Program Self-Study Report for
Computer Science and Engineering

Contents

A. Background Information 1

B. Accreditation Summary 3

- B.1 Students 3
  - B.1.1 Quality of Students and Graduates 3
  - B.1.2 Evaluating, Advising, and Monitoring Students 3
  - B.1.3 Meeting Program Requirements 4
  - B.1.4 Transfer Credits 5
- B.2 Program Educational Objectives 6
  - B.2.1 Educational Objectives 6
  - B.2.2 Process to Determine and Evaluate Educational Objectives 7
  - B.2.3 Curriculum and Processes to Ensure Achievement of Objectives 9
  - B.2.4 Evaluation of and Improvements in the Program 12
- B.3 Program Outcomes and Assessment 15
  - B.3.1 Program Outcomes 15
  - B.3.2 The Curriculum and Criterion 3 Outcomes 16
  - B.3.3 Assement Process 18
  - B.3.4 Materials for Review During On-Site Visit 21
- B.4 Professional Component 22
  - B.4.1 Culminating Design Experience 22
  - B.4.2 Mathematics, Basic Sciences, Engineering Topics, & General Education 27
- B.5 Faculty 29
- B.6 Facilities 30
- B.7 Institutional Support and Financial Resources 31
- B.8 Program Criteria 32

Appendix I – Additional Program Information

Appendix II – Institutional Profile

Program Self-Study Report for Computer Science and Engineering

A. Background Information

A.1 Degree Titles

Bachelor of Science in *Computer Science and Engineering* (*BS* in *CSE*)[1]

A.2 Program Modes

Conventional day classes.

A.3 Actions to Correct Previous Deficiencies

The program has not been previously evaluated by the *EAC*.

A.4 Major Developments During the Last Few Years

This section, although not required by the self-study instructions, is included in order to briefly summarize some recent major developments in the Computer Science and Engineering *(CSE)* program and in the Computer and Information Science (CIS) Department.

A.4.1 Courses

The first year sequence for *CSE* majors has been revised recently to further stress *software engineering* principles. While the previous version did pay considerable attention to software engineering issues, they were not fully integrated with each other. We have also taken this opportunity to move these courses from *Modula-2* to a disciplined version of *C++*, a version that encourages students to pay attention to designing reliable systems rather than just hacking together something that 'works'. Moreover, the revised CIS 221 has knowledge of programming (at the level of CIS 201) as a prerequisite. This was done because most of our incoming students already had considerable programming experience; what they lacked was a grasp of software engineering principles. Thus by assuming this type of experience as a prerequisite, the sequence is able to proceed directly to software engineering issues.

Our capstone design courses continue to be fine-tuned. We currently have six courses designated as capstone design courses. CIS 758, the capstone design course on Software Engineering, is the most popular of these. We often hire practicing software engineers from local industry to supplement faculty in the teaching of this course. This has provided a significant benefit since these outside instructors, being active practitioners of software engineering, are able to bring their real-world experience into the classroom. And since several of these instructors obtained graduate degrees from our department in the recent past, they are also familiar with the overall program, and are able to relate well to the students.

The 1-credit (required) course, CIS 601, on social, ethical, and professional issues in computing has proven remarkably successful. When we introduced the course several years ago, there was some concern that some students, given their strong interest in technical material to the apparent exclusion of all else, may not like

[1] The *BS* in *CSE* program is in the *Computer and Information Science Department* in the *College of Engineering* of the *Ohio State University*; the *Department* also offers several other programs, for example, a *BS* in *Computer and Information Science* degree in the *College of Arts and Sciences*. Only the *BS* in *CSE* is being evaluated.

the course very much. As it turns out, students seem to thoroughly enjoy discussing social, ethical, and legal issues related to computing. As an added bonus, the course provides an excellent opportunity to improve their communication skills, both oral and written.

A.4.2 Processes

Over the last two years, we have been formalizing the processes we use in such matters as establishing program objectives, making changes in the curriculum, introducing new courses in response to the needs of industry, etc. Although some of the processes we now have in place are new, most are simply formalizations of procedures we have followed for many years.

A.4.3 Faculty

One challenge that we, like most other computing programs, have had to face is that of hiring qualified faculty. The enormous growth in the computing industry over the last few years has meant that the number of computing Ph.D.'s going into academia has remained flat or even dropped; at the same time, the demand for qualified faculty has been sharply increasing, as universities try to expand their computing programs to meet the ever increasing demand from students for these programs. Despite this squeeze from both ends, we have been successful in hiring a number of extremely qualified people in many key areas such as networking, computer graphics, software engineering, and artificial intelligence. Also, as noted earlier, we have used this situation as an opportunity to use outside instructors, drawn from the pool of practitioners in the local industry, to teach courses that can benefit from their practical experience.

A.4.4 Facilities

The CIS Department moved into a new building about five years ago. The building has been designed to provide a good working environment for faculty and staff, provide some classrooms wired to support workstations for each student, provide a comfortable environment for students to interact with faculty and teaching assistants, and provide room for the growing research needs of the department.

About four years ago the computing equipment in the various student computing labs, as well as in faculty offices, was replaced by a new generation of equipment. Yet another round of upgrading of the computing equipment is currently under way and is expected to be completed in about a year.

B. Accreditation Summary

This part of the self-study contains eight sections, B.1 through B.8, corresponding respectively to the ABET Criteria 1 through 8. A brief paragraph at the start of each section explains how that particular section is organized.

B.1 Students

This section is organized as follows: Section B.1.1 speaks to the quality of our students and to the high demand from industry for our graduates. In Section B.1.2 we detail the processes used for evaluating, advising, and monitoring our students, in a manner consistent with ABET Criterion 1. In B.1.3 we describe the procedures used to ensure that all students meet all program requirements. And in B.1.4 we describe the policies and procedures used for acceptance of transfer students and for validation of courses taken for credit elsewhere.

B.1.1 Quality of Students and Graduates

Our students are generally of high quality and perform well in a demanding curriculum. The average ACT scores for students admitted into the program last year was 26.5. The average gpa of our students is 2.91. The student population is fairly diverse, with about 13% being women and about 23% minorities[2]. We have a designated *diversity coordinator*, currently Prof. Rick Lewis, who coordinates programs directed at underrepresented groups. As part of this activity, Prof. Lewis is PI on a significant grant from the GE Foundation as part of their 'Faculty of the Future' program.

The program's curricular requirement of 200 (quarter) credit hours can be completed in 4 years, but the average time to complete the degree is 5.3 years. This is due to a large percentage of our students gaining work experience and supporting their education through part-time employment on a continuing basis, as well as through participation in co-op programs and summer internships. Companies that have recently recruited our graduates include *Adobe Systems, Boeing, CompuServe, FedEx, IBM, Lucent, Microsoft, Motorola, Procter & Gamble*, etc. The average starting salary for our graduates is well over $40,000 per year. About 6.5% of our graduates enter graduate school immediately after completion of the program.

B.1.2 Evaluating, Advising, and Monitoring Students

Evaluation of student performance is directly linked to our Program Educational Objectives through course syllabi as indicated in the individual course syllabi, see Appendix I.B. The tables in the individual course syllabi indicate the relation of each course to various program objectives. Thus the performance of students in individual courses, as reflected by their grades in those courses, is a good indication of how well the program is meeting the corresponding objectives. This information is used by individual instructors to make appropriate adjustments to how they present the course in question. The information is also used by course coordinators, when preparing the Course Group Status Reports described in more detail in Section B.2.3, to suggest changes in courses, in prerequisites, etc., and to ensure that program objectives are met.

Another important technique we use to determine success in meeting program objectives is a self-evaluation by students, in the form of an on-line *Exit Survey* that all students must complete prior to graduation. This anonymous[3] survey asks the students to rank, on a scale of 'Strongly Disagree' through 'Strongly Agree', for each of our objectives whether that objective was met for the particular student completing the survey,

[2] The 23% figure includes students of Asian origin; if we exclude those, the figure is 12%.

[3] The survey is anonymous but a separate list of *id* numbers of the students completing the survey is maintained to ensure that all students do in fact complete the survey before graduation; the *id* numbers are not linked in any way to the survey responses.

and how important the objective is to the student. On a regular basis, the survey answers are tabulated and analyzed and the results made available to the Undergraduate Studies Committee, as well as other interested students and faculty.

Students are advised and their progress monitored by a professional advising office staffed by a full-time academic advisor (Ms. Peg Steele) who is assisted by two graduate administrative associates, and by the faculty. Major components of the advising and monitoring system are highlighted below:

1. Information about the program is available to the students in hardcopy from the advising office, and on the department's web page (http://www.cis.ohio-state.edu).

2. Students are admitted to the pre-CSE major upon their acceptance to the university. Once they have completed a series of seven prerequisite courses with the minimum cumulative point hour ratio (*cphr*), they are eligible to apply to the major. Assuming they have met the requirements, they are formally admitted to the major and assigned a faculty advisor. The faculty advisor has an area of expertise that the student expects to specialize in. The faculty-student relationship is expected to be an on-going interaction until the student has graduated from the program. The professional full time advisor remains available for all program and university concerns.

   The minimum *cphr* requirement for admission to the *CSE* major[4] is currently 2.4 (on a 4.0 scale) and will increase to 2.8 starting Summer 1999. During the winter quarter of each year, the Undergraduate Studies committee of the department evaluates both expected demand for the major(s) as well as the faculty and other resources for the coming year and decides whether to propose, to the faculty, any change in this requirement. If a change is proposed, the entire faculty discusses it and votes on it. If the new requirement is approved, all pre-majors are immediately notified by US mail and by announcements on electronic newsgroups; the requirement then goes into effect the following summer quarter. All campus advisors are also informed of the change.

3. Advising reports recording the progress of their advisees are available to faculty advisors (on request to the advising office).

4. The *Degree Audit Registration System* (DARS) is a program that enables the students to verify, on-line, what requirements they still need to meet in order to complete the program.

5. Students are evaluated on a quarterly basis, and those who are not making satisfactory progress are put on probation and advised of what they need to do to get back in good standing. Failure to overcome the deficiencies (typically low overall *cphr* or low *cphr* in computing courses) results in dismissal from the program. Probation and dismissal decisions are made by a standing committee of the College of Engineering consisting of chairs of Undergraduate Studies committees (or representatives thereof) and advising representatives from the various programs.

B.1.3 Meeting Program Requirements

Two quarters prior to graduation, students are required to file a formal application to graduate. The application has to be approved by both the departmental advising office and the Office for Degree Certification of the College of Engineering. The application provides details about all the courses the student has taken, the grades in those courses, as well as the courses the student intends to take in the next two quarters. The advising office and the Graduation Advisor (College of Engineering) review the application carefully to ensure that the student has met all program requirements, including completion of the on-line exit survey.

[4]This requirement also applies to the *CIS* major; the *CIS* major is not being evaluated by *ABET*.

Under exceptional circumstances, students may petition the chair of the Undergraduate Studies committee to substitute a suitable alternate course in place of a required course; such petitions are approved only after careful consideration and only if doing so would not materially diminish the strength of the program. For example, recently a transfer student who had credit for an astronomy course was allowed to use that to partly meet the physics requirement; on the other hand, a petition (from a different student) to use a course on software engineering (CIS 757) as the capstone design course was not approved since, although design tasks do play an important role in CIS 757, there is no large scale design project in the course.

Petitions involving the general education portion of the program are first reviewed by the departmental advising office, and then forwarded to the Petitions Committee of the college. Again, these petitions are approved after careful consideration, provided doing so would not make a material difference in the strength of the program.

B.1.4 Transfer Credits[5]

The department treats the question of *transfer credits* seriously both in order to ensure that students get credit for coursework they have completed elsewhere, and to ensure that transfer students are indeed well prepared to continue with the rest of the program. With this in mind, transfer credits are handled by a two-tier system. The department supplies the University Admissions Office with a *transfer-credit-course* list identifying specific courses from specific universities and the equivalent courses in our program. The correctness and currency of this list is evaluated annually. Any student who has taken course X in university Y, upon filing the requisite application with the Admissions Office, gets transfer credit for the equivalent course Z of the program as per this list.

Occasionally we have students who believe they ought to receive transfer credit for a specific course although the corresponding entry does not appear on the transfer-credit-course list. Hence the department also has a designated faculty member who acts as a *transfer-credit coordinator*. This coordinator is appointed by the chair of the department on an annual basis. The University Admissions Office refers to this coordinator all cases that cannot be resolved on the basis of the list supplied to them. Prof. Doug Kerr is the current (1998-99) transfer credit coordinator.

One important issue having to do with transfer credits concerns the sequence CIS 221, 222, 321. As noted in Section A.4.1, these three courses form a tightly integrated sequence focusing on software engineering issues, rather than mere programming experience in a particular language. We often have students who may have taken one or two courses in programming at another university, and these students often assume that they would get transfer credit for, say, CIS 221. But even if the course(s) the student has taken elsewhere did indeed provide him or her considerable programming experience, giving such credit would not be in the best interest of the student unless the student has also learned and internalized the relevant software engineering principles. Hence in such cases, Prof. Kerr consults closely with Prof. Bruce Weide and Prof. Tim Long, who designed the 221, 222, 321 sequence and are coordinators for the sequence, before awarding such credit. In rare cases, where the student has taken a year-long sequence of courses with a strong software engineering focus, he or she may receive credit for the entire sequence.

[5]Since policies for transfer credits are part of Criterion 1, this information is provided here although the *ABET* self-study instructions have this information listed under Criterion 3.

B.2. Program Educational Objectives

This section is organized as follows: In Section B.2.1 we present our published educational objectives and relate them to our mission. In Section B.2.2 we outline the processes used to determine and evaluate these objectives. In B.2.3 we show how our curriculum and processes ensure the achievement of our objectives. And in Section B.2.4 we describe our system of ongoing evaluation of the program, and present examples of resultant recent changes in the curriculum.

B.2.1 Educational Objectives

The educational objectives of the *CSE* program are published both on the department web pages and in hard-copy literature of the College of Engineering. Our educational objectives are as follows:

1. To provide graduates with a thorough grounding in the key principles and practices of computing, and in the basic engineering, mathematical, and scientific principles that underpin them.

2. To provide graduates with an understanding of additional engineering principles, and the mathematical and scientific principles that underpin them.

3. To provide graduates with an understanding of the overall human context in which engineering and computing activities take place.

4. To prepare graduates for both immediate employment in the CSE profession and for admission to graduate programs in computing.

A key part of the mission of our department is 'to educate undergraduate and graduate majors in computer science and engineering', and the educational objectives enumerated above are clearly consistent with this mission. Similarly they are consistent with the mission of the College of Engineering, in particular with its mission 'to educate professionals in engineering and architecture'.

Corresponding to each of our four objectives, we have identified (and published on the department web pages) specific, detailed, outcomes. In the listing below[6] of these outcomes, *1a, 1b, 1c* are the outcomes corresponding to objective 1, and similarly for each of the other objectives:

- 1a. Students will demonstrate proficiency in the areas of software design and development, algorithms, operating systems, programming languages, and architecture.

- 1b. Students will demonstrate proficiency in relevant aspects of mathematics, including discrete mathematics, as well as the appropriate concepts from physics and electrical circuits and devices.

- 1c. Students will successfully apply these principles and practices to a variety of problems.

- 2a. Students will demonstrate an understanding of differential and integral calculus, differential equations, physics and several areas of basic engineering sciences.

- 2b. Students will have the ability to work with others and on multi-disciplinary teams in both classroom and laboratory environments.

[6]The *ABET* self-study instructions for this section require only the listing of the educational objectives here, not the outcomes corresponding to each objective. However, since the relation between the curriculum and the objectives is *via* the detailed outcomes, and since the instructions call for this relation to be discussed in this section, we have listed the outcomes here so that this relation can be discussed (in Section B.2.3). The outcomes are listed again at the start of Section B.3 as called for by the instructions for that section.

3a. Students will demonstrate an ability to communicate effectively.

3b. Students will obtain familiarity with basic ideas and contemporary issues in the social sciences and humanities.

3c. Students will obtain an understanding of social, professional and ethical issues related to computing.

4a. A large fraction of graduates will be immediately employed in high-technology companies that utilize their computing education.

4b. Strong graduates from the program will be prepared to enter good graduate programs in *CSE*.

B.2.2 Process to Determine and Evaluate Objectives

The main constituencies of our program are our current students, alumni of the program, supervisors and managers of our alumni who are currently employed in industry, and industries who are likely to employ our future graduates. Input from all of these constituencies is used in the process described below to determine and evaluate our objectives.

Getting Feedback

Input from current students is obtained via a number of mechanisms. First, we have student representatives on key committees, including the Undergraduate Studies Committee, the Curriculum Committee, and the Computer Committee. Second, an open undergraduate forum is held each year to discuss all aspects of our program, including its objectives. The forum is held in the spring quarter and is attended by interested students, key faculty members, and the academic advisor. Announcements about the forum are made widely, especially on the electronic (student) newsgroups to ensure wide participation. Following the forum, a summary of the discussion is posted on the newsgroups by the chair of the Undergraduate Studies Committee; this usually leads to further extended discussions where students (including those who could not attend the forum) further express their opinions and ideas on the program. Third, as noted in Section 1.2, before graduation, students are required to complete an exit survey. One portion of this survey asks students to rate the importance (on a scale of 'very unimportant' to 'very important') of each of our objectives as well as each of the corresponding outcomes listed above[7]. On a regular basis, the survey answers are compiled and analyzed and the results made available to the members of the Undergraduate Studies committee, as well as other interested students and faculty.

Input from alumni is obtained by means of an anonymous (hard-copy) survey. Each year this survey is mailed out to alumni who graduated either two years ago or six years ago; this allows us to gather input from alumni who graduated relatively recently as well as some who graduated a while ago, without at the same time asking for input from the same group of people year after year. The College of Engineering keeps track of alumni addresses, etc., and mails out the surveys, collects the returned surveys, and forwards them to the program. As in the case of the exit survey, the responses from the alumni survey are compiled and analyzed and the results made available to the members of the Undergraduate Studies Committee, as well as other interested students and faculty.

Unfortunately, and unlike the student exit survey, the current version of the alumni survey does *not* ask respondents to rank the importance of our objectives; it only asks how well each objective was met. Next

[7] The exit survey asks two sets of questions. The first, as just noted, asks the student the rank the importance of each objective and outcome. The other asks the student to rate how effective the program was in meeting that objective or outcome. This latter set of questions dealing with the effectiveness of the program will be discussed in Section B.2.4. Similar comments apply to the alumni and manager/supervisor surveys discussed below.

year we will revise the alumni survey to include a ranking of the importance of our objectives. Further, we will also include, again as in the exit survey, the specific outcomes corresponding to each objective (listed earlier) rather than just the high-level objectives as is the case in the current version of the alumni survey.

Input from supervisors/managers of our graduates is similarly obtained by means of a survey conducted annually. The survey has two components: the first asks the manager to rate the importance of various objectives and outcomes, the second asks him or her to rate how well graduates of our program were trained with respect to each objective and outcome. Again, the College sends out the survey, collects the completed surveys, and forwards them to the program. As in the case of the exit survey and the alumni survey, the responses from the manager survey are compiled and analyzed and the results made available to the members of the Undergraduate Studies committee and other interested students and faculty.

We faced two problems in this survey. First was the question of asking supervisors to rate how well their employees were prepared by our program, and the likely unwillingness of managers to answer such questions because of legal and privacy considerations. If this turns out to be a major issue with a serious impact on the return rates of the surveys, we will consider eliminating this component of the survey, and ask only for a rating of the importance of each objective.

The second problem we faced with respect to this survey was that of generating a list of names of managers who would be able to complete such a survey. By consensus among the engineering programs, it was decided that students who graduated from the College of Engineering fifteen years ago are now likely to be in managerial positions; hence it would be appropriate, each year, to send out the survey for that year to alumni of the College who graduated fifteen years ago.

Two important constituencies that we have not discussed above are prospective students (and their parents), as well as recruiters of students who are close to graduating from our program. Members of the Undergraduate Studies Committee and the academic advisor occasionally meet with members of both constituencies when they visit campus and get informal feedback from them on our objectives and outcomes. This feedback is also incorporated into our deliberations in the committee. We are investigating ways to formalize this process, perhaps in the form of a brief survey asking recruiters and prospective students (and parents) to rank the importance of our objectives.

Using the Feedback

The Undergraduate Studies Committee, a standing committee of the faculty that includes student representatives, initiates discussions on program objectives and outcomes, taking into account the inputs from the various constituencies. The meetings of the committee are announced in advance on student and faculty newsgroups and are open to all interested students and faculty. The minutes of the meetings are posted on the newsgroups. Following extensive discussions in the committee, the current set of objectives and outcomes (listed in Section B.2.1) were presented to the department faculty in the fall quarter of '97 and were approved by the faculty. The objectives and outcomes were also presented to the members of the department's Industrial Advisory Board at about the same time for their comments and feedback.

The feedback we have received so far from our surveys is that our current set of objectives and outcomes are reasonable and appropriate. By the end of the next academic year (1999-2000), the survey mechanisms would have been in place for about two years. Further, by then we should have received the first set of results from our improved alumni survey. At that time we will undertake a careful review of all the feedback and identify any changes that need to be made. One trend that is already quite visible in the feedback we have received is the importance of communication and team-working skills. These skills are included in the detailed outcomes (2b) and (3a) but they are not explicitly listed in our high-level objectives, and we may wish to revise these objectives to list these skills explicitly. But it is worth repeating that the feedback so far has been very positive, and so any changes are likely to be in wording rearrangements or perhaps in the relative stress that is placed on the various outcomes, rather than wholesale changes in the main objectives. The revised set of objectives and outcomes, as well as a summary of the feedback received and the reasons for the revisions, will be presented to the faculty for discussion and approval; this will be also be presented to our Industrial Advisory Board for its comments and feedback.

B.2.3 Curriculum and Processes to Ensure Achievement of Objectives

Our curriculum has been designed to ensure that each of our educational objectives is fully met. We first identify, for each of our four objectives and the corresponding outcomes listed in Section B.2.1, the specific courses that contribute toward meeting that objective and outcome, and then discuss our curriculum related processes.

B.2.3.1 Curriculum

Outcome (1a), that students will demonstrate proficiency in the key areas of computing, and outcome (1c), that they will apply the principles and practices of these areas to a variety of problems, are addressed by a whole range of our courses, starting from the beginning software development sequence (CIS 221, 222, 321); the introductory course on computer architecture (CIS 360); the course (CIS 560) which brings together software design and architectural issues in the setting of a large team-based project; the microprocessor laboratory (EE 567); the "core" courses on formal languages (CIS 625), programming languages (CIS 655), operating systems (CIS 660), computer architecture (CIS 675), and algorithms (CIS 680); and the capstone design courses.

The first part of outcome (1b), that students will demonstrate proficiency in relevant aspects of mathematics, including in particular discrete mathematics, is addressed by the calculus series (Math 151, 152, 153, 254, 415) as well as the discrete mathematics course (Math 366). In addition, a course (CIS 541) on numerical methods further develops relevant calculus-based ideas and concepts from a computing point of view; and discrete mathematics ideas and concepts are further developed and applied in many courses (e.g., CIS 570, 625, 655, and 680). The second part of outcome (1b), that students will demonstrate proficiency in appropriate concepts of physics and electrical circuits and devices is addressed by the physics series (Phys 131, 132, 133); and by the Electrical Engineering courses (EE 206, 261, 300, 309, 320, and 567). Concepts related to digital electrical circuits are used in two other courses (CIS 360, and to a greater extent in CIS 675). Given the breadth and the depth of this range of courses, objective 1, to provide graduates with a thorough grounding in the principles and practices of computing, and the basic engineering, mathematical, and scientific principles underpinning them, is solidly met.

Objective 2, to provide graduates with an understanding of additional engineering principles, and the mathematical and scientific principles underlying them, has two outcomes, (2a) and (2b). The first part of outcome (2a), that students will demonstrate an understanding of calculus and differential equations, is addressed by the calculus courses (Math 151, 152, 153, 254, 415). The second part of outcome (2a), that students will demonstrate an understanding of physics and several areas of basic engineering sciences is addressed by the physics courses (Phys 131, 132, 133); and by courses in several areas of engineering (Engineering Graphics 166, Engineering Mechanics 400, Industrial and Systems Engineering 504, Materials Science and Engineering 405, and Mechanical Engineering 500).

Outcome (2b) having to do with team-working skills, is met by the large-scale team project in CIS 560 where groups of four or five students work together throughout the quarter, on a range of activities, from designing a system, to documenting it, to conducting design meetings, recording minutes, etc. Before tackling this large-scale team activity, students become familiar with working in a smaller team (two students) on a smaller project, in CIS 321. Team-working skills, especially in a multi-disciplinary setting, are also enhanced by EE 567, the Microprocessor Laboratory. In this course, teams of 2 or 3 students work on building a small microprocessor. Since both *CSE* and *ECE* (Electrical and Computer Engineering) majors take this class, it contributes to the multi-disciplinary component of the experience. Similarly, CIS 321 and 560 also helps in this regard because *ECE* majors, as well as Computer and Information Science majors from the College of Arts and Science, are required to take the course, and often teams in the course have members from these different groups.

Early exposure to building team-working skills is the practice in CIS 221, 222, and 321 of pairing two students at one workstation in 'closed-lab' sessions; and of having, in the same classes, groups of two or three students engage in group-learning activities during class time. Several of the capstone design courses, such as CIS 758 (Software Engineering), CIS 772 (Information Systems Project), and CIS 776 (Hardware/Software Interface Design Project) require students to work in teams. Another capstone design course, CIS 731 (Knowledge-Based Systems), is currently experimenting with having students work in teams; and some of the faculty involved in teaching CIS 762 (Advanced Operating Systems Laboratory) and CIS 778 (Computer-aided Design and Analysis of VLSI Circuits) are discussing the possibility of introducing teams into the projects in those course. Once this is done, all our capstone design courses will have students working in teams.

Objective 3, to provide graduates with an understanding of the overall human context, is met by CIS 601 (*Social and Ethical Issues in Computing*), as well as by the general education courses (GEC) in Social Sciences, Arts and Humanities, and Historical Survey. CIS 601 deals specifically with the relation of computing to society (outcome (3c)), while the GEC courses address outcome (3b), that students are familiar with ideas and issues in the social sciences and humanities. CIS 601 also contributes heavily towards outcome (3a), that students have an ability to communicate effectively. By means of individual presentations, formal in-class debates, as well as written reports on topics to do with the relation of computing to society, the class sharpens both oral and written communication skills of the students. CIS 560 also makes an important contribution to the writing skills of our students since complete design documents are a key requirement of that course. Other courses too, by requiring suitable (if somewhat less extensive) documentation of computer programs, improve the effectiveness with which students can communicate technical material. CIS 560 also helps improve the oral communication skills of students since formal design review meetings as well as interactive grading sessions are part of this course. Further, the relatively small class sizes in all the CIS courses encourages active class discussions, thereby helping students develop their group-communication skills.

Objective 4 has two outcomes; outcome (4a), that a large fraction of our graduates will be employed in high-tech computing companies, is addressed by most of the CIS courses in the curriculum, including especially the 600-level courses, the capstone design courses, and the many elective courses on such current topics as networking and computer graphics. Outcome (4b), that strong graduates will be prepared to enter good graduate programs, is addressed by the 600-level courses as well as the more advanced (graduate-level) courses that interested students can choose as electives; examples of such courses, frequently selected by CSE majors, are CIS 725 on computation theory, CIS 755 on programming languages, CIS 760 on operating systems, and CIS 775 on computer architecure.

The syllabi of individual courses that appear in Appendix I.B specify which outcomes each course contributes to, and to what degree. We have also included, in Appendix I.B, a table that summarizes, for each of our objectives and outcomes, which courses contribute to it, and to what degree.

B.2.3.2 Processes

The departmental Curriculum Committee is a standing committee of the faculty and coordinates all curriculum related processes. The committee has (both graduate and undergraduate) student representatives on it. The meetings of the committee are announced in advance (on student and faculty newsgroups) and are open to all interested students and faculty. The minutes of the meetings are posted on student and faculty newsgroups. Proposals for changes in courses, as well as proposals for new courses, are brought to the committee by interested faculty members. Relatively minor changes in individual courses, such as minor changes in prerequisite courses, are approved by the committee following appropriate discussion (but see later discussion of *Course Groups*). Proposals for more serious changes, such as changes in course description or a major change in course content, as well as proposals for new courses, are first discussed by the committee and, if found appropriate, are presented to the faculty for approval before being forwarded to the appropriate college-level committee (College Committee on Academic Affairs) and university-level committee (Council on Academic Affairs). Pilot versions of new or significantly revised courses are taught at least once and usually twice before permanent versions of the course are considered for approval.

Each regular course offered by the department has a designated faculty course coordinator. Typically, the course coordinator teaches the course at least once each year. The coordinator interacts with other instructors of the course and works with them on any problems in the course. He or she also serves as the person who can provide information about the course to students interested in taking the course, as well as to instructors who are new to the course. In the case of beginning courses which are often taught by graduate teaching associates, the coordinator interacts with the instructors on a regular basis to ensure that GTAs who are new to the course as well as to teaching in general have someone to rely on. Senior GTAs of a given course often provide informal mentoring to their junior colleagues teaching the same course.

Course Groups

A relatively recent component of the curriculum-related process is what we call *Course Groups*. The various courses in the program are organized into groups of related courses. Thus, for example, the Software Engineering group consists of: the beginning courses CIS 221, 222, 321; CIS 560, the systems programming course with the large size team-based project; CIS 757, the survey course on software engineering and CIS 758, the capstone design course on software engineering; and CIS 601, the course on social, ethical, and professional aspects of computing.[8] Another group, this one having to do with an application area, is Computer Graphics, and consists of the four graphics courses, CIS 681, 781, 782, and 881.

Once every two years, the coordinators of the courses in a given group are responsible for producing a *Course Group Report* and presenting it to the Curriculum Committee. The report is expected to address such questions as whether the courses are meeting their objectives, whether prerequisites are appropriate, suitability of the current textbooks, student reactions to the courses in the group, relation to the rest of the program, and whether the courses in the group are helping meet the overall program objectives as intended. The report also provides information about the group's ideas for possible changes in the courses. This mechanism ensures that changes in individual courses are not made without taking account of the likely impact on related courses.

The chair of the Curriculum committee, in consultation with the concerned course coordinators, schedules the presentations to the committee at least one quarter in advance, to give the group time to produce a thoughtful and comprehensive report and presentation. Usually, two groups' reports are scheduled for each quarter. The report of each group is also made available on the web for students and faculty to get a

[8] The reason that CIS 601 is included in this group is that many of the case studies in this course deal with software engineering related issues. Too, as noted earlier, both 560 and 601 have a heavy emphasis on team-working and communication skills.

feel for the thinking of the group and for providing feedback. So far, reports have been produced by the Programming Languages, Computer Graphics, and Software Engineering groups. Copies of all the reports will be available during the on-site visit.

For courses that are taught by other departments, interaction between the respective committee chairs ensures that any changes that are being contemplated are helpful. In some cases, such as mathematics, we have a designated faculty member as a liaison with that department. Equally important, especially for discrete mathematics, is that we have strong informal ties with the relevant faculty of the Mathematics Department to make sure that the course is well suited to the needs of our students. Recently for example, following extensive discussions between some of the mathematics faculty and some of the faculty in our department, an experimental version of the discrete mathematics course (Math 366) with a focus on *proofs* was taught. At the end of that course, the mathematics professor who taught the course produced a report analyzing the advantages and disadvantages of this new version. On the basis of that report, we decided to stay, at least for now, with the previous version of the course, although discussions of other changes are ongoing.

Also worth mentioning is a recent effort, by the various programs in the College of Engineering working together, to redesign the common portion of the all the Engineering programs. A 'Core Committee' consisting of representatives from various programs in the college has been trying to redefine the common ('core') portion of all engineering programs to provide flexibility to individual programs while at the same time ensuring that issues common to all of engineering are not sacrificed. As part of this effort, the committee organized a day long retreat for all engineering faculty, student representatives, faculty from other universities involved with introductory engineering courses, and invitees from key industries, to discuss the concept and content of an 'introduction to engineering' course sequence. The efforts of this committee are continuing and it is expected that a revised engineering core will go into effect during the '00-'01 academic year.

B.2.4 Evaluation of, and Improvements in, the Program

We use a number of different assessment mechanisms to evaluate the effectiveness of our program. These include:

a. Assignments and examinations in individual courses.

b. Student Evaluations of Teaching (SETs) in all sections of all courses.

c. Exit surveys that all students complete about two quarters prior to graduation.

d. Alumni surveys.

e. Supervisor/Manager surveys.

Briefly,[9] examinations in individual sections are standard and include both mid-term and final examinations. The SET allows the student to provide the instructor (anonymous) feedback on the effectiveness of the course as taught by that particular instructor. The exit survey allows the student to rank, for each objective and outcome listed in Section B.2.2, how well the program met that objective or outcome in their individual case. Similarly, the alumni survey allows the respondent to rank how well the program met each objective and outcome in their particular case. The supervisor survey allows the respondent to provide the same ranking with respect to graduates of our program that he or she supervises.

The feedback we receive from the undergraduate forum described in B.2.2, as well as the input from student representatives on the Curriculum Committee and the Undergraduate Studies Committee, play key roles in

[9]It is rather difficult to completely separate the assessment mechansims used at the program level from those at the curriculum level. We provide full descriptions of each mechanism in Section B.3. Here we provide one-line descriptions of each mechanism, and then move on to evidence of improvements in the program as a result of the assessments.

assessing aspects of the program and are, in part, responsible for some of the changes listed below. Another important evaluation tool we have used is outside consultants. Over a year ago, we hired two consultants (with considerable experience with accreditation issues) to evaluate much of the CSE program; their reports also influenced some of the changes listed below.

1. One important piece of feedback we have received from the exit survey, the alumni survey, and the supervisor survey, as well as from the report of our outside consultants, is the importance of communication, especially oral communication skills, and the relative weakness of our program with respect to this skill. In response, CIS 601 was recently modified to include *formal debates* (in addition to individual presentations by students).

   In addition, several of the capstone design courses are experimenting with oral presentations. If these experiments are successful, this will become a standard component of all the capstone design courses.

   Also, as noted earlier, during the next round of revisions to our program objectives and outcomes, we will include an explicit declaration concerning communication and team-working skills among our program objectives.

2. Another important feedback we have received from the surveys, as well as from the undergraduate forum, is the importance of, and the need for courses that deal with, new technologies. This is especially important in a fast moving field such as computing.

   CIS 459.23, a one-credit course on *Java* was recently developed partly in response to this. Although issues relating to different languages, including *Java*, are discussed throughly in CIS 655 (the course on principles of programming languages), this new course will allow students to get practical experience in using the language, and enable them to use *Java* (rather than *C* or *C++* as they used to) for programming projects in higher level courses such as CIS 560 etc.. Hence these students, by the time they graduate, would have had considerable training in this important programming language.

   Another course, CIS 694V, a 3-credit course on multi-media networking is being currently developed. Given the enormous potential for practical applications using multi-media networking, it is clear that many of graduates are likely to be interested in this field. This course will provide them with a solid foundation in the conceptual issues in the field, and will also expose them to its various potential applications. A pilot version of this course taught in Autumn '98 was extremely well received; a second pilot is likely to be taught in '99-'00 before a permanent version is considered for approval.

   But it is important to stress one point: We have resisted (and will continue to resist) calls for courses on relatively narrowly focussed ideas and technologies, e.g., *COM*. The point is that although COM uses OO ideas, it is a particular technology rather than being a conceptual foundation on which a range of technologies can be built. It is understandable that about-to-graduate students as well as recent graduates are keen on being trained in currently popular technologies, but we want to ensure that we do not do this at the expense of conceptual foundations. This is also the reason why, for example, we do not offer a course on *UML*, another currently popular OO technology, or UML-related CASE tools; although CIS 757, the elective course on software engineering, does spend about two weeks on the main ideas and notations of UML.

3. Feedback from a number of alumni as well as current students strongly suggested that while many of the general science and engineering topics that were included in our program were appropriate, and were covered at an appropriate depth, there was far more chemistry than was necessary for most *CSE* majors. The new flexible engineering core currently being discussed (see last paragraph of Section B.2.3) will allow us to address this concern, probably during the '00-'01 academic year. One

possibility that we will consider is to replace the second chemistry course (Chem 125) by, for example, a second discrete mathematics course, since that would be more directly related to computing issues.

Feedback from alumni and current students also strongly suggest that some courses on business and finance would be valuable. We will attempt to address this concern too probably during the '00-'01 academic year, again possibly using the flexibility provided by the proposed new engineering core to include such a course in the program.

4. Using surveys to evaluate the program is fairly new to us and designing the surveys has been somewhat of a challenge. Our experience so far with the surveys has already taught us some important lessons. For example, we have learned that open-ended questions typically are not useful; thus, for example, the alumni survey contained an item that asked the respondent for 'additional comments and suggestions'. We got responses such as 'Some of the faculty in the program were excellent, others were terrible.' While this was probably true, it does not identify any specific problem with the program nor does it tell us how to improve the program. Our conclusion is that it is best not to include such questions in the surveys; or at least, not to expect them to reveal much if they are included.

   A second lesson, this one a positive one, came from our exit survey. It asked respondents not only to rate to what degree the program met each of its objectives and outcomes in their particular case, but also how important, in their opinion, the particular objective or outcome was. The result was that we could immediately identify the most important problems – the ones that were rated as relatively high in importance but relatively low in being met. We did not include the importance question in the alumni survey this year but we will certainly do so in future surveys.

   Another change we are considering is adding to the exit survey a set of questions concerning Criterion 3 outcomes. While the advantage of doing this is clear –it will give us additional data– the flip side is that the length of the survey may become somewhat excessive. We will discuss this in the Undergraduate Studies committee early in the '99-'00 academic year and then make a decision on it.

The changes listed above are, in a sense, at the program level. There have also been a number of important changes within individual courses. Some of these are listed in Section B.3.3.

B.3. Program Outcomes and Assessment

This section is organized as follows: In Section B.3.1 we list the program outcomes (from Section B.2.1) and show how these are related to the outcomes (a) through (k) of ABET Criterion 3, as well as to our program objectives. In Section B.3.2 we discuss the relation between our curriculum and Criterion 3 outcomes. In Section B.3.3 we describe our assessment process and the details of each of our assessment mechanisms (these were listed in B.2.4), and provide further examples of resultant improvements in the program. In Section B.3.4 we describe the materials that will be available for review during the on-site visit.

B.3.1 Program Outcomes

As listed in Section B.2.1, we have established the following outcomes corresponding to our program objectives:

1a. Students will demonstrate proficiency in the areas of software design and development, algorithms, operating systems, programming languages, and architecture.

1b. Students will demonstrate proficiency in relevant aspects of mathematics, including discrete mathematics, as well as the appropriate concepts from physics and electrical circuits and devices.

1c. Students will successfully apply these principles and practices to a variety of problems.

2a. Students will demonstrate an understanding of differential and integral calculus, differential equations, physics and several areas of basic engineering sciences.

2b. Students will have the ability to work with others and on multi-disciplinary teams in both classroom and laboratory environments.

3a. Students will demonstrate an ability to communicate effectively.

3b. Students will obtain familiarity with basic ideas and contemporary issues in the social sciences and humanities.

3c. Students will obtain an understanding of social, professional and ethical issues related to computing.

4a. A large fraction of graduates will be immediately employed in high-technology companies that utilize their computing education.

4b. Strong graduates from the program will be prepared to enter good graduate programs in *CSE*.

The correspondence between our outcomes and ABET Criterion 3 outcomes is summarized in the table below.

Criterion 3 Outcome	**Program Outcomes**	**Criterion 3 Outcome**	**Program Outcomes**
3.a	1b, 1c, 2a	3.g	3a
3.b	1a, 1c	3.h	3b, 3c
3.c	1a, 1c	3.i	4a, 1a
3.d	2b	3.j	3b, 3c
3.e	1a, 1b, 1c, 2a	3.k	1c, 4a
3.f	3c		

The relation between our program outcomes and our program objectives is very direct and is indicated by the numbering we have used for the outcomes and the objectives.

B.3.2 The Curriculum and Criterion 3 Outcomes

Our curriculum has been designed to ensure that each of the ABET Criterion 3 outcomes is fully met. We consider each outcome in turn, and explain how and which parts of the curriculum helps us meet it. The syllabi of individual courses that appear in Appendix I.B specify which of the Criterion 3 outcomes each course contributes to, and to what degree. We have also included, in Appendix I.B, a table that summarizes, for each Criterion 3 outcome, which courses contribute to it, and to what degree.

Outcome (3.a), that graduates will have an ability to apply knowledge of mathematics, science, and engineering, is met by a whole range of courses in the program. These include nearly all the CIS courses, starting from the beginning sequence CIS 221, 222, 321, with its focus on software engineering issues, through courses such as CIS 560 which continues the focus on software engineering; CIS 360 which introduces computer architecture principles, EE 567 which builds on this by designing and building simple systems, and CIS 675, which goes into greater detail of computer architecture; the calculus sequence, Math 151, 152, 153, 254; the discrete math course, Math 366; the physics sequence, Phys 131, 132, 133; and by courses in several areas of engineering, Engineering Graphics 166, Engineering Mechanics 400, Industrial and Systems Engineering 504, Materials Science and Engineering 405, and Mechanical Engineering 500.

Outcome (3.b), that graduates will have an ability to design and conduct experiments, and to analyze and interpret data, is also met by several of the courses in the program, in particular, the science courses such as the physics sequence (Phys 131, 132, 133), as well as engineering courses such as Engineering Mechanics 400 and Mechanical Engineering 500; Statistics 427 and 428 also help strengthen the abilities of students with respect to analyzing and interpreting data. Outcome (3.b) is also ensured by a number of CIS courses; the reason is that a very important activity in building software is to test the software once it has been designed and implemented; this involves careful selection of test cases and interpretation of the results, very much like conducting experiments in physics or chemistry. This is especially true when the software in question is relatively large, as in the case of the projects in CIS 560, as well as in many of the 600-level courses. Thus, each of these courses contributes toward meeting outcome (3.b).

Outcome (3.c), that graduates will have an ability to design a system, component, or process to meet desired needs, is a central focus of the program. The key issue that several of the courses in the program address is, how to design a computing system that meets specified requirements. Specifically, CIS 221, 222, 321, 360, 541, 560, 570, 655, 660, 675, 680, and the capstone design courses all contribute to outcome (3.c).

Outcome (3.d), that graduates will have an ability to function on multi-disciplinary teams, is primarily ensured by CIS 560. In this course, students form teams of four or five members each, and work together on a quarter-long set of projects. The team has to carry out a range of tasks, including design, documentation, implementation, and testing. The members of the team have to find the best ways of dividing up the tasks to best use the different strengths of the individual team members. The team is also required to have regular meetings with the minutes of the meeting being recorded. The team meets with the instructor and grader for the course as a team and its members are evaluated as a team. (There are also examinations in the course which are graded on an individual basis.) CIS 321 also contributes to (3.d), by requiring students to work in small teams (of two members each) on a quarter-long project. In addition, several of the capstone design courses (such as CIS 758 and CIS 776) require students to work in teams on their projects; others (such as CIS 731) are experimenting with team projects.

EE 567, the Microprocessor Laboratory, taught by the Electrical Engineering Department, also contributes to our students' team-working skills, especially in a multi-disciplinary setting. In this course, teams of 2 or 3 students work on building a small microprocessor. Since both *CSE* and *ECE* (Electrical and Computer Engineering) majors take the course, our students learn to work with people from other disciplines. Indeed, even CIS 321 and 560 also help in this regard because *ECE* majors, as well as Computer and Information Science majors from the College of Arts and Science, are required to take these courses, and often teams in the course have members from these different groups.

Outcome (3.e), that graduates will have an ability to identify, formulate, and solve engineering problems, is ensured by several of the CIS courses that require students to build software systems to meet specific requirements. In the early courses, such as CIS 221, the requirements are precisely defined, so the students focus on solving the problem. As they progress through the curriculum, the specifications tend to become relatively loosely defined, and the students assume a greater responsibility in identifying and formulating the problems as well. And in several of the capstone design courses, the problem is stated at a very high and informal level, leaving it to the students to identify it more precisely, come up with a formulation of the exact requirements, and design a solution.

In addition, some of the non-CSE engineering classes such as EE 320, electronic devices and circuits, and ME 500, engineering thermal sciences, sharpen the abilities of our students to identify, formulate, and solve problems in a range of engineering fields.

Outcome (3.f), that graduates will have an understanding of professional and ethical responsibility, is ensured primarily by CIS 601; indeed professional and ethical issues involving *CSE* is the central topic of that course. Following a brief presentation of general ethical principles, this course typically involves looking at a number of case studies to illustrate the professional and ethical issues. These issues are also touched on in earlier courses such as the beginning sequence where the *CSE* professional's responsibility to produce bug-free software is an ever-present, if implicit, consideration.

Outcome (3.g), that graduates will have an ability to communicate effectively, is ensured by a number of courses. Key among these are CIS 560 and CIS 601. CIS 560 requires students to produce fairly complete documentation of the design of their projects as well as its implementation; they are also required to produce a 'user guide' for the system they implement. The students receive feedback on these documents to help them improve the effectiveness of their communication skills.

CIS 601 develops both oral and written communication skills. Students have a unique opportunity to engage in a formal debate in which they are required to present and defend a position – that they may not necessarily even believe in! The instructor assigns to each student the position that he or she must defend in the debate. The course also requires students to write papers involving legal and ethical aspects of computing, thereby also helping to develop their written communication skills.

Many of the capstone design courses also require students to give oral presentations, and to produce fairly extensive documentation of the systems they design, giving the students in these classes a further opportunity to improving their communication skills.

Outcome (3.h), that graduates have the broad education necessary to understand the impact of engineering solutions in a global and societal context, is ensured by CIS 601 since the discussion in that class, while it involves computing issues, is necessarily in the context of society as a whole, so it deals with the relation between computing and society. In addition, the required general education courses, consisting of at least 9 credit hours of social sciences, at least of 9 hours of arts and humanities, and at least 10 hours of historical survey, ensure that outcome (3.h) is met.

Outcome (3.i), that graduates will recognize the need for, and have the ability to engage in life-long learning, is critical for *CSE* majors, given the rapid pace of change in the field. CIS 221 and 222 require students to read columns from, e.g., *IEEE Computer* about professional issues such as licensing of software engineers; students are also required, in these courses, to learn HTML on their own to do lab assignments. Several required courses, as well as several elective courses, explore new ideas in computing. For example, CIS 655, the required course on concepts of programming languages, discusses non-standard programming paradigms such as logic programming and functional programming which present useful ways of designing certain specialized types of systems. CIS 660, the required course on operating systems, explores how certain complex problems having to do with synchronization between concurrently running processes may arise, and how different systems have tried to address them. CIS 677, 678, and 777, the series of (elective) courses on networking expose students to various networking protocols that have been developed and the problems they solve as well as the problems they do not address.

Outcome (3.j), that graduates will have a knowledge of contemporary issues, is ensured by general education courses noted under outcome (3.h). One important requirement not mentioned under (3.h) is that our students are required to take a 5 credit-hour course on social diversity; this course may also be counted toward, for example, the 9 hours of social science, but each student's program must include a 5-credit course that addresses diversity issues.

Outcome (3.k), that graduates will have the ability to use the techniques, skills, and modern engineering tools necessary for engineering practice, is ensured by several of the CIS courses. Thus, for example, in the beginning sequence, students learn to use the World Wide Web; use e-mail and electronic newsgroups to interact with other students and with their instructors; etc. This continues throughout the program, and well before the time the students graduate, using these tools becomes second nature to them.

B.3.3 Assessment Process

The College of Engineering has appointed a committee, the Assessment Committee, with the Associate Dean for Academic Affairs as its chair, and consisting of faculty representatives from each of the programs in the college. This committee was constituted in the fall of 1997, mainly with the goal of helping individual programs prepare for ABET accreditation, and has been meeting on a regular basis, on the average about every other week. Since assessment was the area that most programs needed help with, the committee naturally focussed on that. The concept as well as the details of the alumni survey as well as the manager/supervisor survey, were designed by this committee. The meetings of this committee enable faculty from the different programs to exchange ideas and learn from each others' experiences.

Within the department, the Undergraduate Studies Committee is responsible for assessment related issues. The committee discussed and designed the exit survey for our graduates. The committee also discussed the approaches that will be used to analyze and interpret the survey data. Finally, the committee regularly summarizes the results of these surveys (as well as the other evaluation mechanisms) to the faculty and initiates discussions for possible changes in the program.

Next we describe each of the assessment mechanisms we use:

1. *Assignments and examinations*: Each course in the program relies heavily on the time-tested method of assignments and examinations. They serve a number of purposes: First, they allow an instructor to identify common problems and misunderstandings among students so that the next time the instructor teaches the course, he or she can make appropriate adjustments in the way the topic in question is approached, or in the amount of time that is spent on the topic. Second, they allow the instructor to identify any potential problems in prerequisite courses or topics that should have been presented in those courses. Third, they allow individual students to get a very good feel for how well they have

mastered the material in the course. And fourth, they similarly allow the instructor to determine to what degree each of the students has achieved the various objectives of the course. This is especially critical given the direct correlation described earlier between the various courses in the program and our program objectives and outcomes as well as Criterion 3 outcomes

2. *Student Evaluations of Teaching (SETs)*: Each instructor of each course in the program also gets feedback via the mechanism of SETs. Near the end (typically in the eighth or the ninth week) of the quarter, students are asked to complete an evaluation of the teaching in the particular course. Students' names do not appear anywhere on the evaluation; the instructor is not present in the classroom when the evaluations are completed; a volunteer student collects all the evaluations and returns them to the department's main office where they remain until the quarter is over and the instructor has assigned grades for the class. At that point, the chair of the department (and in the case of instructors who are graduate teaching associates, also the GTA coordinator) reads through the evaluations, before giving them to instructor.

   The SET form we use consists of a set of questions that ask students to rate, on a scale of 'very unsatisfactory' to 'very satisfactory', a whole range of items concerning the course. This includes, for example, the effectiveness of the instructor in communicating the subject matter and stimulating interest in it, and the appropriateness of textbooks, homeworks and programming labs, and exams. The form also provides room for students to provide general comments on the course material and suggestions for changes. While the assignments and exams often allow an instructor to identify a problem in the course, the SETs allow him or her to also identify the *reason* for the problem. The feedback from the SETs as well as from the assignment and exams allow instructors to discuss possible changes in the course to help it better meet its objectives, or perhaps question the appropriateness of some of the objectives of the course. These discussions and the resulting ideas all feed into the course group reports, described in Section B.2.3, for the group that the particular course is a part of.

3. *Exit survey*: All *CSE* majors, as part of their application to graduate, are required to complete an on-line exit survey. The survey asks students two sets of questions. The first one asks the student to rank, on a scale of 'Very Unimportant' through 'Very Important', the importance of each of our program objectives and outcomes. Next, the student is asked to rank, on a scale of 'Strongly Disagree' through 'Strongly Agree', how well the particular objective or outcome was met by the program in his or her individual case (as against their perception of how well the objective or outcome was met for the average student). The survey is conducted via a web page, so students can take the survey at their convenience. Anonymous responses from the survey are stored on-line for processing at a later time. The student ids of the individual completing the survey is sent to the advising office, so that office is aware that the student has completed this requirement. Periodically the raw responses are analyzed and interpreted, and the results discussed by the Undergraduate Studies Committee as indicated earlier.

4. *Alumni survey*: Each year, alumni of the program, who graduated either two years ago or six years ago, are asked to complete a survey. The first part of the survey asks the respondent to rank the importance of each of several general educational outcomes (these were decided on by the College Assessment committee), as well as how well the program prepared them with respect to these outcomes. These outcomes are very much based on Criterion 3 outcomes. The second part of the survey asks the respondent to assess his or her educational experience in the program, by ranking, on a scale of 'Unsatisfactory' through 'Excellent', a range of items such as the quality of instruction by faculty in the major courses, the quality of the computer labs, etc. The third part of the survey asks the respondent to rank, on a scale of 'Strongly Disagree' through 'Strongly Agree', the degree to which

each of our published program objectives was met in their particular case. The fourth part of the survey asks for general comments and suggestions for improvement. As in the case of the exit survey the raw responses are analyzed and interpreted, and the results discussed by the Undergraduate Studies committee.

5. *Manager/Supervisor-survey*: Each year, alumni of the college who graduated fifteen years ago are asked to complete a survey. The first part of the survey asks the respondent to rank the importance of each of several general educational outcomes (these are the same as in the first part of the alumni-survey) as well as how well the program prepared the OSU graduates that they supervise with respect to these outcomes. The second part of the survey asks for general comments and suggestions for improvement. As in the case of the other surveys, the raw responses are the raw responses are analyzed and interpreted, and the results discussed by the Undergraduate Studies committee.

Next we turn to some changes in the program that have resulted from the feedback received from these various assessments. We should stress that many of these changes are a result of feedback received by several of the involved faculty, over a period of years, rather than being triggered by one observation in one particular assessment using one specific tool.

1. Feedback from SETs and interactions with current and prospective students had made it clear that most incoming students had a fair amount of programming experience. We decided to take advantage of this by having programming experience at the level of CIS 201 as a prerequisite for CIS 221 to allow that course to get into software engineering issues much earlier.

2. Prior to the recent changes in CIS 221, 222, and 321, and the integration of those three courses with each other, CIS 680 was partly used to address software engineering issues. One result of this was a 'split personality' for 680 in that it dealt with design and analysis of algorithms as well as software engineering concerns. While these two topics are certainly related, both students and instructors found that because of having to address software engineering considerations, there was some sacrificing of the depth to which design and analysis of algorithms was being addressed. With the increased software engineering coverage in the redesigned CIS 221, 222, and 321, CIS 680 is currently being modified to focus more heavily on the design and analysis of algorithms. A pilot version of this modified 680 was taught in Winter '99, and this is likely to become permanent shortly.

3. When preparing its recent course group report, the faculty involved with the software engineering group discovered that the 1-credit course CIS 459.22 had not been revised to take advantage of the changes in the introductory-sequence. In particular, student feedback from CIS 459.22 was that it was not building on that sequence. The main reason is that 459.22 was designed when the introductory-sequence used Modula-2 as the programming language. With the change to (a disciplined version of) *C++* in CIS 221, 222, and 321, CIS 459.22 can be revised to deal with more advanced issues such as dynamic memory management. This change in CIS 459.22 will be undertaken shortly. At the same time, we will also make sure that CIS 459.23 (on *Java*) also deals with these issues. This will allow students, in their programming projects in CIS 560 and later courses, to more confidently exploit all these aspects of the languages.

4. CIS 731 is our capstone design course in Artificial Intelligence (AI). Feedback from students, as well as the consultants we had hired to evaluate our program, indicated that the design focus in this course was not as strong as in the other capstone design courses. A major difficulty was that many of the design problems, such as problems in medical diagnosis, that seemed appropriate for the course, required considerable background (that the students did not have) in the problem domain. As a result, the course had to split its attention between this and the design task. Recently, Prof. Rick Lewis, the

coordinator for the course, has been able to identify some problems in the domain of creating and maintaining computer accounts, with which the students are extremely familiar, that can serve as the problem domain for the CIS 731 projects. Prof. Lewis is currently teaching a section of CIS 731 using problems from this domain, and we expect that students in this course will be able to focus more on the design activities. Indeed, some students have had real-life experience (as part-time members of our system staff) in this domain, so they bring those experiences to the classroom. This has also allowed Prof. Lewis to experiment with in-class presentations by students on their analysis of the design problem.

B.3.4 Materials for Review during On-site Visit

Three sets of materials will be available for review during the on-site visit. First, there will be samples of student work from all required courses, as well as several of the elective courses. In addition, for each of these courses, copies of such items as course syllabi, as well as class notes and other materials that the instructor distributes to the students, and copies of mid-terms and final examinations, will be available. Second, there will be a set of materials corresponding to our evaluation mechanisms. These will include summary results from our three different types of surveys. Third, there will be a set of materials to illustrate our processes in action. In particular, copies of minutes of the Undergraduate Studies as well as the Curriculum Committee's minutes will be available. More importantly, the various Course Group Reports will be available.

B.4 Professional Component

This section is organized as follows: In Section B.4.1 we consider each of our approved capstone design courses and show how it meets the requirements of ABET Criterion 4 for a culminating design experience. In Section B.4.2 we show how the program meets Criterion 4 requirements with respect to mathematics and basic sciences, engineering topics, and general education. Table 1 in Appendix IA lists the courses in the order they are normally taken by the students and classified in the appropriate categories; Table 2 in Appendix IA provides course and section size information; Appendix IB contains syllabi for all courses.

B.4.1 Culminating Design Experience

A number of our senior-level courses, each with a strong focus on design tasks, have been designated as capstone design courses. Depending on his or her specific area of interest, each *CSE* major chooses one of these courses to meet the capstone design course requirement of the program. The designation of a specific course as an approved capstone design course is made by the Undergraduate Studies Committee after a careful analysis of the course to see how well it meets the following criteria (approved by the entire faculty):

1. Should be at the senior level.
2. Should include as prerequisites all relevant courses which are part of the *CSE* core or the option for which the design course is intended.
3. Design latitude: Should include evaluation of design alternatives; should be at least somewhat open ended.
4. More than one-half of the course objective should be design:
    - Course objectives should include design in a significant way.
    - Grade plans should base grades at least one third on the design component (in contrast to domain knowledge, pure implementation, etc.)
    - Class time distribution should include at least one hour of lab credit based on a minimum of 3 hours of nominal lab time (usually unstructured). So a 4-credit class should not meet for more than 3 hours per week.
5. Documentation: Project deliverables should include design documentation; the grading scheme should account for the quality of the design documentation/presentation (including evaluation of grammar, organization, clarity, etc.)
6. Desirable features: Oral presentations, use of realistic constraints, etc.

The committee applies these criteria in a flexible manner. Currently, six of our courses have been approved as capstone design courses: CIS 731 (Knowledge-Based Systems), CIS 758 (Software Engineering Project), CIS 762 (Advanced Operating Systems Laboratory), CIS 772 (Information Systems Project), CIS 776 (Hardware / Software Interface Design Project), and CIS 778 (Computer-aided Design and Analysis of VLSI Circuits). Below, we consider each course in turn and explain how it meets the ABET Criterion 4 requirements.

We should first note that these requirements need to be interpreted appropriately when applied to *software* systems, which is the type of system that students design in (most of) the courses described below. Thus we interpret 'economic considerations' as not just dollars and cents but, rather, resource considerations in general; for example, this could include such issues as the amount of memory a system occupies since memory is often a resource in short supply. We interpret 'sustainability' to mean how easy it would be to modify the design of the software system if the requirements were to be slightly changed. And again, to an extent, social issues are involved in the design of user interfaces since good design of these interfaces requires an understanding of the backgrounds and cognitive skills of members of the intended user community and their preferences. One of the most critical professional responsbilities that software engineers have is that of ensuring that their systems function reliably; and reliability is a key requirement for the systems that students implement in these courses.

CIS 731 (Knowledge-Based Systems): Students interested in Artificial Intelligence (AI) choose this course as their capstone design course. The focus in this course is to design *knowledge-based* systems. General AI concepts, as well as basic concepts of building knowledge-based systems are discussed in the prerequisite course CIS 630. In CIS 731, students focus on using these ideas to design and implement a knowledge-based system to solve specific problem in some specific application domain. Students are also expected to rely upon the software design and implementation skills they have acquired in courses such as CIS 560.[10] A typical application domain for the course has been medical diagnosis; the problem with this has been the lack of student-background in the domain. As mentioned earlier, Prof. Lewis, the coordinator for the course, has identified what seems to be a better application domain (from the point of view of this course), indeed one that our students are intimately familiar with. This application is that of creating and maintaining computer accounts in a large system, such as the one our department maintains. The idea is to develop an automated system that starts with the presented symptoms ('I get a directory missing error') and takes the user through an intelligently guided dialogue to pinpoint the cause, then suggest a recommended fix, using the knowledge-based approach.

In their actual projects, students start from a set of sample 'real-world' problem cases provided by the departmental computer staff. The students must then generate a detailed specification for the implementation that consists of a structured description of the solution space, mapping from problem space to the solution space, etc. Students form groups in which they discuss their individual analyses of the problems, with the goal of coming up with a group consensus about the best analysis of this domain. Each group then makes a presentation to the class describing their consensus analysis. Actual implementation of the system is done by individual students. In the future, the course may move toward including teams for this aspect of the project also.

These activities of starting from a loosely-specified requirements, going through a team-based analysis of the problem, to arriving at a design for the system, and then implementing it, are very typical of industry practice for such systems. Further, the key reason for using the knowledge-based approach for designing such systems is that it allows the designer to make decisions efficiently, rather than by searching exhaustively through all possible answers. This makes it possible for such systems to function within specified time constraints, so that these resource considerations are a running theme in the course and in the design tasks. Sustainability is another running theme since an important aspect of these systems is that they allow you to account for cases that may not have been considered during the original design. And since students would already have taken CIS 601, they are expected to take account of social, ethical, and political issues as appropriate; indeed, this applies to all our capstone design courses, not just CIS 731.

[10]Courses such as CIS 560 are not official prerequisites for CIS 731, although most students would have taken these courses by the time they take CIS 731. The reason we do not include these courses as official prerequisites is that occasionally a few graduate students enroll in the course; these students are well qualified for CIS 731 since, although they may not have taken CIS 560 and other similar courses in our department, they have considerable software design and implementation experience.

CIS 758 (Software Engineering Project): This is our most popular capstone design course. The type of system that the students design in this course is very similar to the kinds of systems they are likely to design in their future careers in the software industry. Further, the processes they are must use in arriving at the design also reflect current industrial practice.

The prerequisites for the course are CIS 560 and 757. CIS 560 provides students with a first serious experience in software design and implementation, working in teams, and in producing documentation. CIS 757 provides additional knowledge of software development practices, methods for implementation and maintainence of software, as well as the importance of reliability of software and ways of achieving it. CIS 758 allows students to utilize these skills, as well as other industry-standard approaches, to design a software system to meet given (informal) requirements. A typical example project for this course might be the design of a Customer Support System that a software vendor might use to provide technical and other support for its customers.

The class is organized into project teams of 4 or 5 people each. The teams are required to take account of project management issues, cost-estimation, requirements elicitation, analysis, user-interface design, system design, and implementation planning and scheduling. Each team is required to present its deliverables for each step of the process; each team might also be required to review some documents of some of the other teams. Each team is required to make oral presentations. The teams are also expected to generate and evaluate alternative designs.

CIS 762 (Advanced Operating Systems Laboratory): While most *CSE* students, following their graduation, will be employed in software industries that develop *application* software, a few will be engaged in developing *systems* software, e.g., parts of the underlying operating system. CIS 762 is excellent preparation for such work. Students work with (an educational version of) Unix, an industry standard operating system (OS). Given a new requirement that the OS must meet, students are then required to redesign some part of the system and implement their design to meet this requirement.

The prerequisites for this course are CIS 660 and 662. CIS 660 discusses fundamental operating system concepts, while CIS 662 is a laboratory course where students implement simple system components. CIS 762 is a natural continuation of CIS 662. Students in CIS 762 are expected to solve any problems they encounter when they modify the operating system to implement whatever new feature they are trying to implement. This is a key skill that system designers will need in actual practice and the only way to develop it is by going through a design and implementation experience as in CIS 762. Thus, as we noted, the course provides excellent training for students interested in systems development work following graduation.

Resource considerations are fundamental to all systems work. CIS 762 students become acutely aware of this; in class discussions, resource issues form a running theme. Sustainability is another fundamental aspect of the course since the whole point of the design projects is to modify the existing operating system to meet new requirements. Reliability is yet another critical requirement of operating systems design. But because of the numerous interacting processes in an OS, reliability is also quite difficult to achieve. Students in CIS 762 get first hand experience in this.

CIS 772 (Information Systems Project): One of the fastest-growing areas in computing is the design and development of large information systems, especially web-based systems. CIS 772 gives students the opportunity to design and implement such a system, starting from requirements analysis, through design, implementation, testing, and evaluation, working in a team-setting.

The prerequisites for the course are CIS 560, 670, and either CIS 516 or 757. CIS 772 builds on the software design and implementation experience that students have acquired in CIS 560, the database fundamentals they have acquired in CIS 670, and the software engineering knowledge acquired in CIS 516 or 757. The project that students work on in CIS 772 is very realistic; it requires students to build a system that can be used to maintain a database of information about a fairly large organization (specfically, our department). As the first step, students perform a careful requirements analysis and arrive at a precise set of requirements that their design must meet. Next, students from each team are required to obtain information about the existing database by talking to non-computing people (the office staff of the department). They then design their new system to meet the requirements they have established, and proceed with the implementation. During the implementation, they use industry-standard tools such as Software Through Pictures. They then proceed to test and evaluate their implementation. Members from each team are required to make oral presentations to the entire class on different aspects of their analysis, design, and implementation. Each member is also required to submit written documentation on different aspects of the project.

One of the most common type of work that our graduates will be involved with in the next few years is replacing legacy systems with better designed, more capable, systems. The CIS 772 project is precisely this type of project, and so prepares students for this important type of work. As in all software systems, both reliability and effective use of machine resources are key issues in the project. User interface design questions also play a prominent role in the project, so students need to come up with reasonable designs for this aspect of the project. The interaction with other members of the team, as well as with non-computing people that the project requires, is another important respect in which the course prepares students for work in industry.

CIS 776 (Hardware/Software Interface Design Project): The goal of the project in this course is to get students to call upon both their software and hardware knowledge to design and implement a functional robot. Students working in teams of 3 or 4 students each, design the robot, assemble hardware, instrument sensors, write control software, and test the robot. Near the end of the quarter, each robot is tested to see how well it performs predetermined tasks such as obstacle avoidance. The robots are to be constructed using off-the-shelf components, consisting of lego parts, microprocessor board, assorted sensors, motors, and batteries.

The main prerequisites for the course are CIS 660 and EE 567. CIS 660 prepares students for this design by addressing such issues as synchronization between independent processes, and by discussing techniques for writing software that can interact appropriately with devices (both sensors and motors). EE 567 prepares students for the course by giving them general experience in the design of digital systems. An interesting aspect of CIS 776 is a set of contests in which robots of different teams compete against each other to see which robot functions most effectively. Teams are allowed to purchase and use in their robots, whatever electronic items they wish – but they have to work under a restricted budget ($20 per team). The documentation that the team provides is required to itemize these expenses clearly. Team members are also required to give several presentations during the quarter, addressing such questions as what design alternatives were considered, how the alternatives were evaluated, how the chosen design was implemented, etc.

Autonomous robots, with the controlling software embedded in on-board microprocessors is one of the most interesting applications of computing 'in-the-small'. From the Mars Rover to robots designed to perform hazardous clean-up, if designed well, they can carry out remarkable tasks. But the requirements are demanding; weight, size, and power consumption may not exceed strictly defined limits. CIS 776 gives students a wonderful opportunity to experience designing such systems. Cost considerations are emphasized in a very practical manner. The teams are required to build the robots in such a way that they can easily be disassembled at the end of the quarter, so that the components can be used by students in the next offering of the course; this contributes to improving students' environmental awareness. The use of off-the-shelf components emphasizes easy manufacturability.

CIS 778 (Computer Aided Design and Analysis of VLSI Circuits): The goal of the project in this course is to have students design a number of VLSI circuits using tools similar to those used in industry, and use industry-standard hardware description languages to describe the circuits. Most of our students plan to work, after graduation, in the software industry; CIS 778 is designed for the few who plan to work in hardware design.

The prerequisites for the course are CIS 560, EE 561, and CIS 675. Since one of the important projects in CIS 560 is the design of a simulator of a simple computer, it prepares the students for CIS 778 by equipping them with a programmer's model of the computer; CIS 675 refines this by discussing the underlying implementation at the circuit and register-transfer levels; and EE 561 gives them a thorough grounding in digital design principles. Students in CIS 778 draw on all of this preparation to design two circuits that are specified informally. Students have to first translate the informal specification into a precise one, and then design a circuit that meets the specification, and implement it using standard circuit layout tools. A typical example of a circuit that students might design in CIS 778 is a traffic light controller.

Having students start from loosely defined specifications of the circuits reflects the common situation they will encounter in industry. Moreover, it also gets students to ask themselves who will use the circuit and for what purpose. Thus in the example described above, of a traffic light controller, it gets students to draw upon not just their technical knowledge from courses like EE 561 and CIS 675, but also on their real-life knowledge and experience to arrive at a set of requirements for the circuit that the intended users will approve of. Students learn to appreciate the importance of using standard circuit layout tools and hardware description languages (HDL) (as against designing everything on a custom-basis) and the resulting savings in cost, as well as the easy manufacturability (since once the designs are expressed in the HDL, the actual fabrication can be automated).

B.4.2 Mathematics, Basic Sciences, Engineering Topics, and General Education

B.4.2.1 Mathematics and Basic Sciences

The program requires all students to take a series of mathematics courses starting from Math 151 (Calculus and Analytic Geometry), through Math 152, 153, 254, and culminating in Math 415 (Ordinary and Partial Differential Equations). This sequence provides students with a solid foundation in continuous mathematics, not just for computing but for any engineering discipline. Students are also required to take Math 366 (Discrete Mathematical Structures) to provide the added foundation in discrete mathematics that computing depends upon. Students are also required to take Statistics 427 and 428. This adds up to a total of 33 (quarter) credit hours of mathematics and statistics.

All students are required to take a series of three physics courses, Physics 131, 132, 133. They are also required to take two chemistry courses, Chemistry 121, 125. Both the physics and the chemistry courses involve experimental aspects. These courses prepare students for the engineering courses that they are also required to take. The physics and chemistry courses add up to a total of 24 credit hours of basic sciences. Mathematics and basic sciences together add up to a total of 57 credit hours. Thus the Criterion 4 requirements in this respect are satisfied.

B.4.2.2 Engineering Topics

Students are required to take a range of engineering topics courses, both in CSE and in other engineering disciplines. The relevant CSE courses are CIS 221, 222, 321, 360, 459, 541, 560, 570, 601, 625, 655, 660, 675.01, 680, and one of the capstone design courses CIS 731, 758, 762, 772, 776, or 778. These courses add up to a total of 48 credit hours. Most of these courses include engineering design, the exceptions being CIS 459, 601, 625, and 675.01.

The engineering topics courses from other engineering disciplines that our students are required to take are: Electrical Engineering 261, 206, 300, 309, 320, and 567; Engineering Mechanics 400; Industrial and Systems Engineering 504; Engineering Graphics 166; Materials Science and Engineering 405; Mechanical Engineering 500. Of these, Electrical Engineering 300, Engineering Mechanics 400, Engineering Graphics 166, Materials Science and Engineering 405, and Mechanical Engineering 500, do not have a design component; all the others do. The total of these non-computing courses is 32 credit hours. The combined total of computing and non-computing engineering topics courses is 80 credit hours, of which courses totaling 53 hours include design. Thus the program meets Criterion 4 requirements on engineering topics.

B.4.2.3 General Education

Students are required to take English 110 (5 credit hours) and a second general education course that stresses writing (5 hours). The College of Engineering maintains a list of approved second writing courses that students are required to choose from. These two courses complement the extensive writing that students engage in in CIS 560 (as part of the documentation required for that course); indeed, the second writing course is a prerequisite for CIS 560. These two writing courses help us meet our outcome (3a) (see Section B.3.1), that students will demonstrate an ability to communicate effectively.

Students are required to take courses totaling at least 9 credit hours of Social Sciences. Of these, at least 5 hours must be a course (or courses) that deal with diversity issues; or students may, at their option, take a separate course (at least 5 hours) dealing with diversity issues. Students are further required to take at least 9 hours of Arts and Humanities, and 10 credit hours of Historical Survey. Together, these courses contribute toward meeting our outcome (3b), that students will obtain familiarity with basic ideas and contemporary issues in the social sciences and humanities. To an extent, these courses also contribute toward meeting outcome (3c), that students will obtain an understanding of social, professional and ethical issues related to computing, since many general social considerations also apply to computing. And to an extent, they also contribute toward meeting outcome (2b), specifically the part concerning the ability to work with others, by addressing diversity issues. Thus the general education component of the program complements its technical content and is consitent with the program objectives. Hence the Criterion 4 requirements with respect to the general education component are satisfied.

Tables

Table 1 in Appendix IA lists all the courses that CSE students take, in the order in which they usually take them; the courses are classified in the appropriate categories to indicate how they meet Criterion 4 and Criterion 8 requirements. Table 2 in Appendix IA provides Course and Section Size summary.

Appendix IB provides course syllabi for courses used to satisfy Criterion 4 requirements.

B.5 Faculty

The tenure-track faculty of the CIS Department have a broad range of technical interests and expertise. Particular strength in the department exists in the areas of software engineering, artificial intelligence, graphics, networking, and parallel and distributed computing. Tenure-track faculty each have the Ph.D. in computer science or a closely related subject.

The department's full-time lecturers and senior lecturers have graduate education in computer science, and in some cases the Ph.D. in computer science. They interact regularly with tenure-track faculty on their instructional assignments, and in some cases also participate in research activities.

Part-time lecturers and senior lecturers also have the equivalent of graduate education in computer science, and most have graduate degrees in computer science. They are practicing computing professionals in the Central Ohio community, and bring a practical side to the delivery of the curriculum. This enriches the students' experience, and students often comment positively about this on their class evaluations (SETs, see Section B.3.3).

Upon admission to the major, each student is assigned a faculty advisor. While registration each term does not require that a student meet with an advisor, many do so, and in any case, advisors are available to meet with students to give both academic and career counseling. Many faculty also interact with undergraduate students as part of our diversity enhancement program. This program, among other aspects, provides incentives for undergraduate students to engage in research-related projects with faculty. There also are opportunities for students to do independent study work with faculty. Furthermore, there are student chapters of the ACM, IEEE-CS, and UPE, each with faculty advisors. Finally, the department has a faculty member as industry coordinator. This individual helps promote interaction between the department and industry, including assisting in arranging student visits to company installations.

The size of the faculty is adequate to deliver the program, because the size of the program is regulated to be sensitive to the faculty size and other resources. The university has approved an enrollment management plan, which restricts entry into the major based on a GPA cutoff, which is set each year based on financial and personnel resource availability (see also Section B.1.2).

B.6 Facilities

The CIS Department's computing facilities provide access to modern Unix and NT systems in all faculty and staff offices and in all student laboratories used by students in the program. The Department maintains facilities for the program that include 130 seats for open lab use, and an additional 20-seat closed laboratory that is used for the introductory computer science sequence (and for other courses on occasion). Remote access to the laboratory stations is available, and students in the program can use special dial-in numbers that are not available to non-majors. Thus, they do not compete with the rest of the students on campus for access to the systems.

The CIS network includes both ATM and fast ethernet, with high-powered servers for both compute power and file management. Printing facilities exist in each lab (except for the closed lab, whose print facility is just down the hall). Appropriate software resides on the servers and is accessible by all accounts from any lab seat or office.

CIS controls three classrooms that have full internet access, including the closed laboratory classroom used by the introductory sequence. Other classrooms assigned to our course have overhead projectors as standard equipment. The Department also maintains laptops and laptop projectors that can be reserved by faculty to give, for example, power point presentations or other demos.

CIS's own 13-person computing staff maintains the facilities used in the program by faculty, staff, and students. Because they are employees of the Department, they can be more responsive to the needs of the program and can plan upgrades and maintenance activities to minimize disruption to the program.

The Department's Computer Committee is responsible for computing policies and plans. The director of the Department's computing staff, and representatives from the student community, sit on this committee with department faculty to ensure that the different perspectives are represented when setting these policies and plans.

B.7 Institutional Support and Financial Reserves

Budgets for the CIS Department are set by the Dean of the College of Engineering. The Department has a separate budget for its computing facilities management, and this budget is managed by the computing facilities director. Beginning in FY2001, the College (and all colleges at Ohio State) will be budgeted by a process that more closely reflects actual enrollments. The Dean has indicated that his method of allocating his budget to the departments will largely reflect the method by which he received his budget from Central Administration.

All levels of the university administration are supportive of our desire to provide a quality program to the students. Toward that end, they supported the reinstitution of enrollment management when demand for the major increased significantly. This ensures that students admitted to the program will be able to get the courses they need and allows the department to keep classes supported in a manner consistent with its faculty and financial resources.

The university supports a faculty professional leave (sabbatical) activity after seven years of service. Most faculty take advantage of this opportunity. The Department Chair also routinely supports, from the department's budget, travel by faculty to professional meetings, including workshops that promote effective teaching.

Resources to acquire, maintain, and operate the computing facilities in use by the program are contained in the department's computing facilities budget. This budget includes a $200K equipment line. In addition, the students in the program pay a computing fee, which is returned to the department to purchase hardware, software, and access to the labs. Approximately $150K per year is in this budget.

The Department maintains an advising staff consisting of a full-time professional and two half-time graduate students. These individuals advise students prior to their becoming majors. There are 7.5 additional administrative staff who provide various kinds of support to the Department and the program.

The College of Engineering also maintains an advising staff, a coop office, and a career services office that are of great value to students in the college, and hence, in our program. Standard support functions exist on an institution-wide basis to meet other student needs.

B.8 Program Criteria

In this section we consider the program criteria applicable to "electrical, computer, and similarly named engineering programs", and show how our program meets each criterion.

Knowledge of probablity and statistics: The program requires all students to take Statistics 427 and 428. Stat 427 (Introduction to Probability and Statistics for Engineering and the Sciences I, 3 credit hours) introduces students to probablity, discrete and continuous random variables, and sampling. Stat 428 (Introduction to Probability and Statistics for Engineering and the Sciences II, 3 credit hours) continues the discussion from the first course, and ensures that students have a good understanding of probablity and statistics.

This knowledge is applied to computing related problems in the required courses CIS660 (Operating Systems), and CIS680 (Design and Analysis of Algorithms). Statistical considerations are important in scheduling activities such as disk arm access, as well as process scheduling, and these are important topics in CIS660. Analyzing running-times of programs under various ('best case', 'average case', and 'worst case') assumptions is a central topic in CIS680 and statistical considerations are a key part of this analysis. CIS680 also exposes students to analyzing the behavior of probablistic alogrithms, thereby making effective use of the knowledge of probablity and statistics that students have acquired in Stat 427, 428.

Knowledge of mathematics through differential and integral calculus: All students are required to take a series of mathematics courses starting from Math 151 (Calculus and Analytic Geometry), through Math 152, 153, 254, and culminating in Math 415 (Ordinary and Partial Differential Equations). This sequence, adding up to a total of 24 credit hours, provides students with a solid foundation in continuous mathematics.

Knowledge of basic sciences: All students are required to take a series of three physics courses, Physics 131, 132, 133. They are also required to take two chemistry courses, Chemistry 121, 125. Both the physics and the chemistry courses involve experimental experience. These basic sciences courses add up to a total of 24 credit hours of basic sciences.

Knowledge of engineering sciences: Students are required to take a range of engineering sciences courses, both in CSE, as well in other engineering disciplines. The relevant CSE courses are CIS 221, 222, 321, 360, 459, 541, 570, 601, 625, 655, 660, 675.01, and 680. These add up to a total of 40 credit hours.

The engineering topics courses from other engineering disciplines that all our students are required to take are: Electrical Engineering 261, 206, 300, 309, 320, and 567; Engineering Mechanics 400; Industrial and Systems Engineering 504; Engineering Graphics 166; Materials Science and Engineering 405; Mechanical Engineering 500. The total of these non-CSE-engineering sciences courses is 30 credit hours. The combined total of computing and non-computing engineering sciences courses is 70 credit hours.

Several of the listed computing courses including CIS 221, 222, 321, 541, 570, 655, and 680 require students to use the knowledge gained from the mathematics, basic sciences, and engineering sciences courses to design and analyze software components and systems of varying degrees of complexity. Similarly, EE 206 and EE 309 require students to use the knowledge gained in these coures to design and analyze digital and analog circuits. In addition, CIS 560 (5 credit hours) which is required of all students, focuses on a large scale software design task appealing to all of the knowledge in the previous courses; and EE 567 focuses on using the knowledge gained in the previous courses to explore the design of digital systems. And the the capstone design courses, one of which each student is required to take, uses the knowledge gained in the courses throughout the curriculum to focus on quarter-long design tasks. Most of our capstone design courses focus on the design of software but, for students interested in both hardward and software, CIS 776 (Hardware/Software interface design project) provides students an opportunity to design a system with both hardware and software components.

Thus the program meets the requirements of Criterion 8 concerning knowledge of mathematics, basic sciences, and engineering sciences necessary to analyze and design complex electrical and electronic devices, software, and systems containing hardware and software components.

Knowledge of discrete mathematics: All students are required to take Math 366 (3 credits) which focuses on discrete mathematics. In addition, students obtain a thorough understanding of and facility with ideas in boolean logic and their application to digital circuits in courses such as EE 206, 261, and CIS 360. The students knowledge of discrete mathematics is put to use and further developed, in a variety of computing courses including CIS 321 (Math 366 is a co-requisite for this course), CIS 570, 625, 655, and 675.01.

Thus the program meets the requirements of Criterion 8 concerning knowledge of discrete mathematics.