Chapter XIII
Software Engineering Accreditation in the United States

James McDonald
Monmouth University, USA

Mark J. Sebern
Milwaukee School of Engineering, USA

James R. Vallino
Rochester Institute of Technology, USA

ABSTRACT

This chapter provides a brief history of the accreditation of software engineering programs in the United States and describes some of the experiences encountered by programs in achieving their accreditation and by program evaluators in reviewing those programs. It also describes how the accredited programs have addressed the most difficult issues that they have faced during the accreditation process. The authors have served as leaders of the accreditation efforts at their own institutions and as ABET program evaluators at several other academic institutions that have achieved accreditation. The objective of this chapter is to provide those software engineering programs that will be seeking accreditation in the future with some of the experiences of those who are familiar with the process from both the programs' and the evaluators' points of view. Leaders of programs that are planning to request an accreditation review will be well prepared for that review if they combine the information contained in this chapter with the recommendations contained in Chapter XIX of this text.

INTRODUCTION

The history of software engineering education dates to the generally accepted origin of the software engineering discipline in 1968. This year is associated with the first NATO conference on software engineering in Garmisch, Germany. Tomayko (1998) points out, however, that the same year also marked what is apparently the first offering, by Douglas Ross at the Massachusetts In-

stitute of Technology, of an academic course with the term "software engineering" in its title. For a variety of reasons, considerable time passed before courses with significant software engineering content became more common (Tomayko, 1998; Duggins 2002). Beginning in 1977, a number of graduate programs in software engineering were developed and began operation, including those at Seattle University, Texas Christian University, and the Wang Institute of Graduate Studies (Tomayko, 1998). At the undergraduate level, a number of computer science and computer engineering programs incorporated one or two courses in software engineering, typically taught using survey textbooks that offered reasonable breadth but relatively little depth. Although undergraduate software engineering programs began to emerge internationally as early as 1985 (Joint Task Force on Computing Curricula, 2004), it was not until 1996 that the Rochester Institute of Technology initiated what was to become, in 2003, one of the first four software engineering programs to receive accreditation in the United States; the other programs in this group were offered by Clarkson University, Milwaukee School of Engineering, and Mississippi State University.

While we recognize that software engineering programs in other countries have been accredited by accrediting agencies in those countries, this chapter addresses only the history and experiences of software engineering programs that have achieved accreditation in the United States. It is hoped that the material presented here will be of value to software engineering educators in both the United States and around the world.

ABET AND ENGINEERING PROGRAM ACCREDITATION

ABET, Inc., formerly known as the Accreditation Board for Engineering and Technology, is the recognized accreditation body in the United States for college and university programs in applied science, computing, engineering, and technology. It is a federation of professional and technical societies (28 at present) representing those fields. ABET accreditation activities are managed by four commissions; the two most directly related to software engineering are the Engineering Accreditation Commission (EAC) and the Computing Accreditation Commission (CAC). Like other engineering disciplines, software engineering falls under the EAC, while the CAC is responsible for computer science, information systems, and information technology. In possible contrast to some other disciplines, accreditation has historically been an expected attribute of United States engineering programs, and is thus an important concern for software engineering educators.

Each discipline has an associated "lead society", which is one of the member societies of ABET. This society has primary responsibility for defining discipline-specific accreditation criteria, as well as for selecting, training, and evaluating program evaluators. Initially, the lead society for software engineering was the Institute of Electrical and Electronic Engineers (IEEE), which prepared the original version of the software engineering program criteria (Engineering Accreditation Commission, 1999, p. 47), discussed later in this chapter.

With the integration of ABET and the Computing Sciences Accreditation Board (CSAB) in November 2001, CSAB took over the role of lead society for software engineering, and the IEEE became a "cooperating society." Unlike the IEEE and most other member societies of ABET, CSAB is not itself a membership society. Instead, the current members of CSAB are three other professional societies: the Association for Computing Machinery (ACM), the IEEE Computer Society (IEEE-CS), and the Association for Information Systems (AIS).

From the point of view of a software engineering program seeking initial accreditation, the process begins with a request for evaluation,

which must be submitted by January of the year in which an evaluation visit is being requested. Since ABET policies require that a program have at least one graduate at the time of the evaluation visit, the request for evaluation is generally submitted in the year when the first graduates are anticipated.

Of course, the work of program and curriculum definition must begin much earlier. It is common for program faculty to attend ABET faculty workshops and to send representatives to training sessions for ABET program evaluators, in order to gain familiarity with the accreditation criteria, process, and practices. The program must also define its educational objectives and outcomes, discussed in more detail below.

Once the request for evaluation has been submitted, the next task is to complete the self-study report, which provides detailed data and evidence to show that the program meets the applicable accreditation criteria. The self-study report is based on an ABET-provided template (Engineering Accreditation Commission, 2007a) and must be submitted by the end of June during the year in which the request was made.

The evaluation visit takes place in the fall. The visiting team consists of a team chair (usually a member of the EAC) and at least one program evaluator (PEV) for each program to be evaluated. The minimum team size is three members (ABET, 2006, p. 8), so it is possible that two program evaluators may be assigned to a single program if no other program is being evaluated during the same visit. Prior to the visit, the program evaluator examines the self-study report and related materials such as student transcripts. Ongoing communication with the program leadership helps to resolve as many issues as possible before the team arrives on campus. During the visit, the evaluator interviews faculty members and students, examines additional materials such as examples of student work, evaluates facilities, and gathers any other necessary information.

During an exit session at the end of the visit, the accreditation team provides the institution with a summary of its evaluation. After the visit, the program has the opportunity to submit additional evidence, primarily to address any shortcomings that were identified during the visit. The team chair and program evaluators then prepare a draft statement of their findings, which is sent to the institution for comment. The final version of the statement incorporates any changes resulting from the institution's "due process" response and is sent to the EAC for final action during the summer after the visit. If accreditation is granted, it is common practice to extend accreditation retroactively to the prior year graduates, since it was their work and curriculum that were examined during the accreditation review.

CRITERIA FOR ACCREDITATION

The current engineering accreditation criteria (Engineering Accreditation Commission, 2007) are based on a major revision originally known as Engineering Criteria 2000 (often abbreviated as "EC2000" or "EC2K"). Prior versions of the criteria focused on detailed prescriptions and, in the view of many engineering educators, limited opportunities for flexibility and innovation. The revised criteria adopted an approach of setting general goals and assigning to individual programs the responsibility for demonstrating achievement of those goals through appropriate assessment and evaluation.

Each of the ABET criteria for accrediting baccalaureate-level engineering programs addresses a specific area of concern. During 2007, changes to the numbering and organization of the criteria were proposed, as indicated in Table 1; these changes will take effect for the 2008-2009 accreditation cycle.

Despite the change in organization, the content of each of the areas of concern has remained fairly

Table 1. Areas of concern covered in each ABET criterion

Area of Concern	Criterion (2007-2008)	Criterion (2008-2009)
Students	Criterion 1	Criterion 1
Program Educational Objectives	Criterion 2	Criterion 2
Program Outcomes	Criterion 3	Criterion 3
Continuous Improvement	Criteria 2-3	Criterion 4
Curriculum	Criterion 4	Criterion 5
Faculty	Criterion 5	Criterion 6
Facilities	Criterion 6	Criterion 7
Support	Criterion 7	Criterion 8
Program Criteria	Criterion 8	Criterion 9

stable from the introduction of the EC2000 criteria until the present time. The criteria are:

Students. For historical reasons, the criteria first address the relationship between an engineering program and its students, even though logically it would make more sense to begin with the program educational objectives and outcomes. Programs are required to evaluate students and monitor their progress, while providing both curricular and career advising. Specific note is made of the need for effective policies and procedures for the admission of transfer students, granting of transfer credit, and verification that all students meet all program requirements.

Program Educational Objectives. Since the initial introduction of the EC2000 criteria, there has been a continuing evolution and clarification of the terminology used to specify the results that an engineering program strives to achieve. By the current definition, the program educational objectives deal with the broad career and professional accomplishments for which graduates are being prepared. It is common for the program leadership and faculty to consult with employers and other stakeholders to ensure that the program objectives accurately reflect the environment in which the program's graduates will work. Since these achievements relate to performance after graduation, the program's success in this regard cannot, in general, be determined until some time has passed. Even then, it may be difficult to assess the program's contribution to the individual graduate's success in meeting these longer-term objectives.

A program's educational objectives are expected to be consistent with its institutional mission and to communicate its specific goals to potential students and to the public at large. A typical program objective might be, "Graduates of the program are expected to obtain employment in the software development industry and/or to enter graduate school within six months after graduation."

Program Outcomes. To complement the program educational objectives, programs are also required to define and assess program outcomes, which are narrower statements that describe the knowledge and skills expected of students at the time of graduation. The underlying assumption is that this knowledge and skill will provide the basis for achievement of the longer-term career and professional achievements. This criterion requires that a set of eleven specific outcomes be incorporated (often referred to as "a-k" because of

the way they are enumerated), but programs are free to articulate additional outcomes. A typical outcome is: "By the time students have graduated from the program they must demonstrate the ability to apply knowledge of mathematics, engineering and science," which is outcome a) in the specific list of outcomes.

Historically, programs have been encouraged to formulate their own outcomes based on their specific program objectives. These program-specific outcomes are often designed to incorporate the "a-k" outcomes. For example, a software engineering program might adopt a program outcome related to designing software components and systems, implicitly referencing the "3(c)" outcome that deals with designing a system, component, or process within realistic constraints.

However, defining a complete set of program-specific outcomes can also mean extra work for the program in preparing for an accreditation visit, since it is then necessary to demonstrate student achievement of both the "a-k" and the additional "program-defined" outcomes. One alternative is to augment the standard "a-k" outcomes by articulating a small number of additional outcomes, if the program judges that the generic outcomes are not sufficient. The proposed 2008-2009 engineering criteria omit a previous requirement that the program must "formulate program outcomes" related to the program objectives, perhaps suggesting a shift away from program-specific outcomes.

Continuous Improvement. The requirement for ongoing actions to improve the program, previously called out in the context of program objectives and outcomes, has become a separate criterion in the proposed 2008-2009 draft. Programs are required to show evidence for these actions, which are expected to be based on the results of assessment and evaluation processes called for in the criteria related to program objectives and program outcomes.

Curriculum. This section of the engineering criteria has two major parts. The first deals with minimum standards for curriculum content. The curriculum must include at least one year (typically 32 semester credits or 48 quarter credits) of college-level mathematics and basic sciences. At least some of the basic sciences course work must include experimental experience. A minimum of one and one-half years (48 semester credits or 72 quarter credits) of engineering topics is also required. The engineering topics consist of engineering sciences and engineering design. The curriculum is also required to incorporate a general education component that complements the technical content, but no quantitative specifications are mandated for this component.

One question for software engineering programs is whether some computer science content can be used to meet the "mathematics and basic science" requirement. This type of accounting seems quite reasonable, since the relationship between computer science and software engineering resembles that between, for example, physics of mechanics and mechanical engineering. In addition, many computer science topics are mathematical in nature. However, there is at present no explicit policy on this matter, so many programs have taken a defensive position that ensures the credit requirement is met using course content consistent with a more traditional definition of mathematics and basic science.

The second part of the curriculum criterion imposes a requirement that students be prepared for engineering practice through the curriculum and that this course work culminate in a major design experience that incorporates engineering standards and multiple realistic constraints. The requirement for a major design experience is often addressed by a "senior design project" course or course sequence.

Faculty. The criterion related to the program faculty addresses three primary concerns. First, the number of faculty members and their competencies must be sufficient to cover all curricular areas of the program, while also assuring that faculty members have time to advise and in-

teract with students, support university service activities, continue their own professional development, and maintain links with practitioners and employers.

Second, the program faculty must be invested with sufficient authority to provide effective guidance for the program and to define and execute processes for assessment, evaluation, and continuous improvement of the program's objectives, outcomes, and curriculum.

Third, the criterion provides guidance for evaluating the competence of the faculty, citing factors such as education, diversity, engineering experience, teaching effectiveness, communication ability, scholarship, participation in professional societies, and professional engineering licensure. In addition to these traditional measures, the criterion also makes explicit the need for "enthusiasm for developing more effective programs" (Engineering Accreditation Commission, 2007, p. 3), perhaps recognizing the personal and communal investment that is required to institute and maintain effective assessment and improvement processes.

Facilities. Programs are required to ensure that classrooms, laboratories, and equipment are adequate and that they provide an atmosphere conducive to learning, foster student-faculty interaction, and support professional development and activities. Students must have opportunities to learn the use of modern engineering tools and adequate computing facilities must be available to support both students and faculty.

Support. Programs must have, and demonstrate that they have, the institutional support and financial resources needed to maintain the faculty and facilities. This criterion also explicitly requires adequate support personnel and institutional services. Specific mention is also made of the need for "constructive leadership" to assure the quality and continuity of the program.

Program Criteria. The general engineering accreditation criteria are intended to apply across widely disparate engineering disciplines. While this commonality and consistency is valuable, it is also understood that each discipline may have its own specific requirements. To address these issues, the engineering criteria incorporate sets of program-specific criteria, which are (at least nominally) limited to curricular topics and faculty qualifications. The applicability of a given set of program criteria is determined by the name of the program; for example, a program in "computer and software engineering" would be expected to meet the program criteria for both computer engineering and software engineering. When multiple sets of program criteria are applicable, overlapping requirements need only to be satisfied once. The program criteria for software engineering are discussed in the following section.

PROGRAM CRITERIA FOR SOFTWARE ENGINEERING

As noted above, program criteria are limited to curricular topics and faculty qualifications. The curriculum-related portion of the current software engineering program criteria (Engineering Accreditation Commission, 2007, p. 18) states two primary requirements.

First, the curriculum is required to provide breadth and depth across the range of engineering and computer science topics implied by the title and objectives of the program. Except in unusual cases (e.g., a program that focuses on applying software engineering to aeronautics or to financial modeling), this will normally imply compliance with an accepted "community" definition of the software engineering discipline. Two such definitions are given in the Guide to the Software Engineering Body of Knowledge (2004) and in the undergraduate software engineering curriculum guidelines prepared by the Joint Task Force on Computing Curricula (2004).

Second, the curriculum section of the program criteria for software engineering requires that the

program demonstrate a number of specific student outcomes. While these mandated outcomes are not really "curricular topics", there is precedent for requirements of this type in the program criteria for many other disciplines (Engineering Accreditation Commission, 2007, pp. 5-18).

The software engineering program criteria require the program to demonstrate that graduates have the ability to analyze, design, verify, validate, implement, apply, and maintain software systems. Although the term "analyze" has a generic engineering meaning, in this context it is generally understood to refer to requirements analysis. Graduates must also be able to apply, in the context of complex software systems, discrete mathematics, probability, statistics, and relevant topics in computer science and supporting disciplines.

Additionally, the program must demonstrate that graduates have the ability to work in one or more significant application domains. In itself, this requirement does not dictate any particular curricular content, but it does imply some coursework or other experience beyond core software engineering and computer science topics. Some existing software engineering programs have chosen to require specific courses in one or more application domains such as embedded software, gaming software or web applications. Other programs have defined a set of elective course sequences, in a variety of areas, allowing students to choose according to their own interests. A few programs have adopted both of these strategies.

In regard to faculty qualifications, the current program criteria for software engineering do not impose any additional requirements. Effective for the 2001-2002 accreditation cycle, the program criteria were amended to require that "those faculty teaching core software engineering material have practical software engineering experience" (Engineering Accreditation Commission, 2000, p. 16), but that section was later deleted (Engineering Accreditation Commission, 2002, p. 22) with little public explanation for the change.

GROWTH OF ACCREDITED SOFTWARE ENGINEERING PROGRAMS

The first undergraduate program in software engineering in the United States was started in 1996 at Rochester Institute of Technology. Since that program took root and showed the viability of an undergraduate software engineering program, there has been a steady growth in the number of programs, with several new ones started each year. This has happened despite the general downturn in undergraduate computing program enrollments since 2000 (Computing Research News, 2007). There are currently 35 programs leading to an undergraduate degree in Software Engineering. Through the summer of 2007, fifteen of these programs have been accredited by ABET. The Rochester Institute of Technology program graduated its first class of baccalaureate-level software engineers in May 2001. The first four programs applying for accreditation had their campus visits in fall of 2002, and received accreditation approval in the summer of 2003. The EAC granted the Rochester Institute of Technology program an extended grandfathering which covered their May 2001 class. That gave the program the distinction of awarding the first ABET accredited BS in Software Engineering degrees. Figure 1 shows the growth in both the total number of undergraduate software engineering programs and the number of accredited programs.

CURRENTLY ACCREDITED PROGRAMS

Table 2 lists the fifteen software engineering programs accredited by ABET as of 2007. All

Figure 1. Number of undergraduate software engineering programs

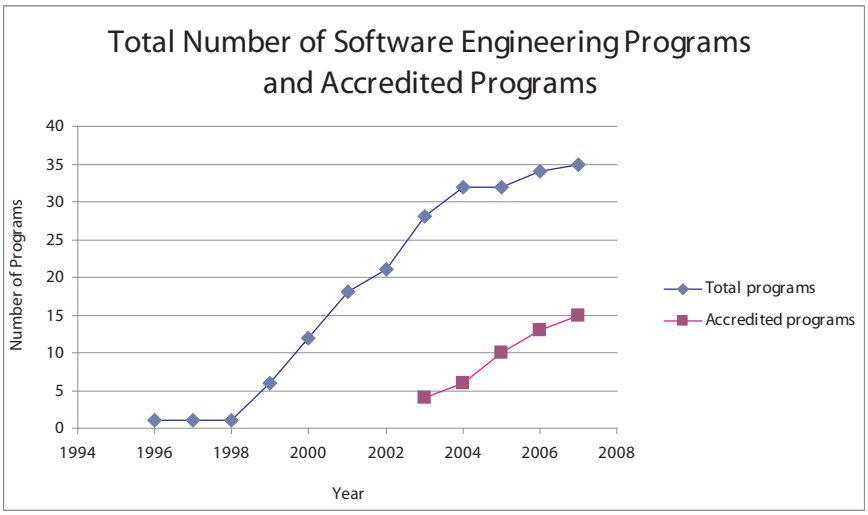

Table 2. Year when program was accredited

Name of Institution	Year Accreditation Awarded
Auburn University	2005
Clarkson University	2003
Embry-Riddle Aeronautical University (Florida)	2005
Fairfield University	2006
Florida Institute of Technology	2004
University of Michigan-Dearborn	2005
Milwaukee School of Engineering	2003
Mississippi State University	2003
Monmouth University	2005
Penn State University – Erie	2006
Rochester Institute of Technology	2003
University of Texas at Arlington	2004
University of Texas at Dallas	2006
Rose-Hulman Institute of Technology	2007
University of Wisconsin - Platteville	2007

of these programs award a Bachelor of Science degree in Software Engineering. The programs have a range of student populations from 30 to over 400.

EXPERIENCES OF PROGRAMS AND PROGRAM EVALUATORS

The authors have completed informal on-line surveys of both software engineering programs that have been accredited by ABET and the ABET program evaluators who have been involved in reviewing those programs. We have supplemented the data gathered in those surveys with our personal experiences as program evaluators and as program leaders to characterize the experiences of programs that have been accredited.

Both programs and program evaluators report that the programs that have been accredited have typically had little difficulty meeting the requirements of the **Facilities** and **Support** criteria. However, both programs and program evaluators report that several programs have had to take action, sometimes significant action, to meet the requirements of the **Students, Program Educational Objectives, Program Outcomes** and **Curriculum** criteria**.** Survey results indicate a few cases of disagreement, or even contention, between programs and program evaluators, specifically in the areas of faculty qualifications and curricular topics. The next two sections of this chapter highlight evaluation findings related to the criteria that have resulted in improvement actions by the software engineering programs and those criteria which have caused some tension between programs and their evaluators.

CRITERIA RESULTING IN IMPROVEMENT ACTIONS BY PROGRAMS

Many programs reported that they have adopted automated grade tracking and degree audit systems that are being used to replace some regular face-to-face student advising. This has made it more difficult to demonstrate that student progress is being properly evaluated and monitored by the faculty for conformance to program requirements as required by the **Students** criterion. A few programs found that they were not advising and monitoring their students carefully enough. This sometimes resulted in students not completing all of the courses required by the program, usually due to course substitutions that were done without appropriate review. The programs that have had this problem have generally tightened their advising, monitoring and course substitution approval processes.

Most programs have had difficulty meeting the **Program Educational Objectives** criterion. These objectives represent achievements that students would be expected to reach after graduation. As such, the data are not under the program's direct control. One program commented:

"Assessing educational objectives is difficult. You must rely on outside information to get assessment data, and it is difficult to get enough results to make a reasonable measurement. Traditional alumni survey completion rates are very low and when the number of graduates is relatively low, it is difficult to get enough data from alumni survey results. Employer surveys are equally difficult to get unless you have dedicated employers that hire a large number of your graduates."

The **Program Educational Objectives** criterion requires a process, based on the needs of the program's constituents, in which the objectives are determined and an ongoing evaluation of the extent to which the objectives are being attained,

the result of which must be used to improve the program.

Programs have sometimes created their **Program Educational Objectives** without the involvement of the program's constituencies or, in a few cases, without even explicitly defining those constituencies. To avoid this problem, successful programs have usually defined their constituents very explicitly in their self-study report. The constituents described are usually the program's students, the program's faculty and an industrial advisory committee representing potential employers of the program's alumni. Some programs have added parents of students, administrators of the institution and the state or region's economy. Reasonable and acceptable **Program Educational Objectives** have typically been created by first having the faculty draft a set of six to eight specific things that they would expect their graduates to achieve within a few years after graduation. Then these objectives are discussed with, and perhaps modified by, an industrial advisory committee, after forming such a committee if one doesn't already exist. A description of the interaction with constituents is documented and the objectives are published, usually in the institution's catalog, on the program's web site and in any materials being used to market the program. Some have developed employer surveys to get feedback on achievement of **Program Educational Objectives** and a few have modified the wording of their educational objectives to eliminate misunderstandings of the wording.

Most programs seeking initial accreditation have found it very difficult to measure achievement of their objectives by the time of the first evaluation visit, which usually occurs in the fall after the first alumni have graduated from the program. About the only thing the program can practically do within those few months is to informally speak with members of their industrial advisory board who may have hired the program's first graduates to get feedback on their opinions about the students' likelihood of meeting the objectives. Some programs have put off this step until several months after the visit and simply describe what the program is planning to do to evaluate achievement of the objectives.

In the period following the introduction of the EC2000 criteria, a common source of difficulty was confusion among program leaders and program faculty about the differences between educational objectives and program outcomes. Self study reports frequently made the objectives and the outcomes sound very similar to each other. Sometimes programs have used the same set of capabilities in describing the objectives and the outcomes and have simply grouped them in different ways. The intent of the ABET criteria is that the objectives and the outcomes are clearly different things. The easiest way to distinguish them from each other are that the outcomes should be things that students are expected to achieve by the time they graduate while the objectives are career and professional accomplishments which they would be expected to achieve after graduation. As time has passed program leaders and faculty seem to have become more familiar with this distinction and the confusion has been diminishing.

Some programs and evaluators noted issues with the **Program Outcomes** criterion. One program, which was using student portfolios as the primary method for assessing outcomes, augmented their collection and evaluation of student portfolios based on suggestions made by the program evaluator. This augmentation involved developing very explicit instructions for students describing what they should include in their portfolios, how it should be organized and a rubric for use by the faculty describing how to evaluate the portfolio contents.

With regard to the specific "a-k" outcomes, some programs expressed difficulty sufficiently demonstrating achievement of: f) an understanding of professional and ethical responsibility; h) the broad education necessary to understand the impact of engineering solutions in a global, economic, environmental, and societal context;

i) a recognition of the need for, and an ability to engage in life-long learning; and, j) a knowledge of contemporary issues. They have usually developed additional methods for measuring these outcomes and sometimes have developed new courses or added content to existing courses.

Some programs have had difficulty in complying with the requirements of the **Curriculum** criterion related to the culminating major design experience. This program component must provide a significant software engineering design experience to each student. In some cases this "capstone" experience may fall more into the realm of research than design or fail to incorporate appropriate engineering standards and constraints. Programs encountering this problem have had to develop methods to ensure that their projects have significant design content, that the work was clearly and completely documented, and that engineering standards and constraints were appropriately considered.

THE MOST DIFFICULT ISSUES

While the survey results indicated a good deal of agreement between program leaders and program evaluators, there were some exceptions. Specifically, there was some evidence, of inconsistency, and even some contention, related to faculty qualifications and curricular content.

Program leaders generally reported no problems related to faculty qualifications. However, several program evaluators expressed concerns regarding a low proportion of faculty with true breadth and depth of experience in software engineering. This issue seemed to arise primarily in software engineering programs housed in computer science departments. As one evaluator stated, "It is sometimes difficult to agree with established CS programs adding an SE program that they have sufficient breadth and stability in SE to satisfy the ABET criteria."

Another concern of some evaluators related to the isolation of some software engineering faculty members, who seemed to have little involvement with the software engineering practitioner community and with the software engineering education community.

PEVs noted a need for all faculty to be aware of and be involved with the ABET/EAC procedures and self-study preparation. The problem most frequently observed across all criteria has been defining appropriate and viable assessment and evaluation processes. Even when adequate processes have been defined, PEVs often identify problems with faculty compliance. To satisfy the requirements of outcomes assessment, the program faculty members must be committed to ongoing execution of the defined processes. Most programs and evaluators understood that the **Outcomes Criterion** requires the direct measurement of student outcomes via capstone projects, portfolio evaluations or specific quiz or exam questions. However, almost all agreed that the overhead required to do this rigorously placed a high burden on the programs, particularly for programs that had decided to evaluate all outcomes and all students every year.

As noted previously, the software engineering **Program Criteria** require appropriate curricular content. Several evaluators commented that there were problems with programs' interpretations of the breadth and depth of software engineering material required to satisfy these criteria. They said that these problems have most frequently been seen when programs are developed from a base of a computer science or a computer engineering curriculum.

Two programs reported that they have had problems with a specific program evaluator's interpretation of the requirements related to **Program Criteria**. These evaluators, they say, were looking for coverage of a specific topic area, such as software evolution, as part of the maintenance activities which students are required to be able

to do by the time they graduate according to this criterion.

In the case of programs that have had problems with curricular content, faculty members have sometimes felt that they were already covering many of the required software engineering topics. By requiring students to take specific existing computer science courses and adding a software engineering capstone course to the curriculum, they felt that they would meet the breadth and depth requirements.

The programs that have been most successful in satisfying the curriculum requirements of the program criteria have linked their curricula to accepted frameworks such as the Guide to the Software Engineering Body of Knowledge (2004) and Joint Task Force on Computing Curricula (2004) and have made these links explicit in their course syllabi, by describing which courses cover which topics outlined in those documents. The number of specific software engineering courses in these programs usually ranges from six to twelve. Typically those courses cover 50% to 80% of the topics specified in the referenced documents.

While the program criteria do require breadth and depth of software engineering content, it is not necessary that these topics be covered in specific "software engineering" courses. However, if this content is embedded in other (e. g., computer science) courses it must be very clear from the course syllabi and from the work done by students that the software engineering topics are, in fact, being covered. It is a common expectation that at least some of these courses employ textbooks that address a variety of advanced software engineering topics, and that they do not rely primarily on the small number of commonly used introductory software engineering textbooks.

IMPROVEMENTS MADE

The variety of improvements that have been made as a result of assessment and preparation for accreditation visits is extremely long. This section will summarize a subset of those with which the authors are familiar.

For the requirements related to **Students,** a few programs have found that they were not advising and monitoring their students carefully enough. This sometimes resulted in students not completing all of the courses required by the program, usually due to course substitutions that were done without appropriate review. The programs that have had this problem have typically tightened their advising and monitoring processes to insure that the problem does not happen in the future.

Several programs have formed new industrial advisory committees and gotten them deeply involved in helping to specify **Program Educational Objectives**. A few have developed employer surveys to get feedback on achievement of program educational objectives and at least one has modified the wording of its objectives to eliminate misunderstandings of the wording. Based on our experience, with our own programs and with programs that we have evaluated, we believe that the greatest benefits to the programs have been the improved relationships between the programs and local industry that have resulted from the involvement of industrial advisory committees in the accreditation process.

In response to shortcomings identified in the **Program Outcomes** area and to the measurement of specific outcomes, many programs have modified the content of specific courses, usually with small changes to assure that prerequisite courses were meeting the expectations of instructors in later courses. Some programs have developed specific courses to assure that students were developing an understanding of professional and ethical responsibilities. Others have developed new methods and courses for assuring that students were receiving a broad education, recognizing the

need to engage in life long learning and developing an understanding of contemporary issues. All of these improvements were made as direct results of measurements indicating that student learning results were below expectations for one or more of the specified outcomes.

To effectively demonstrate compliance with the requirements for a major design experience by the **Curriculum** criterion, some programs have provided additional encouragement for students to document their engineering processes, design approaches and their consideration of engineering standards and multiple practical constraints in their design projects.

To address shortcomings related to faculty experience and competencies to cover all curricular areas, as required by the **Faculty** criterion, a few programs have added one or more faculty members. Typically they have taken advantage of existing open positions or of planned retirements to add these resources. To strengthen faculty guidance and oversight, some programs have decided to encourage faculty member participation in workshops related to ABET accreditation and assessment.

To meet the **Program Criteria** requirements for curricular breadth and depth, a number of programs have modified their courses and their curricula to insure that adequate coverage of topics such as verification, validation and maintenance. Some have developed completely new courses to address missing content or to provide additional depth in certain areas.

While few of the programs from which data were collected reported unexpected benefits, those who have made improvements uniformly reported that the improvements made were beneficial and should have been made, with or without an accreditation process. In several cases program leaders agreed that the results of the accreditation review gave them leverage with both members of their faculty and with their institutions' administration to make appropriate improvements. And, finally, all agreed that having ABET accreditation gives credibility to their programs by certifying that their software engineering program is a real engineering program.

FUTURE DIRECTIONS

At the time this chapter was being written, there were 35 undergraduate software engineering programs being offered by colleges and universities in the United States. Fifteen of them have been accredited by ABET. It appears likely that most of the remaining programs, which are not yet accredited, will be seeking initial accreditation within the next few years.

Finally, the National Academy of Engineering has made a recommendation that the master's degree should become the first professional degree accepted for entry into the engineering profession. Currently, ABET allows only one degree level at each institution in each field of engineering to be accredited. If the master's degree becomes the entry point into the engineering profession, that would imply a policy or practice change for ABET to allow accreditation at both the masters and bachelors level or to award accreditation primarily at the masters level. There are several good arguments for and against each of these proposals. Only time will tell if any change will be made and what form that change is likely to take.

REFERENCES

ABET. (2006). *Accreditation policy and procedure manual*. Baltimore, MD. ABET, Inc. Retrieved May 13, 2007, from http://abet.org/forms.shtml.

Computing Research News (2007), *2005-2006 Taulbe Survey*, May 2007.

Duggins, S. L., & Thomas, B. B. (2002). An historical investigation of graduate software engineering curricula. *Proceedings of the 15th*

Conference on Software Engineering Education and Training (CSEET'02), Los Alamitos, CA, IEEE Computer Society Press.

Engineering Accreditation Commission. (1999). *Criteria for accrediting engineering programs: Effective for evaluations during the 2000-2001 Accreditation Cycle.* Baltimore, MD. ABET, Inc.

Engineering Accreditation Commission. (2000). *Criteria for accrediting engineering programs: Effective for evaluations during the 2001-2002 Accreditation Cycle.* Baltimore, MD. ABET, Inc.

Engineering Accreditation Commission. (2001). *Criteria for accrediting engineering programs: Effective for evaluations during the 2002-2003 Accreditation Cycle.* Baltimore, MD. ABET, Inc.

Engineering Accreditation Commission. (2002). *Criteria for accrediting engineering programs: Effective for evaluations during the 2003-2004 Accreditation Cycle.* Baltimore, MD. ABET, Inc.

Engineering Accreditation Commission. (2007). *Criteria for accrediting engineering programs: Effective for evaluations during the 2007-2008 Accreditation Cycle.* Baltimore, MD. ABET, Inc. Retrieved May 13, 2007, from http://abet.org/forms.shtml.

Engineering Accreditation Commission. (2007a). *Engineering self-study questionnaire.* Baltimore, MD. ABET, Inc. Retrieved May 13, 2007, from http://abet.org/forms.shtml.

Guide to the Software Engineering Body of Knowledge (2004), Bourque, P. and Dupuis, R., (Eds.), Los Alamitos, CA, IEEE Computer Society Press.

Joint Task Force on Computing Curricula. (2004). *Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering.* IEEE Computer Society and Association for Computing Machinery.

Tomayko, J. E. (1998). Forging a discipline: An outline history of software engineering education. *Annals of Software Engineering 6(1998)*, 3-18.