

SOFTWARE ENGINEERING BACCALAUREATE PROGRAMS IN THE UNITED STATES: AN OVERVIEW

Donald J. Bagert¹ and Mark A. Ardis²

Abstract - There are currently over 20 Bachelor of Science in Software Engineering degree programs in the United States. The first accredited software engineering programs in the U.S. are likely in the 2002-03 cycle, and it is expected that the total number of such programs will continue to see steady growth for several years to come. The authors have provided a comparison of programs in order to determine what trends are emerging, which will benefit both current software engineering undergraduate programs, as well as those institutions which are thinking of creating new degrees of this type. The curriculum content of these programs are broken down by subject area and compared with curriculum models and accreditation criteria. The results of a survey of undergraduate software engineering programs worldwide that was conducted by the authors is used both to provide additional data about the U.S. programs, as well as to compare them as a group to their counterparts in other countries.

Index Terms – Accreditation guidelines, Curriculum issues, Software engineering, Undergraduate degree programs

INTRODUCTION

Although Master's degree programs in Software Engineering (SE) have existed in the United States since the late 1970's, and baccalaureate SE programs have existed internationally for about fifteen years, undergraduate programs of this type in the U.S. have been a more recent phenomenon [4]. The first school in the United States to offer a Bachelor of Science in Software Engineering (BSSE) was the Rochester Institute of Technology in 1996. The Working Group on Software Engineering Education and Training (WGSEET), who is maintaining a list of existing SE Bachelor's degree programs worldwide, has identified 21 such programs in the U.S. existing at the beginning of the 2002-03 academic year, with several more being proposed.

The number of SE undergraduate programs in the United States is now of a sufficient number to start examining them to see if any trends are emerging, as well as to see how they compare to their more established counterparts from other countries. The authors have analyzed these programs using two different sources:

- Details of the curricula for the U.S. programs, using information provided at the websites of the respective institutions, and

- A survey requesting additional information, created by the authors and sent to all of the programs on the WGSEET list.

The following sections provide:

- A general profile of the U.S. programs,
- A comparison of these programs with curriculum models and accreditation criteria,
- The results of the aforementioned survey of Bachelor's degree programs worldwide, and
- A summary of the trends that have emerged.

PROFILE OF SE DEGREE PROGRAMS

Of the 21 U.S. institutions in the WGSEET list of software engineering Bachelor's degree programs, the authors were able to find detailed information on 18 of those programs. Table I contains the list of those 18 schools along with the URL containing the software engineering curriculum description for that institution.

These schools represent a wide range of institutions. About one-third are public schools, two of them being flagship schools of their respective states. Five of them are members of the Association of Independent Technological Universities (AITU), an 18-member group which also includes MIT, Carnegie Mellon and Cal Tech among their number.

All but one of the programs studied have "Bachelor of Science in Software Engineering" as the name of its degree. Three of the 18 institutions do not offer a computer science degree, and one other is phasing it out for the SE program.

Academic Colleges and Departments

A breakdown of SE programs by school/college revealed that

- 11 were housed in the same school as the traditional engineering degrees (e.g. mechanical engineering);
- 2 institutions are not split into separate schools, but offered traditional engineering degrees;
- 3 programs were part of a separate school of computing or information systems, and
- 2 were part of a school of sciences.

So there is a strong tendency to align with traditional engineering programs, although a few institutions are seeing computing as a peer of engineering at the college level.

¹ Donald J. Bagert, Rose-Hulman Institute of Technology, Computer Science & Software Engineering, Terre Haute IN 47803 Don.Bagert@rose-hulman.edu

² Mark A. Ardis, Rose-Hulman Institute of Technology, Computer Science & Software Engineering, Terre Haute IN 47803 Mark.Ardis@rose-hulman.edu

TABLE I

INSTITUTIONS OFFERING BACHELOR'S PROGRAMS IN SOFTWARE ENGINEERING, WITH THEIR CURRICULUM PAGES	
School	URL
Auburn University	http://www.eng.auburn.edu/csse
Capitol College	http://www.capitol-college.edu/academics/ugrad/degrees/sebs.html
Champlain College	http://www.champlain.edu/majors/softwareeng
Clarkson University	http://www.clarkson.edu/software_engineering
Cogswell Polytechnical College	http://www.cogswell.edu/software.html
Drexel University	http://www.drexel.edu/ecm/ugsite/undergrad/ugacad.html
Embry-Riddle Aeronautical University	http://www.erau.edu/0Universe/01/01b-softwareengineering.html
Fairfield University	http://www.ffldusoe.edu/bsse.html
Milwaukee School of Engineering	http://www.msoe.edu/eecs/se/
Mississippi State University	http://www.cs.msstate.edu/~dampier/bsse/
Missouri Tech	http://www.motech.edu/Catalog/19.asp
Monmouth University	http://bluehawk.monmouth.edu/~se/
Montana Tech	http://www.mtech.edu/admission/programs.cfm?Program_ID='BSSE'
Penn State University - Erie	http://www.pserie.psu.edu/academic/engineering/degrees/ece/se-index.htm
Rochester Institute of Technology	http://www.se.rit.edu/degrees.php
Southern Polytechnic State University	http://cs.spsu.edu/csdept/Software%20Engineering%20Main%20Page.htm
University of Michigan-Dearborn	http://www.engin.umd.umich.edu/CIS
University of Wisconsin-Platteville	http://www.uwplatt.edu/~csse

Only one of the institutions studied had no traditional engineering programs.

Twelve of the 18 programs are housed in academic departments, with the rest being in college/schools that separately administer each program in lieu of departments. Of those 12 programs:

- 5 are in the same department as computer science,
- 3 are with electrical and computer engineering,
- 1 is jointly administered by electrical engineering and computer science, and
- 3 are in separate software engineering departments.

So there is currently no consensus as to which department should administer software engineering programs.

Curriculum Content

The courses in the various software engineering curricula can be broken down into the following areas: computer science, software engineering, mathematics, traditional engineering, and other courses.

As far as computer science courses are concerned, the schools showed a number of similarities to each other as well as to their respective computer science degrees. All of the programs include a sequence of 2 or 3 introductory computer science courses that teach programming skills, and a computer organization/assembly language course of some sort. More than half of the curricula also offered courses in operating systems, programming language concepts, database systems and networks.

The total number of software engineering courses varies across programs from 2 to 10, with an average of 6 per program. All of the programs require a one or two term capstone experience ("senior project") where students work in teams in order to develop a software project. More than half of the schools have as their first software engineering course(s) a one or two-term sequence that covers a broad

spectrum of software engineering topics. Almost all of the programs also have at least one software engineering course focusing on a specific topic. Most of the schools offered at least one course in software architecture or design, reflecting the design content traditionally seen in engineering curricula.

Less than half of the programs had courses devoted specifically to Project Management (7 programs), Quality Assurance (7 programs), or Software Requirements (6 programs). The other programs presumably cover these topics in overview software engineering courses. Courses in ethics, human-computer interface, and formal methods in software engineering are offered by about a third of the programs.

Each of the programs studied include the standard calculus sequence. All but two require probability and/or statistics, and all except one program require one discrete mathematics course. (The other program requires 2 discrete math courses.) About half of those discrete math courses are offered by the math department, while the others list it as a computer science course. About half of the programs require differential equations (7 require it), a similar number require linear algebra, and two give an option of one or the other. Thus, the mathematics requirements of software engineering programs are similar to that seen in computer science curricula, and are slightly different in content to traditional engineering program (although the total number of mathematics classes are about the same).

Most of the U.S. software engineering programs require little in the way of traditional engineering courses, with the main exception being digital logic, which is required in nine of the degree plans. Besides circuits and engineering economics (3 each), no other (non-SE) engineering course is required by more than one program.

All of the programs include general education content required by their particular institution, including basic

science and humanities. Many of the degree plans also required the student to take three or more courses in a particular application domain area; these courses are intended to provide a background in that domain sufficient to develop software applications in that area upon graduation. Some of the institutions require a particular application domain (e.g. Mississippi State requires one in computer security), while others (such as the Milwaukee School of Engineering) allow the student to choose among one of several possible domain areas.

SE ACCREDITATION CRITERIA

In the late 1990's, the Accreditation Board for Engineering and Technology (ABET), which accredits engineering, computing and technology programs in the United States, approved criteria for accrediting software engineering under the Engineering Accreditation Commission (EAC). The first undergraduate software engineering programs were considered in the 2002-03 accreditation cycle; at least four schools (all included in this study) have publicly stated that they were visited by ABET in the fall of 2002. The ABET/EAC criteria [1] contains eight general criteria, of which Criterion 4 (Professional Component) and Criterion 8 (Program Criteria) specific address requirements for specific curriculum content.

Criterion 4 states "Students must be prepared for engineering practice through the curriculum culminating in a major design experience based on the knowledge and skills acquired in earlier course work..." As stated in the previous section, all of the programs studied include such a project. This criterion goes on to state that "The professional component must include: (a) one year of a combination of college level mathematics and basic sciences (some with experimental experience) appropriate to the discipline; (b) one and one-half years of engineering topics, consisting of engineering sciences and engineering design appropriate to the student's field of study and (c) a general education component that complements the technical content of the curriculum and is consistent with the program and institution objectives." In general, the 18 programs included in this study are following these guidelines.

Criterion 8 specifies criteria for each individual discipline. The curriculum section of the software engineering criteria states that "The curriculum must provide both breadth and depth across the range of engineering and computer science topics implied by the title and objectives of the program. The program must demonstrate that graduates have: the ability to analyze, design, verify, validate, implement, apply, and maintain software systems; the ability to appropriately apply discrete mathematics, probability and statistics, and relevant topics in computer science and supporting disciplines to complex software systems; and the ability to work in one or more significant application domains." As stated in the previous section, the software engineering content of the various programs vary

widely, all provide a foundation in computer science, all require discrete mathematics, almost all require probability and statistics, and several programs require specific application domains.

SE CURRICULUM MODELS

In the late 1990's, WGSEET developed the *Guidelines for Software Engineering Education* [3], which subsequently became the *de facto* source for undergraduate software engineering curriculum models. All of the places where the software engineering programs were in general agreement (e.g. computer science introductory programming sequence, introductory SE course, capstone senior project, calculus, and discrete mathematics) are also contained in the *Guidelines*, while the programs vary in applying the remainder of the report's recommendations.

A current effort by the ACM/IEEE-CS joint Computing Curricula-Software Engineering (CCSE) project is addressed in the next section.

SURVEY RESULTS

We received 17 responses to a survey about BSSE programs: 9 from the U.S., 4 from Australia, 2 from Canada, 1 from Ireland and 1 from New Zealand. Table II summarizes our results.

We asked how many graduates each program had, or if no graduates yet, when they were expected. Most U.S. programs have not graduated any students yet, but all hope to do so by 2004. Non-U.S. programs have produced almost 500 graduates so far.

We next asked about graduation class size and number of faculty teaching core courses. U.S. programs seem on average to have a smaller number of students, though the maximum was the same for U.S. and non-U.S. (100). Our question about faculty size was not interpreted consistently by all respondents, but most U.S. programs have a relatively small number of faculty (mean of 6).

Almost all programs have a capstone project that spans one academic year. Clients for those projects may come from on-campus or off, with no clear preference. Only about half of the programs received contributions from their off-campus clients.

Finally, we asked about accreditation plans. All programs are accredited or expect to seek accreditation. In the U.S. all seek accreditation from ABET. All non-U.S. programs have at least provisional accreditation.

TABLE II
SURVEY RESPONSES

Question	U.S. Responses	Non-U.S. Responses
1. How many students have completed your program?	3 programs have a total of 20 graduates	6 out of 7 have a total of 476 graduates
2. If no students have completed, when do you expect the first students to finish?	4 in 2003 1 in 2004 1 in 2006	2003
3. How many students do you expect to graduate per year when you reach a steady state?	Low = 10 High = 100 Mean = 30	Low = 15 High = 100 Mean = 50
4. How many full-time equivalent faculty teach the core courses in your curriculum?	Low* = 3 High** = 15 Mean = 6	Not applicable -- many schools have faculty that teach across many disciplines
5. If your program has a capstone project experience, how long does it last?	whole academic year: 12, 2 quarters: 2, 1 semester: 1	whole academic year: 7
6. If your program has a capstone project experience, do the clients for those projects come from on-campus or off-campus?	off-campus: 2 both: 6	on-campus: 2 both: 6
7. If you have off-campus clients for projects, do they contribute any funds or equipment to your department or school?	yes: 4 no: 4	yes: 2 no: 5
8. Do you expect to seek accreditation for your program? If so, when and by what agency?	All programs interested in accreditation by ABET, 2 have already been visited, rest will seek by 2006	All have at least provisional accreditation

* Excludes one school that is just getting started

** May include most CS faculty

Another part of our survey asked about coverage of topics identified in the *Software Engineering Education Knowledge (SEEK)* [6], an activity of the ACM/IEEE-CS Computing Curricula-Software Engineering project. *SEEK* contains a collection of topics considered important in the education of software engineering students and was created and reviewed by volunteers in the software engineering education community. Topics in *SEEK* are divided by Knowledge Areas (KAs) and then further subdivided by Units. In this study only KAs and individual topics were studied. For example, REQ.ma.8 is a specific topic in the Modeling and Analysis (ma) unit of the Requirements knowledge area. Table III describes these KAs.

Each topic in *SEEK* is also categorized for its importance: Essential, Desired, or Optional. Essential topics are also annotated with indicators from Bloom's Taxonomy in the Cognitive Domain [5] to show the level of mastery expected. Only three of the six values from Bloom are used: knowledge, comprehension, and application.

We received 9 responses to our survey that included information about coverage of *SEEK* topics. As would be expected, most programs cover most, if not all, of the topics. We were interested in finding topics that were *not* covered in multiple programs. No topic was mentioned (that is, was listed as not covered) by more than 5 programs.

Table IV lists all of the *SEEK* topics that were mentioned by at least 3 programs. One interesting pattern to observe is the lack of coverage of Evolution (EVO) topics. Another pattern is the absence of any Essential topics whose level of mastery was greater than comprehension. That is, all application level Essential topics seem to be covered by most programs. Finally, none of the Management or Professional Practice topics were mentioned more than twice. That is, virtually all programs cover all topics in those areas.

TABLE III
SEEK KNOWLEDGE AREAS [FROM 6]

Knowledge Area	Description
Fundamentals (FND)	Theoretical and scientific underpinnings, mathematical foundations to model and facilitate reasoning, and first principles that produce predictable results.
Professional Practice (PRF)	Knowledge, skills, and attitudes to practice software engineering in a professional, responsible, and ethical manner; including technical communication, group dynamics, and professional responsibilities.
Requirements (REQ)	Analysis of feasibility, elicitation and analysis of stakeholders' needs, creation of a precise description of the system, and validation by the stakeholders.
Design (DES)	Issues, techniques, strategies, representations, and patterns used to determine how to implement a component or a system. Includes specification of internal interfaces, architectural design, data design, user interface design, design tools, and evaluation of design.
Construction (CON)	Development of software components, including translation of a design into an implementation language, development and execution of component tests, and development and use of program documentation.
Verification & Validation (VAV)	Static and dynamic system checking to ensure that the resulting system satisfies its specification and meets the expectations of the stakeholders.
Evolution (EVO)	Supporting the stakeholders' mission in the face of changing assumptions, problems, requirements, architectures and technologies; including activities before and after each release.
Process (PRO)	Commonly used software life-cycle process models; definition, implementation, measurement, management, change and improvement of software processes; use of a defined process to perform the technical and managerial activities needed for software development and maintenance.
Quality (QUA)	Quality of products and of processes used to develop them. Includes usability, reliability, safety, security, maintainability, flexibility, efficiency, performance and availability.
Management (MGT)	Planning, organization, and monitoring of all software life cycle phases.

TABLE IV
SEEK TOPICS NOT COVERED IN PROGRAMS AS REPORTED IN SURVEY

Topic	Description	Bloom Level	Importance	Programs Without
CON.fm.1	Application of abstract machines (e.g. SDL, Paisley, etc.)	k	E	3
CON.fm.6	Mapping of a specification to different implementations	k	E	4
CON.fm.7	Refinement	c	E	3
DES.hci.10	Psychology of HCI		D	3
DES.str.4	Aspect oriented design		O	4
EVO.ac.1	Working with legacy systems (e.g. use of wrappers, etc.)	k	E	3
EVO.ac.2	Program comprehension and reverse engineering	k	E	3
EVO.ac.3	System and process re-engineering (technical and business)	k	E	3
EVO.ac.6	Refactoring	k	E	3
EVO.ac.7	Program transformation		D	5
EVO.ac.8	Data reverse engineering		D	5
EVO.pro.4	Cost models of evolution		D	3
FND.ef.1	Empirical methods and experimental techniques (computer-related measuring techniques for CPU and memory usage)	c	E	3
FND.ef.6	Engineering science for other engineering disciplines (strength of materials, digital system principles, logic design, fundamentals of thermodynamics, etc.)		O	3
FND.md.4	Model checking and development tools	k	E	3
PRO.imp.7	ISO/IEEE Standard 12207: requirements of processes	k	E	3
QUA.pda.3	Quality product models	k	E	3
QUA.pro.4	Quality-related process areas of ISO 15504	k	E	3
QUA.pro.6	The Baldrige Award criteria for software engineering		O	5
QUA.pro.7	Quality aspects of other process models		O	3
QUA.std.4	IEEE software quality-related standards		D	3
REQ.ma.8	Modeling embedded systems (e.g. real-time schedulability analysis, external interface analysis, etc.)		D	3
VAV.hct.5	Web usability; testing techniques for web sites	c	E	3
VAV.hct.6	Formal experiments to test hypotheses about specific HCI controls		D	5
VAV.tst.12	Deployment process		D	4

COMPARISON WITH GRADUATE PROGRAMS

Master of Software Engineering (MSE) programs have been in existence longer than undergraduate programs. At last count there were about 25 of these programs in the U.S. Many of these programs resemble the model introduced by the Software Engineering Institute (SEI) in 1989 [2]. That model prescribed:

- 6 core courses in software engineering:
 - Specification of Software,
 - Software Verification and Validation
 - Software Generation and Maintenance
 - Principles and Applications of Software Design
 - Software Systems Engineering
 - Software Project Management
- 4 technical electives
- 2 semesters of project work

It is possible to complete this program in one year of full-time work, but many of the students who study in MSE programs do it part-time while they work in software jobs.

Some of the BSSE programs we studied resemble these MSE programs, but most do not. In particular, many undergraduate programs include an overview software engineering course, while very few of the MSE programs include one. This may reflect the difference in maturity of the students: MSE students are often already familiar with general software engineering concepts through their jobs, while undergraduates have not had that experience.

Another difference is the number of courses devoted to specific software engineering topics. For example, almost all MSE programs have a course on project management, but only 7 of the 18 U.S. BSSE programs we studied have such a course. Although MSE programs package their material differently from BSSE programs, both types of programs cover roughly the same material.

One area of commonality between BSSE and MSE programs is in project work. All programs, BSSE and MSE, have a capstone project, usually spanning one academic year. All such projects involve teams of students working for a client.

SUMMARY

In this paper we have surveyed existing BSSE programs in the U.S. We have compared them with their counterparts in other countries and with graduate level programs. We have examined the curriculum content and structure and reported the results of a survey of several programs.

BSSE programs are a new, but growing phenomenon in the U.S. Most seek accreditation by ABET and have paid careful attention to curriculum recommendations, such as the *Guidelines for Software Engineering Education* and *SEEK*. BSSE programs share much in common with undergraduate programs in computer science, including common foundation computer science and mathematics courses. Although BSSE programs cover the same material as MSE programs, it is packaged a little differently in courses. In particular, many BSSE programs contain an overview software engineering course, while MSE programs usually do not.

ACKNOWLEDGMENT

The authors appreciate the help of Joseph Clifton of the University of Wisconsin-Platteville and Stephen Frezza of Gannon University, who maintain the WGSEET list of software engineering undergraduate programs, and to the survey respondents.

REFERENCES

- [1] Accreditation Board for Engineering and Technology, Inc., *Criteria for Evaluating Engineering Programs*, 3 November 2001.
- [2] Ardis, Mark and Ford, Gary, *1989 SEI Report on Graduate Software Engineering Education*, Carnegie Mellon TR CMU/SEI-89-TR-21, June 1989.
- [3] Bagert, Donald; Hilburn, Thomas B.; Hislop, Greg; Lutz, Michael; McCracken, Michael and Mengel, Susan., *Guidelines for Software Engineering Education Version 1.0*, Technical Report CMU/SEI-99-TR-032, Software Engineering Institute, Carnegie Mellon University, Pittsburgh PA, October 1999.
- [4] Bagert, Donald, "Education and training in software engineering", *Encyclopedia of Software Engineering*, Second Edition, John Wiley and Sons, 2002, pp. 452-465.
- [5] Bloom, B. et al., *Taxonomy of Educational Objectives: Handbook I: Cognitive Domain*, David McKay, 1956.
- [6] *Second Draft of the Software Engineering Education Knowledge*, December 6, 2002. <sites.computer.org/ccse/know/SecondDraft.pdf>