Software Courses in Computer Engineering

Dr. Dick Blandford, University of Evansville

Dick Blandford is the department chair of the Electrical Engineering and Computer Science Department at the University of Evansville.

Dr. Christopher Miller, Rose-Hulman Institute of Technology

Chris is an Assistant Professor of Electrical & Computer Engineering at Rose-Hulman Institute of Technology. His interests include engineering education, embedded systems, and ubiquitous computing.

Software Courses in Computer Engineering

Abstract
There are approximately 250 ABET[1] accredited computer engineering programs in the U.S. Most of these were developed as a specialty out of existing electrical engineering programs. Some developed from computer science programs and some developed as a combination of computer science and electrical engineering in one department. As a result there is some significant variance between the amount of hardware and software that computer engineering programs require.

For this paper we examined forty randomly selected programs in computer engineering and attempted to determine the distribution of required software courses. We have categorized software-related courses and summarized the program requirements with respect to these categories. Every program has a published online curriculum as well as catalog descriptions of the courses. Most of our analysis comes from that data. In several cases where the software content of a course was not clear from a catalog, we contacted the department chair by email or phone.

We compared our results with the data from the version of the Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering published by the Joint Task Group on Computer Engineering Curricula Version 2015 October 3.

We provide a summary statement but we make no recommendations. We believe this information will be useful to anyone developing a new computer engineering program or to those who are in the process of curricular revision.

Introduction
The ABET website currently lists 250 accredited programs with "computer engineering" in their title. Most of these programs grew out of established electrical engineering programs but many came from elsewhere including a few which were established independent of an accredited electrical engineering program. Titles for these programs have some variation including: "computer engineering", "electrical and computer engineering", "computer science and engineering", "electrical engineering – computer engineering concentration", "computer engineering option in electrical engineering", "computer systems engineering", and some combination degrees that marry computer engineering to other engineering or computer science areas. These variations in program names are meant to imply a slightly different emphasis in course requirement content but all share the same ABET accreditation process for computer engineering.

By far the most common name for the degree is "computer engineering" (204 programs) followed by "electrical and computer engineering" (22 programs). For this paper we have limited ourselves to the examination of these 226 programs.

Computer engineering is by its nature a combination degree that straddles both electrical engineering and computer science. While a typical computer science program has some hardware content from computer organization and architecture classes, and the typical electrical

engineering program has some software content from microcontroller classes, neither has enough of the other to allow a graduate, for example, to design an engine controller for an automobile. Such a task requires a detailed knowledge of both computer science and electrical engineering. Most computer engineering undergraduates take courses that are exclusively hardware, exclusively software, and some combination of the two. Our interest in this paper is in the required courses which are exclusively software courses that are given by ABET accredited computer engineering programs in the United States.

Methodology

Almost all of our data for this study was gathered from data presented online by each program. Indeed, every program in computer engineering has online resources that includes a list of required courses, a paragraph or two describing each course, and a typical degree plan that presents the usual ordering of the courses in the student's plan of study. In most cases, it is evident as to which category a course falls into: hardware, software, hardware/software, or something else (systems, mathematics…). In those cases, where it was not clear which category the course fell into, we attempted to find the course syllabus or, in a few cases, we sent email to the department chair seeking further information.

As to the number of programs examined, we initially believed that a random sample of fifty programs would be adequate. After tallying the data for the first twenty programs a pattern emerged and this pattern changed little after forty programs so we stopped there.

The common software courses that we sought included the following:

1. Introductory course in programming. This is typically a first course in a high level language. We call this CS Fundamentals I.

2. Introductory course on programming and data structures. This course may introduce a second high level language and typically focuses on data structures and some algorithms. We call this CS Fundamentals II.

3. Object-oriented courses. About a third of the programs had a separate course in object-oriented programming. Many of those that did not have such a course had at least an introduction to OOP in the first two courses.

4. Algorithms. This course is sometimes taught as primarily a mathematics course and some programs teach it with very little programming. We nevertheless classified it as software because of its intimate connection to the software design process and because it is usually taught by computer science faculty. Roughly half of the programs had a separate course on algorithms.

5. Programming languages. This course does a comparative analysis of a variety of high-level programming languages with regard to programming constructs, memory requirements, operational semantics, etc. It typically has a strong software component. Only about one program in ten had a course on programming languages.

6. Computer organization and architecture. Strictly speaking, some may not call this a software course but it does typically include microprogramming, benchmarking, operating system mechanics, and assembly language structure. Since it is usually taught by computer science faculty and is a mainstay of most computer science programs, we called it a software class. Nearly every program had a required course on computer organization and/or architecture, and many had two. Those that did not have such a course covered the subject in other courses. A few programs had architecture courses that focused on microcontrollers rather than on computers in general.

7. Operating systems. This course often has the organization and architecture class as a prerequisite along with a course covering data structures. In most versions of this class there is a strong software component. Almost every program had a course on operating systems.

8. Real-time programming. This course has a variety of names including "embedded systems". It can be a mixed hardware/software course. We included it on the software side only if there was an indication that it was primarily software in the course description. In some cases, the material covered in these courses has overlap with computer organization and architecture.

9. Software engineering. Some programs require a course or two titled "software engineering". In some cases, the catalog description for such courses does not appear to be substantially different from courses called algorithms or data structures elsewhere. For this paper we took the term software engineering to refer to courses that deal with the modelling, design, analysis, or verification of a software system.

10. Computer system concepts. This course may have many different variations, but in general covers the design, organization and implementation of system software. In many cases it focuses on practical application. Topics may include operating system concepts including processes, threads, memory management, and inter-process communication, the use and control of various input/output (I/O) devices, memory organization, and concurrency management, command language interpreters (shells), file systems, and elementary network concepts.

We considered several other classes including networks but, for computer engineering students, this class is most often a mix of hardware and software.

The data we accumulated is presented in Table A-1 of the appendix. Figures 1 to 3 provide a graphical comparison of the categories and the computer engineering programs.

Results

Figure 1 shows the percentage of schools surveyed which require courses in the ten core programming categories described above. As can be seen, four categories stand out as being common core requirements: CS Fundamentals I and II; Organization and Architecture; and Operating Systems. In many cases, programming courses which are not required under the computer engineering curriculum may satisfy elective requirements.

Figure 2 shows the percent of credit hours devoted to software based courses. For example, 8 of the 40 schools sampled had programs consisting of 18% software. Most of the schools in our sample required 10% to 18% software-based courses in their computer engineering programs. Since the average computer engineering program requires about 128 credit hours this comes to 13 to 23 hours of software. To put this in perspective a minor typically requires about 20 hours of course work in a discipline and ABET requires about 32 hours of math and science.

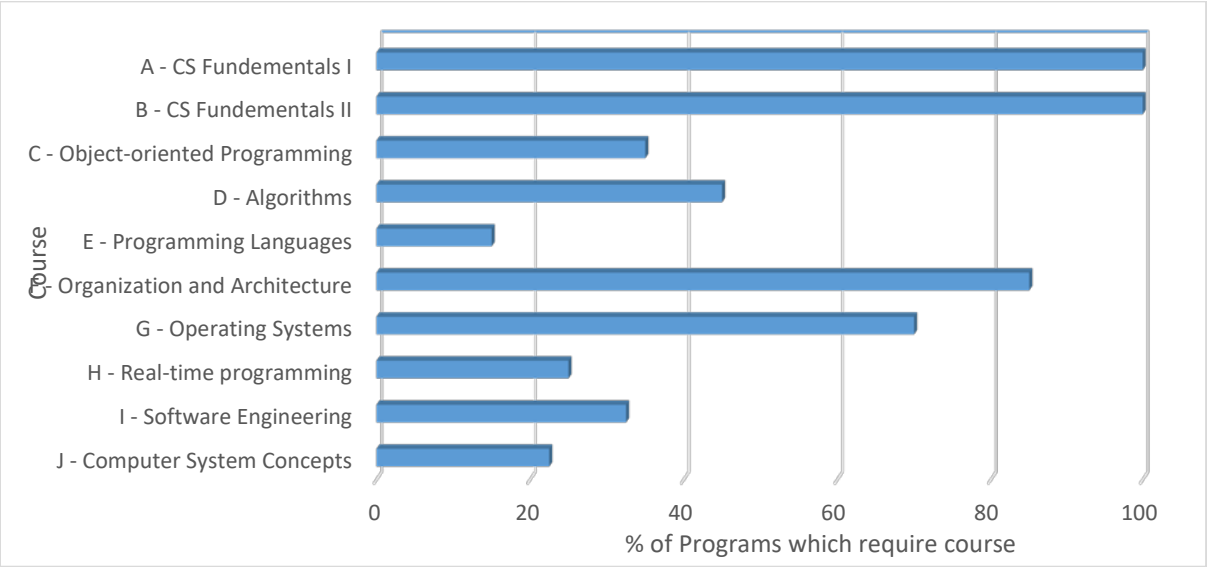

Figure 1 Percentage of programs surveyed which require a course within core categories.

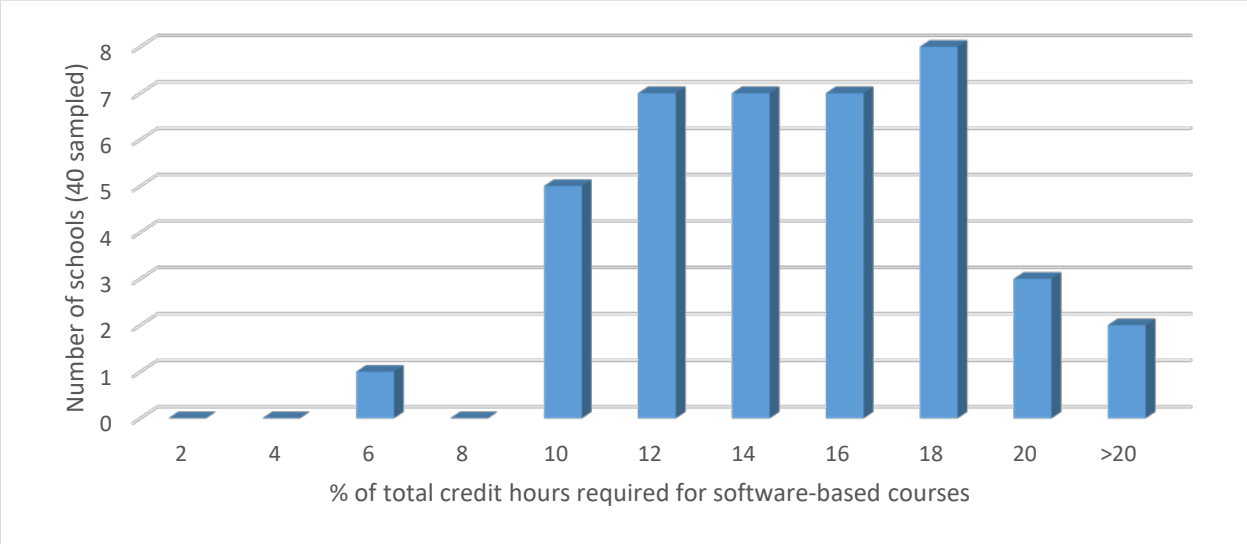

Figure 2 Number of schools out of forty in the sample with percentage of required software courses.

Figure 3 shows the percentage of the total credit hours required for each software course over all of the schools which require that course. Consider, for example, the Organization and

Architecture category in Figure 3. This varies from about 2.1% to 7.6% meaning that for schools which require Organization and Architecture, this course occupies from 2.1% to 7.6% of the total hours.

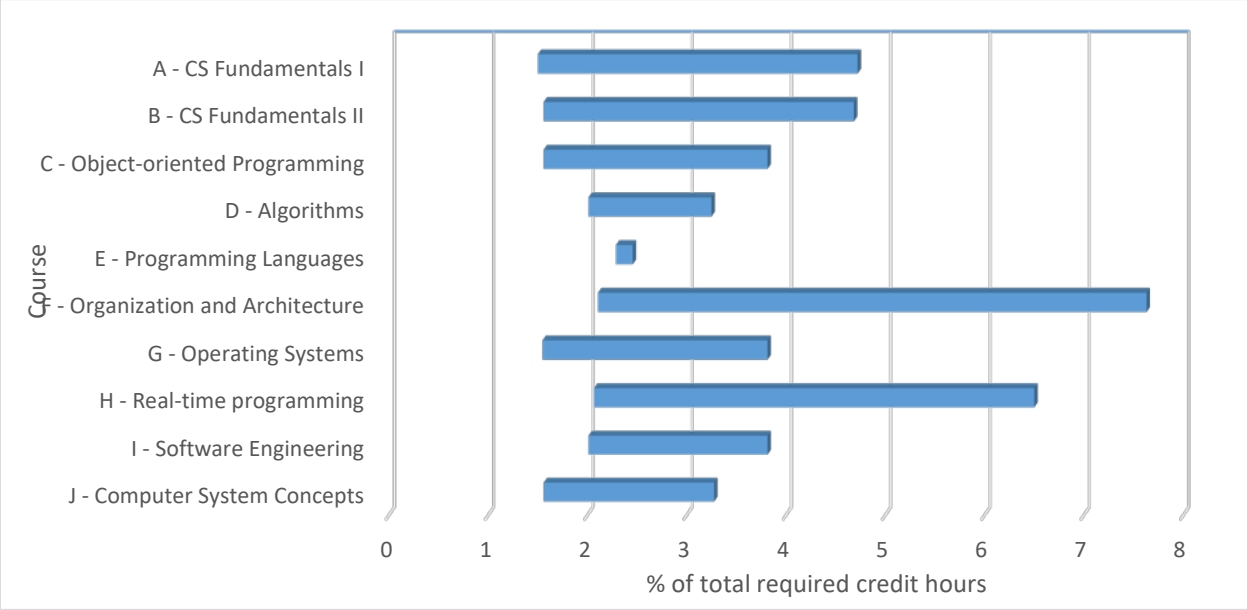

Figure 3The percent of total credit hours for each course over all schools which require that course.

Figures 2 and 3 exhibit the variance in balance between software courses and hardware courses in computer engineering programs. Even within the software course requirements, there is significant variance in the focus of these courses, with some programs placing significant weight in architecture and embedded applications. This could be attributed to the different paths in which these programs arose, as discussed in the introduction.

In terms of total credit hours required for a bachelor's degree in computer engineering, the average number over the forty schools in our sample was 128 semester hours. The minimum number was at 120 which is likely due to a state mandate. The maximum number is a little uncertain. One school reported 149 required credits for a degree, which included credits for cooperative education experience. Outside of this anomaly, the maximum was at 136 credit hours. The uncertainty in this number is due to what courses are counted as required credit hours. In at least one program, for example, there is a graduation requirement for 6 hours of college level foreign language but this is not counted in the credit hours required for the degree.

Sample Curricula from Computer Engineering 2016 Report

A report titled "Computer Engineering Curricula 2016" was issued by the Joint Task Group on Computer Engineering Curricula from the Association for Computing Machinery and the IEEE Computer Society[2]. The report contains six sample computer engineering curricula. Two of these were relevant to this paper: Curricula A for computer engineering programs developed from an electrical and computer engineering department and Curricula C developed from a joint electrical engineering and computer science department.

Curricula A has required software courses shown in Table 1. The required software courses for Curricula C is shown in Table 2.

Table 1
Curricula A required software courses from "Computer Engineering Curricula 2016"

Software category	Hours	Notes
A - CS Fundamentals I	4	
B - CS Fundamentals II	4	This program has no separate OOP course but OOP is covered in these two courses along with data structures and the analysis of algorithms.
C - Object-oriented Programming	3	
D – Algorithms	3	
E - Programming Languages	0	
F - Organization and Architecture	3	
G - Operating Systems	3	
H - Real-time programming	3	
I - Software Engineering	0	
J - Computer System Concepts	3	This program has a course titled "Client-Server Programming" that includes multi-threading and distributed computing that very roughly falls into this category.
Computer Security	3	

Table 2
Curricula C required software courses from "Computer Engineering Curricula 2016"

Software category	Hours	Notes
A - CS Fundamentals I	3	This program has no separate OOP course but OOP is covered in these two courses along with data structures and the analysis of algorithms.
B - CS Fundamentals II	3	
C - Object-oriented Programming	0	
D - Algorithms	4	
E - Programming Languages	0	
F - Organization and Architecture	3	
G - Operating Systems	3	
H - Real-time programming	-	There is no separate course in this category but it is contained in parts of several other courses.
I - Software Engineering	3	This is called Computer Systems Engineering but it includes many software engineering concepts.
J - Computer System Concepts	-	There is no separate course in this category but it is in parts of several other courses.
Network Security	3	

Concluding comments

1. The number of hours required for a computer engineering degree varied from a low of 120 to a high of 136 with the average being 128.
2. Every computer engineering program had at least two introductory software courses which we called CS Fundamentals I and CS Fundamentals II. Most often this was a six credit-hour sequence that did an introduction to programming, covered data structures, and did at least an introduction to object oriented programming. Some schools had a third course titled object oriented programming but this was not the norm.
3. Nearly every computer engineering program had at least one course titled "computer organization" or sometimes "computer organization and architecture". In some cases there was a computer organization course which was hardware oriented followed by a computer architecture class.
4. A full course in operating systems was also included in most computer engineering programs. A typical operating system class covered operating system components, resource management, interrupt handling, and concurrent processing.
5. About half of the computer engineering programs which we surveyed had a separate class in algorithms. In some cases algorithms coverage was built into CS2 or was covered in something other than a separate class. In several examples algorithms was combined with data structures in a separate course which followed the CS 1 and CS 2 sequence. In some cases, the separate algorithms class had only a small programming component and a heavy mathematics component.
6. Many programs had a class on embedded systems (sometimes called real time systems) which appeared to be a course which covered both hardware and software. In a few cases there was evidence that such classes were software-oriented and focused on real-time operating systems.
7. A full course on network and/or computer security was notably missing from many programs but this appears to be changing. Note that both of the sample curricula from the computer engineering study have at least one course on security.
8. Very few programs has a full course on parallel or multicore processing although this was evident in the electives and in the content of other courses.
9. Only two programs of the forty that we looked at had a course on programming languages. One program included a required course on numerical methods and another program required a course on the theory of computing.
10. On average about 15% of computer engineering required courses are software based. For a program with 128 total hours this comes to a little more than 19 credit hours. Eighteen to twenty-one credit hours typically qualifies as a minor in many programs.

Future work

The amount of required software does not tell the whole story for computer engineering. A computer engineering degree is *not* an electrical engineering degree with a minor in computer science. There are differences in the hardware courses and the courses that mix hardware and software between electrical and computer engineering. Likewise, many programs teach hardware and software to computer engineering students by way of laboratory projects and experiments. We have made no attempt to distinguish computer engineering courses in this area and we leave this to future studies.

References

1. http://www.abet.org/ABET
2. "Computer Engineering Curricula 2016" was issued by the Joint Task Group on Computer Engineering Curricula from the Association for Computing Machinery and the IEEE Computer Society. It is available at https://www.acm.org/binaries/content/assets/education/ce2016-final-report.pdf

Appendix A

Table A-1. Summary of credit hours required per school per category.

School	Total hrs	A	B	C	D	E	F	G	H	I	J
Boston University	132	4	4		4		4				
Clarkson University	121	2	3	3			3	3		β	
Lipscomb University	132	3	3	3			3	3			
Seattle Pacific	131	5	5	5			10	5		5	
SUNY Binghamton	127	3	3				3	3			
U of Alabama Huntsville	129	3	3		3		3	3		3	
U of Colorado	128	3	6	3	3		3	3			
U of Evansville	131	3	3		3	3	3	3	3		
U of Idaho	128	3	3			3	3	3			3
U of Wisconsin	124	3	3								
Wright State	124	3	3		3		3	3			
Cal State Poly Pomona α	129	3	3	3			3	2*			3
Cal State Chico	120	3	3	3						3	
Cal State San Bernardino	132	4	4		4		5	4			
Drexel α	128	2	2	2			2.7				2
Florida State University	128	4	3	3				3			
IUPU Ft. Wayne	120	4	3					γ	3		
The University of Kansas	129	4	4			3	3	3		3	
University of South Alabama	130	3	3				3	3			
University of Texas at Austin	125	3	3	3	3		3				
Colorado Technical University α	127	2.67	5.33	2.67			5.33	2.67		2.67	
Grand Valley State University	149	4	4		3			4		3	
Rose-Hulman Institute of Technology α	129	2.67	2.67	2.67			5.33	2.67δ	2.67		
University of Bridgeport	132	4	4				3	3	3	3	
University of Maryland College Park	122	4	4	*	3	3		4			4
University of New Haven	127	6	3				3	3	3		
University of Scranton	129	3	3		3	3	3	3		3	
University of Texas at Arlington	121	3	3	3	3		6	3		3	
University of Turabo	128	6	3	3	3	3	3	3	3	3	
Rochester Institute of Technology	129	4	4		3		6		3		
Clemson University	127	3	3				3	3			3
University of Colorado at Boulder	128	4	4				3				3
Minnesota State University, Mankato	128	3	4				3	3	6		
Kansas State University	129	4	3	3	3		3	3	3		
Bucknell University ε	136	4	4		4		8	4			
Lehigh University	133	2	3				3	3		3	4
University of Texas Rio Grande Valley	126	4	3		3		3	3		3	3
The University of Alabama	123	4	4		4		3		8		
Gonzaga University	136	3	3		3		3				
West Virginia University	130	4	4				3	3		4	3
Count	128.425	40	40	14	18	6	34	28	10	13	9

α - semester equivalent for credit hours required on quarters system
β - some of these courses are titled software engineering
γ - partially covered in real-time programming course
δ - at least one of Data Structures & Algorithms or Operating Systems
ε - 4 credit hour per course equivalent