Strengthening the "Engineering" in Software Engineering Education: A Software Engineering Bachelor of Engineering Program for the 21st Century

Linda Laird
Stevens Institute of Technology
Hoboken 07030, USA
linda.laird@stevens.edu

Abstract— **In the fall of 2015, Stevens Institute of Technology welcomed the first freshmen into a newly launched Software Engineering Undergraduate Program based largely on the most recent ACM and IEEE-CS guidelines for undergraduate software engineering programs [1]. This is the first such program in the US that also has an ABET accredited general engineering curriculum. Students will receive a B.E. in Software Engineering Degree, and be prepared to sit for the Fundamentals of Engineering (FE) examination [2]. In addition to its strong engineering foundation, the program benefits from the success of the Stevens graduate program in software engineering.**

This paper describes the program: the motivation, the curriculum, the program assessment plan, the early challenges, and the results to date, including some consideration as a potential model of an undergraduate Cyber-Physical Systems Engineering curriculum.

Keywords—software engineering education, software engineering curriculum, cyber-physical systems education

I. Introduction

On any given day, there are over 150,000 job openings for software engineers in the United States [3]. This represents a huge variety of jobs, ranging from Python programmers or network administrators to overall systems architects. Frequently jobs may be called software engineering but in reality, only require basic software programming skills. Beyond this misnomer, there is a tendency in Silicon Valley to call the software development organizations "Engineering." This situation is reflected in the comment in an opinion piece in the Atlantic [4],

'"Engineer' is an aspirational title in software development. Traditional engineers are regulated, certified, and subject to apprenticeship and continuing education. Engineering claims an explicit responsibility to public safety and reliability, even if it doesn't always deliver."

Stevens has offered an M.S. in Software Engineering since 2001; it has is based on the IEEE/ACM SWE curriculum guidelines for graduate programs in software engineering [5], similarly to many other schools. The Stevens M.S. covers the full development life cycle, is quantitative and empirical, and emphasizes the engineering and development of trusted systems. In this context, the "engineering" in software engineering means using good engineering judgment to select the appropriate architecture and designs, tools and techniques, and applying them to reliably build trustable software.

Stevens is now offering a B.E. in Software Engineering. This new undergraduate program goes beyond the M.S. program in adding the foundation of traditional engineering to the software engineer. It strengthens the *Engineer* in Software Engineer.[1] We believe this is a unique program: our graduates will be qualified to sit for the F.E. exam and eventually the P.E. exam; it is ABET accredited, it meets the recent ACM/IEEE SWE undergraduate guidelines, and even includes some systems engineering. These graduates will be able to effectively build a variety of challenging systems, utilizing their broad engineering and their deep software engineering skills, including advanced and complex cyber-physical systems.

In ICSE 2016, one of the keynotes speeches is "Progress Toward an Engineering Discipline of Software" by Prof. M. Shaw [6]. Our intention is that the Stevens' undergraduate B.E. in Software Engineering is one more step in this path of progress.

The SWE curriculum is a standard Stevens Engineering Undergraduate Curriculum, consisting of 141 total credit hours. The SWE program-specific content consist of 9 required courses (27 credit hours), 2 courses for Senior Design (6 credit hours), and 2 courses for Domain Electives (6 credit hours).

The results to date are very encouraging; the fall 2015 enrollment is 18 freshman, 5 sophomores, and one junior. We expect a small number of graduates in 2018, with the number growing to 20 or 25 per year by 2022. Our goal is to remain a small, high-touch program.

II. Motivation for Program

Stevens' Mission is to *"inspire, nurture and educate leaders in tomorrow's technology-centric environment while contributing to the solution of the most challenging problems of our time."* Modern society depends upon systems of increasing complexity to sustain our quality of life, and the engineered systems being conceived and developed today have an increasing and significant percentage of their functionality allocated to software. This requires our future engineers to be strongly rooted in the fundamentals of

[1] This statement does not imply that all software engineers should have an engineering background. There are many pathways to

2377-570X/16 $31.00 © 2016 IEEE
DOI 10.1109/CSEET.2016.13

engineering and science, while also being equipped with a strong capability to develop and integrate software as a central feature in these systems. The software aspect of engineered systems today is not just another component of the system. It also provides the overarching integration framework, allowing systems to be both rich in functionality and capabilities, while being adaptive to context and control. This program also includes a thread of systems thinking, with electives that allow students to explore a domain of interest – domains such as healthcare, embedded systems, financial systems, naval engineering and architecting, and control systems.

III. Curriculum:

The Stevens Undergraduate Engineering Curriculum is a rigorous, hands-on curriculum, consisting of 141 total credit hours. The SWE program-specific content consists of 9 required courses (27 credit hours), 2 courses for Senior Design (6 credit hours), and 2 courses for Domain Electives (6 credit hours). Of the 9 required courses, 5 are significantly new courses and 4 are adapted from the Software Engineering Master's Program.

Two distinctive aspects of the Stevens engineering curriculum are the traditional breadth of engineering education (see curriculum below) and the integrative, eight-course Design Spine The Design Spine is a fundamental component of the engineering curriculum that is required for all B.E. students, regardless of discipline. It consists of eight core design courses taken throughout all eight undergraduate semesters of study including a two-semester capstone senior design project, which introduces students to the underlying principles of engineering design through hands-on and project-based learning. For the software engineering students, the senior design project will be a multi-disciplinary project focused on the students' domain of interest. The Design Spine causes broad exposure to the many fields of engineering with hands-on problem solving experience, which in itself is critical in preparing the students to succeed in the workplace. Students learn and practice interdisciplinary skills through their projects.

The 9 required SWE courses are shown in the table below.

Course Number	Course Name	Credits
SSW 215 (New)	Individual Software Engineering	3
SSW 315 (New)	Object-oriented software development	3
SSW 322 (New)	Software Design and Evolution	3
SSW 345 (New)	Model-Based Software Engineering	3
SSW 533 (Adapted)	Software Estimation and Measurement	3
SSW 555 (Adapted)	Agile Methods for Software Development	3
SSW 564 (Adapted)	Software Requirements Engineering	3
SSW 567 (Adapted)	Software Testing and Quality Assurance	3
SYS 481 (New)	Systems Engineering and Architecture	3

These required courses cover all the phases of software development and maintenance. The first three courses teach fundamental skills of computing, while later courses cover specific lifecycle activities, such as requirements, architecture and testing. All the courses emphasize teamwork and the role of software in larger systems.

The curriculum follows the standard Stevens Engineering curriculum as shown below:

TERM I
CH115 - General Chemistry I
CH117 - General Chemistry Lab I
MA121 - Calculus IA: Differential Calculus
MA122 - Calculus IB: Integral Calculus
E101 - Engineering Experience I
E120 - Engineering Graphics
E121 - Engineering Design I
E115 - Introduction to Programming
CAL 103 or 105

TERM II
MA123 - Series, Vectors, Functions, and Surfaces
MA124 - Calculus of Two Variables
PEP111 - Mechanics
MGT103 – Introduction to Entrepreneurial Thinking
E122 - Engineering Design I
CAL 105 or 103
Science Elective

TERM III
MA221 - Differential Equations
PEP112 - Electricity and Magnetism
E126 - Mechanics of Solids
E245 – Circuits and Systems
E231 – Engineering Design III
Humanities

TERM IV
MA134 - Discrete Math
E232 - Engineering Design IV
E234 – Thermodynamics
SSW215 - Individual Software Engineering
Science Elective
Humanities

TERM IV
MA134 - Discrete Math
E232 - Engineering Design IV
E234 – Thermodynamics
SSW215 - Individual Software Engineering
Science Elective
Humanities

TERM V

CE342 - Transport/Fluid Dynamics
E344 - Materials Processing
E321 - Engineering Design V
E243: Probability & Statistics
SSW: 315 Object-Oriented Software Development
Humanities

TERM VI

SSW 345: Model-based Software Engineering
E355: Engineering Economics
SSW 322: Software Design Evolution
SSW 564: Software Requirements Engineering
Domain Elective
General Elective

TERM VII

SSW 555: Agile Methods
SSW 567: Software Testing and QA
Domain Elective
General Elective
SSW 423 - Engineering Design VII
TG403: Senior Innovation I

TERM VIII

SSW 533: Software Estimation and Measurement
SYS 481: Systems Engineering and Architecture
General Elective:
SSW 424: Engineering Design VIII
Humanities:
TG404: Senior Innovation II

IV. Program Outcomes and Assessment:

Graduates of this program will:

- Employ sound principles and practices to design and implement software for complex engineered systems
- Assume a variety of roles on multidisciplinary engineering teams
- Communicate effectively with stakeholders in oral, written, and newly developing modes and media
- Demonstrate professionalism, including continued learning and professional activities
- Contribute to society by behaving ethically and responsibly

The achievement of these outcomes and the interpretation of results will be assessed using the standard Stevens processes for engineering programs, including the student grades and their accomplishments in Senior Design. Yearly program reviews, for at least the first five years, will be held to review the achievement of the outcomes and to determine improvements for the following year. These reviews will be based on student surveys and faculty assessments of learning outcomes.

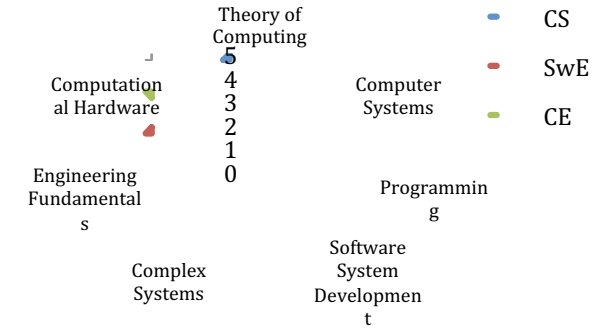
V. Early Challenges

Starting new degree programs can be challenging. Our biggest challenges to date have been:

- Differentiating SWE from other computing programs at Stevens
- Explaining Software Engineering to Students and Parents
- Start-up scheduling of courses, especially with Student Co-ops

Differentiating this computing program from the other computing programs at Stevens, both internally and for students and parents, was challenging. After many false starts, including trying to use 2006 IEEE/ACM diagram [8] made for this purpose, we created the model below, primarily for internal use.

Software Engineering, Computer Science, and Computer Engineering are all distinct disciplines, distinct programs, with distinct curriculums. This model compares and contrasts these programs based upon the coverage of six knowledge areas; 0 indicates no coverage and 5 indicates maximum coverage. It demonstrates that there are both intersections and unique aspects of each program.



For parents and students, the concepts of "Software is now the Building Material of Choice" and "Software Engineers: Builders of the 21st Century" seemed to work well to differentiate the program from Computer Engineering and Computer Science. Also, given Stevens' strong engineering culture, "Software Engineer" may just make sense to our prospective students.

Co-op is highly recommended for the students; Stevens has a superb career center and co-op program. Nevertheless, scheduling courses to support co-op is difficult during the startup, due to fewer students. These problems are overcome by careful scheduling and by working extensively with the individual students on their study plans.

VI. Pedagogy

The teaching style varies somewhat by professor, but there are many common attributes. The first common attribute is small classes. This program is designed to be a small, high-touch program with significant faculty interaction. The typical class size is expected to be approximately 20 students. The second is the use of collaborative learning [8] in many different formats. All of the courses use team projects; programming is taught as either pair programming or side-by-side programming. In the fall of 2015, Stevens built a Software Engineering Studio to 1) support cyber-physical systems

projects, 2) support a Problem-Based Learning (PBL) approach [9], and 3) encourage a sense of belonging, community, and foster relationships among the Stevens SwE students[10]. Professors are also experimenting with Engage Engineering techniques, such as using LEGOs in teaching software engineering estimation [11].

VII. Cyber-physical Systems (CPS) and Stevens' B.E. of Software Engineering

James Sturges, co-chairman of the National Academies Committee on 21st Century Cyber-Physical Systems Education (funded by NSF), observed that the SWE B.E. could be considered a model for an undergraduate CPS Engineering Curriculum [13][2]. NSF has a significant program on Cyber-Physical Systems (CPS) that they describe as: *"engineered systems that are built from, and depend upon, the seamless integration of computational algorithms and physical components. Advances in CPS will enable capability, adaptability, scalability, resiliency, safety, security, and usability that will far exceed the simple embedded systems of today. CPS technology will transform the way people interact with engineered systems …. New smart CPS will drive innovation and competition in sectors such as agriculture, energy, transportation, building design, and automation, healthcare, and manufacturing[14]*."

Based on preliminary work at Stevens, our faculty postulates the following required core competencies for a CPS engineer: Software Engineering, Computer Engineering, Interdisciplinary Engineering, and some Systems and Mechanical Engineering, consistent with the National Academies Interim Report [15]. The Stevens' SWE B.E. students will have a reasonable foundation in all of these areas, depth in Software Engineering, and significant hands-on experiences through both the Design Spine and co-op.

VIII. Summary:

The B.E. of Software Engineering Program is well underway at Stevens. It is a unique program in many respects, with its broad engineering base and depth in software engineering. It unequivocally strengthens the *Engineer* in Software Engineer. It may also be an early version of a CPS Engineering undergraduate program.

As the program progresses, we will continue to assess the outcomes and report any significant results, including career path data.

IX. Acknowledgements:

This visionary for this program is the provost of Stevens, George Korfiatis, who asked for a software engineering program embedded into the undergraduate engineering program.

Our curriculum expert is Mark Ardis, a self-described curriculum geek, who has contributed so much in so many ways to Software Engineering curricula worldwide. He is the true designer of this curriculum.

[2] Mr. Sturges also suggested that with the 141 required hours, the program might be considered a masters program.

X. References

[1] Ardis, M., Budgen, D., Hislop, G., Offutt, J., Sebern, M., Visser, W., "Software Engineering 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering," joint effort of the ACM and the IEEE-Computer Society, 2014, http://securriculum.org.

[2] http://ncees.org/exams/fe-exam/, accessed 12/6/2015

[3] www.engineeringjobs.com, accessed 12/1/2015

[4] Bogost, I., The Atlantic, Nov. 5, 2015, "Programmers: Stop Calling Yourselves Engineers"

[5] Pyster, A., "Software Engineering 2009 (GSwE2009): Curriculum Guidelines for Graduate Degree Programs in Software Engineering," Integrated Software & Systems Engineering Curriculum Project Stevens Institute (2009)

[6] http://2016.icse.cs.txstate.edu/keynotes, accessed 12/6/2105

[7] Gallois, B., Sheppard, K., "The Design Spine: Revision Of The Engineering Curriculum To Include A Design Experience Each Semester." (2009) Paper presented at 1999 Annual Conference, Charlotte, North Carolina. https://peer.asee.org/755

[8] Brown, H., & Ciuffetelli, D.C. (Eds.). Foundational methods: Understanding teaching and learning, p. 508. Toronto: Pearson Education. 2009

[9] Hmelo-Silver, C., Duncan, R., & Clark, A., "Scaffolding and Achievement in Problem-Based and Inquiry Learning: A Response to Kischner, Sweller, and Clark (2006)". Educational Pyschologist , *42* (2). 2007

[10] Bull, C., & Whittle, J., "Observations of a Software Engineering Studio: Reflecting with the Studio Framework." CSEE&T 2014 (pp. 74-83). Klagenfurt: IEEE. 2014

[11] Laird, L., Yang, Y., "Engaging Software Engineering Education using LEGOs: A Case Study," accepted IEEE Conference on Software Engineering (ICSE-C 2016),

[12] http://www.acm.org/education/curric_vols/CC2005-March06Final.pdf

[13] Sturgis, J., private conversation, November 2015

[14] http://www.nsf.gov/pubs/2014/nsf14542/nsf14542.htm

[15] Interim Report on 21st Century Cyber-Physical Systems Education, National Academies Press, 2014