Developing an Undergraduate Software Engineering Degree

J. Fernando Naveda
(Moderator)
Rochester Institute of Technology
jfnics@rit.edu

Donald J. Bagert
Rose Hulman Institute of Technology
Don.Bagert@rose-hulman.edu

Steve Seidman
New Jersey Institute of Technology
Stephen.Seidman@njit.edu

Jocelyn Armarego
Murdoch University, Australia
Jocelyn@eng.murdoch.edu.au

Thomas B. Hilburn
Embry-Riddle Aeronautical University
hilburn@db.erau.edu

Susan Eisenbach
Imperial College London
S.Eisenbach@ic.ac.uk

Keywords: undergraduate software engineering, undergraduate degree, software engineering curriculum, software engineering faculty, accreditation

Panel Summary

As those who have done it can attest, developing an undergraduate degree in software engineering is a daunting and challenging task, and there have been instances where a department has tried, but failed to get its program approved. A strong desire to develop a program in software engineering together with interested faculty may not be enough to build a credible degree, let alone a curriculum that will be approved by all the administrative and State organizations who may have a say in it .This panel brings together a group whose experience in developing software engineering degrees at their respective institutions may be helpful to those thinking about doing so. Each member of the group will describe his/her experiences in developing an undergraduate program in software engineering and address key issues and problems that should be considered in any such effort. There will also be ample opportunity for interaction among the participants.

The following are some of the issues that will be discussed:

- Building a credible program. The difference between paper programs and real programs
- Developing a curriculum. What should the curriculum look like? What sort of support and guidance is available?
- Finding and developing qualified faculty
- Dealing with the politics: Where will the program live? Is the engineering of software real engineering? Is software engineering applied computer science?
- Accreditation issues
- Landmines
- Tactical issues: How can an SE program be built from existing programs of study?

Since it has become clear that software engineering plays a central role in computing, an increasing number of institutions of higher learning are turning their attention to the development of baccalaureate degree programs in software engineering. This trend has been

encouraged by the creation of numerous software engineering degrees at masters
level, and by the publication of an influential document on undergraduate software engineering education[1] in 1990.

In 1993, the Rochester Institute of Technology (RIT) allowed a core number of its faculty to develop an undergraduate degree in software engineering [2], and in 1996 admitted a group of students to what would become the first baccalaureate degree in software engineering in the United States (interestingly, various Australian and British universities had already been offering such degrees). RIT would not remain alone for long in the US. In 1998 the Milwaukee School of Engineering started its own undergraduate program in software engineering, admitting sophomores from its Computer Engineering program. In 2000, Monmouth University opened its doors to its new program in software engineering. Also in 2000, Georgia's Southern Polytechnic State University's School of Computing and Software Engineering obtained State approval to develop an undergraduate software engineering degree.

These are just a few examples of schools that have successfully implemented software engineering degrees in the US. Many others are sure to follow and it may be just a matter of time before software engineering becomes as popular as computer science. Some actually believe that software engineering could replace computer science as the program of choice for software professionals.

Jocelyn Armarego

Advocates of Software Engineering programs (at either undergraduate or graduate level) agree on one principle at least – SE is different from Computer Science, Computer Engineering and Information Systems, to name a few IT-based disciplines. It is these differences, in goals, focus and approach as well as content that make the development of SE programs necessary. However, that these differences are often not explicitly described makes SE program development problematic.

Developers of SE programs juggle these forces. For example, aligning SE programs within Engineering provides some benefits: the core of engineering practice and professionalism (accreditation, societal considerations, economics, etc) applies to SE as well as more traditional engineering programs. However, consensus on the place of advanced mathematics or the natural sciences may not be so easily achieved (but may be mandated). Hence common goals are in tension with different content needs. Approach and content are also in tension: while there is agreement that CS fundamentals provide some core SE knowledge and skills, software engineering principles and practices should underlie this learning (which implies faculty exists to take this approach). Yet organizational (i.e., university) policy may dictate that content determines who teaches what (e.g., that Computer Science must teach CS fundamentals (probably implying a CS approach).

While the result of this juggling is context-dependent (so what works in Australia may not be viable in other parts, and what are important issues for one group have no bearing on another), the insights we each bring will hopefully help achieve a global perspective on SE program development.

Don Bagert

The development of a proposal for a Bachelor of Science in Software Engineering (BSSE) at Rose-Hulman has been made considerably easier by the fact that there has been general support for the concept of the BSSE by faculty both inside and outside the Department of Computer Science and Software Engineering. However, some of the details were more difficult in gaining consensus and in proposing an actual curriculum model and its implementation. Among those issues were: 1) how the new program would impact the CS curriculum, 2) properly distinguishing the differences between the CS and SE curricula, 3) estimating the impact of the new degree on other departments and 4) dealing with issues involving the culture at Rose-Hulman e.g. the question of whether or not to offer differential equations in an institution where every single existing major did, and desire for the freshman year to be much the same throughout the disciplines.

Tom Hilburn

In the fall of 2002, we formally began offering a B.S. in Software Engineering (BSSE) at Embry-Riddle Aeronautical University. However, the genesis of the degree started much earlier. In early 1990s, we began to make changes in the content and emphasis of our undergraduate computer science program. There were a number of factors that encouraged this path: several of our faculty had software engineering background and experience; the ACM/IEEE-CS *Curriculum 1991* included SE modules and a sample SE curriculum; the employers of our graduates (Boeing, Lockheed-Martin, Raytheon, Rockwell-Collins, Sikorsky, etc.) voiced their need for software developers that were educated as "engineers"; and the university's mission and focus on professional aviation/aerospace programs supported the BSSE. Our approach to developing a BSSE has been evolutionary. We have, through the years, introduced software engineering topics and activities into existing courses, developed new SE courses, encouraged and supported the professional development of our faculty in SE areas, and finally changed the name of our program from computer science to software engineering. Although we face the same problems as others (hiring qualified SE faculty, the lack of exiting SE course material, and the acceptance of SE by the other engineering disciplines), the growing awareness of the criticality of software and its importance in all human-product products is helping us to resolve these difficulties.

Susan Eisenbach

Imperial College Computing has offered a named Software Engineering degree since 1984. It met no opposition either within the Department or within the College, because we were in an Engineering faculty and we've always had an Engineering ethos. We were only a bit concerned that there might not be enough good students who wanted to do it. Although only one of our programmes is named Software Engineering, all our undergraduate degrees are Software Engineering in flavour. The Department has always been in an Engineering Faculty and is funded the same as any other Engineering department. Its degrees are accredited by both the British Computer Society and the Institute of Electrical Engineers for Chartered Engineering status. Intake is highly competitive and we have control over both the number and quality of students who study for all our programmes including our four-year programme MEng Computing (Software

Engineering). Our degrees are extremely popular currently attracting 12 applications for each place.

Within the undergraduate degree, the programme is integrated with substantial components of both theory and practice. Students study 10 courses a year for the first two years almost all required. In their third and fourth years the choice is wide and flexible. There is much practical work. In addition to the integrated lab of the first two years there are group projects every year, a six-month work placement and a substantial final individual project. The students are in great demand upon graduation.

At the Masters level we have two programmes. One is a conversion course that produces Software Engineers. The other is an Advanced course. At this level we feel a more research-based programme is the type of course we wish to run. It's hard to remember when we weren't offering a Software Engineering degree programme.

Steven Seidman

I am the dean of a college of computing sciences (CCS) that currently consists of two departments: computer science (CS) and information systems (IS). CCS is independent of our College of Engineering. One of my primary short-term goals is to develop an undergraduate degree program in software engineering. Each of our departments has faculty members with software engineering research interests: formal methods in the CS department and requirements engineering in the IS department. A software engineering program must therefore cross department boundaries. The path to creating a SE program in this context is far from straightforward.

References

[1] Ford, G. *1990 Report on Undergraduate Software Engineering Education*. (CMU/SEI-90-TR-003). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, 1990.
[2] Naveda, J. F. and Lutz, M. J. "Crafting a Baccalaureate Program in Software Engineering," *Tenth Conference on Software Engineering Education & Training*, Virginia Beach, 1997, pp. 74 – 80.