanford does it)
  Classes on game logic (CMU has one)

- Should be more opportunities for tutoring and mentoring

- I would like to see more courses about real world application of CS. For example, I took cse590ip, software entrepreneurship, and I found it very educational.

- Having CSE461 offered jointly with EE dept. does a great disservice to both the CSE people and the EE people. The CSE people are disappointed because the EE people want it easier with less programming assignments because they don't know how to very well. And the EE people think it is too hard and irrelevant for them to be doing programming assignments as EE majors. Both sides have a good point, and it really makes the class suck for half the class, and which half gets ripped off obviously has to do with which department is teaching it. I know a lot of people want to take it when their respective department teaches it, but it is hard to do that when STAR only lets half of the class be that major. (pre-requisites adjusted twice in last two years)

- Having most of the CS-oriented 400-level classes as 3 credits is hard, because:
  (a) you do way more work than a 3 credit class coupled with
  (b) it makes is incredibly hard to put together a schedule... if I take 2 400-level CSE classes.. I 5 credit 'other' class.. that's only 11 credits and still not full time!
  .. and taking 4 classes with a schedule like that is terrible, since there are pretty much no 1-2 credit classes. These two are a frustrating combination often times.
And now something positive:
With all the noise on campus about tolerance of diversity, I think everyone in CSE is doing great. I've never felt treated badly/different because of my gender, and the atmosphere has been nothing but positive. Thanks!

- For the senior electives, I liked how there was flexibility so you could choose electives that were interesting to you. The inner and outer classes were all really good. All the professors were awesome.

- Networks should be an outer core course (done in spring 2001)

- Also, 466 is a class with good intentions that doesn't serve anyone well, they offer it to comp sci, and comp e. but then try to teach about single transistor amplifiers etc. this is frustrating to comp e. students who have spent several quarters studying transistor amplifiers etc. and bewildering to comp. sci students who have never seen transistors at all, and dont know even basic things about electronics. It should either really be software, or really be hardware, and if its hardware, it should have pre-reqs. The department overall does a good job, there are a lot of opportunities, and more every year, I think that in the 4 years since I have been here it has become enormously easier to get involved with research, and they are doing a better job of informing people of all the opportunities that exist (they could still do better in the future, advertising interesting courses like DSP classes etc. over the mailing list, like the college honors program does). When I first got here I couldn't find any research to get involved with, even at the lowest level, now its easy.

- The department has admitted far too many students without expanding class schedule and size. I was not seriously affected, but frustration is growing very high among those graduating after me. Overall my education was great. We have a world-class faculty, and by far most of them are genuinely interested in undergraduate education.

- Encourage more women to apply ... also have more female teachers, are there any?

- More hardware classes building up to the capstone courses. More than one VLSI class

- Have more classes. Have more blending with Business. SBE is a great thing.

- I'd like to see Internet Systems or something similar become a real course instead of just a Special Topics class.

- More pure software engineering courses (2001 grad)

- I think EE courses targeted at CSE students would be much better and more effective for Computer Engineering majors. Many topics are not covered in a way that relates to the future work of CSE majors. Also, some topics, such as powerlines, are covered that have little or no application to CSE majors. I am glad to see that a CSE-specific Stat 390 class has been created. I really wish I had taken that instead of normal Stat 390.

- I think we definitely need more lab and meeting space. It gets difficult to do any work towards the end of the quarter when things start getting crowded.