Creating a Jointly Sponsored Master of Science in Software Engineering

Gregory W. Hislop, College of Information Science and Technology
Spiros Mancoridis, College of Arts and Science
P. M. Shankar, College of Engineering
Drexel University
Philadelphia, PA 19104-2875

Published in:
IEEE / ASEE
Frontiers in Education 1999

Correspondence to:
Dr. Gregory W. Hislop
College of Information Science and Technology
Drexel University
3141 Chestnut St.
Philadelphia, PA 19104

Phone: 215-895-2179
Fax: 215-895-2494
Email: hislopg@post.drexel.edu

Creating a Jointly Sponsored Master of Science in Software Engineering

Gregory W. Hislop, College of Information Science and Technology
Spiros Mancoridis, College of Arts and Science
P. M. Shankar, College of Engineering
Drexel University
Philadelphia, PA 19104-2875

Abstract *- This paper discusses a Master of Science in Software Engineering that is jointly sponsored by three colleges of Drexel University. The sponsors span the disciplinary areas of computer science, electrical and computer engineering, and information systems. The program is a product of the synergy that exists among these three colleges to support software engineering. The combination of disciplinary areas helps provide a broad and deep foundation for the program.*

The paper begins with a discussion of the development of the degree program including the process used and issues that had to be addressed along the way. Next, the curricular content of the degree is outlined, with particular attention to contribution and perspective provided by each of the degree sponsors. Finally, the paper discusses recent developments that will affect the development of software engineering programs today.

Introduction

Software engineering degree programs first emerged in the United States in the 1970's and have continued to spread during the years since. Early programs were at the graduate level, although undergraduate programs have started to appear [6]. Many of these early programs were influenced by a model curriculum developed by the Software Engineering Institute (SEI), but there is substantial variation among the degree programs being offered today.

There are already several jointly sponsored software engineering programs in addition to the one described in this paper, and as interest in software engineering degrees grows, more universities are likely to consider this type of arrangement. Developments in accreditation of software engineering degrees and licensing of software engineering professionals indicate that colleges of engineering will be active in this area. At the same time, many of the faculty members interested in software engineering are in computer science or other departments. This state of affairs will make joint sponsorship an obvious option at many universities.

While the idea of joint sponsorship may make sense in many places, putting that idea into practice has its difficulties. Academic units may have differences in organizational culture, and disciplines may vary in research tradition and curricular expectations. Administrative issues also seem to be more complicated for a jointly sponsored program. All of these factors may make it more difficult to achieve a satisfactory result. The next section describes how these issues affected the development of a software engineering degree at Drexel University.

Degree Program Development

Drexel University is a private, non-sectarian institution with about ten thousand students. The university, which was founded in 1891, has long been widely recognized for its academic strengths in engineering, science, and technology. Given the disciplinary strengths of the university, it is natural that Drexel should have extensive curricular offerings related to computing and information technology, and every college of the university has programs that reflect this. These include computer engineering in the College of Engineering, computer science in the College of Arts and Science, information systems in the College of Information Science and Technology, management information systems in the College of Business and Administration, and digital media in the College of Design Arts.

The Master of Science in Software Engineering (MSSE) is a University degree sponsored by the College of Arts and Science, the College of Engineering, and the College of Information Science and Technology. This joint sponsorship approach builds on existing Drexel strengths in computer science, engineering, and information systems. Drawing on broad strengths of the university was a sensible goal for the MSSE, but it also added to the complexity of developing and administering the program. This section discusses the process of developing the degree with particular emphasis on the issues that relate to joint sponsorship and the approach to addressing the issues at Drexel.

Chronology

Work on an MSSE at Drexel University began as an initiative of the College of Information Science and Technology (IST) in early 1992. The first MSSE students did not begin the degree program until 1997. The phases of activity during the development period can be summarized as follows:

Initial proposal – The IST proposal envisioned a degree offered by IST with operation to begin in fall of

1993. The university approved development of this proposal, and as development work commenced, faculty in both electrical and computer engineering and in computer science expressed interest in the degree, and their respective colleges requested joint discussion of the program.

MSSE Working Group 1 – The Colleges formed an initial working group with several representatives from each of the three colleges. This group established the concept of a jointly sponsored degree and outlined an initial curriculum. The University approved the degree as defined by this group.

MSSE Working Group 2 – Although the first working group outlined an initial curriculum, there was considerable sentiment that the curriculum definition would benefit from additional development. In addition, many of the administrative details of offering a joint degree had not been resolved. The deans of the three colleges created a second, smaller working group consisting of one faculty member from each college to address these issues. This group worked within the framework established by the first working group to refine the curriculum and resolve the administrative issues. The curriculum produced by this group was implemented in 1997.

MSSE Coordinating Group – The members of the second working group continue to serve as faculty coordinators for MSSE issues for each of their respective colleges. They also operate as an informal coordinating group to address issues across college boundaries.

Issues

The five-year chronology outlined above is a rather long process for developing a single degree program. The time was extended in part because of staff changes during the project, but also because of issues particular to software engineering. The paragraphs that follow discuss some of these issues.

Age of the Discipline – The start of software engineering as a discipline is generally traced back no further than a NATO conference on the topic in 1968 [5]. There is still considerable discussion of whether software engineering is a discipline at all. Within the engineering community there is substantial skepticism as to whether software engineering is engineering. Shaw [8] discusses some of the questions related to software engineering as an engineering discipline, and Parnas [7] makes some more detailed arguments for approaching software engineering education from an engineering perspective. Within computer science there is some tendency to treat software engineering as a sub-discipline of computer science, for example in Computing Curricula ’91 [9].

With this context, proposals for software engineering degrees are unlikely to receive unanimous support from the faculty at any large university. At a minimum, there is likely to be a reservoir of skepticism as to whether such a degree is appropriate. This was certainly true in Drexel’s case and the need to consider and discuss doubts and objections contributed to the long development process.

Curricular Model – Software engineering degrees appeared at a variety of institutions starting in the 1970’s, but the number of programs is still fairly small, and the curricula for them varies considerably. Starting in the 1980’s, the Software Engineering Institute (SEI) provided one strong source of support for advancing software engineering education. SEI served as a clearinghouse by tracking general information about degree programs. SEI also produced a model curriculum [3] and quite a few sample course modules. In spite of this work, there still is no widely accepted model curriculum for software engineering. This is particularly true if you consider acceptance beyond the software engineering community by the broader communities of computer science and engineering.

In the case of the Drexel project, the original proposal followed the SEI model quite closely, and all of the Colleges accepted the SEI model as relevant. However, in developing the joint degree, several Colleges felt that the model did not support their particular strengths and domain interests very well, and so several tracks within the final degree are substantially different from the model.

Location and Ownership – If software engineering is indeed a branch of engineering, it makes perfect sense to put software engineering degree programs in colleges of engineering. However, as mentioned previously, the problem with that approach is that many faculty members with interests in software engineering are in other parts of universities.

The software engineering initiative at Drexel started in the College of Information Science and Technology, which has several faculty members whose primary interests are in software engineering. The computer science faculty also includes several members with some interest related to software engineering, and this faculty considers software engineering a potential growth area. At that time, the College of Engineering had no faculty members who would have identified software engineering as their primary area of interest. The College joined the initiative due to a legitimate concern for integrity and credibility of any program related to engineering. In addition, many engineering faculty members work extensively with various types of software including embedded software and software used as a tool to solve engineering problems. Finally, the College’s work in computer engineering implies a need for connections to software engineering.

The range of organizational units with some interest in software engineering made some sort of sharing arrangement a natural option. The approach taken at Drexel was to have the degree established as a University degree, an approach already established at Drexel for other multidisciplinary programs. Under this arrangement, the Colleges jointly sponsor the program, but the University, rather than any specific College, awards the degree.

Language and Culture – The specialization typical of academic disciplines seems to work against collaboration across organizational boundaries. Each of the units involved in this initiative has its own research tradition, academic specialties, and at least some differences in vocabulary. A considerable amount of effort in the MSSE working groups was devoted to establishing working relationships among the faculty members and common understanding of the software engineering curricular material.

Approval Process – Because three academic units were involved, some of the steps of the approval process had to be done three times instead of once. These steps included time required for further discussion and learning about software engineering within the context of each of the College cultures.

Success Factors

There were a number of key factors that were useful in successful establishment of the MSSE. This section discusses those factors and describes how they were useful in addressing some of the issues described above.

Administrative Commitment – Administrative support for the degree helped overcome resistance and generate momentum for the project. The MSSE had general support at the level of Provost, Deans, and Department Heads throughout the project. The support of the Deans of the three colleges in creating and providing a charter for the second working group was particularly important to the final outcome. Without support, especially at key points in the development, progress might have come to a complete halt and the project remained uncompleted.

Industry Demand – The success of other computing and information technology programs at Drexel, and regular expressions of interest in software engineering by people in industry helped generate and maintain momentum for the software engineering initiative. Drexel has strong industry connections, and there are many opportunities for input by people in industry. The MSSE project relied on this type of feedback rather than direct participation in the project by industry representatives.

Outside Review – Academic experts in software engineering from outside of Drexel provided review and input to the original MSSE proposal. This was useful in establishing a sound basis for the initial curriculum plan and gaining credibility for initial approval to develop the degree.

Working Group Size – The second working group consisted of a single representative of each of the three Colleges. This made the process of spanning organizational boundaries much simpler than it would have been with a larger group.

Tolerance and Persistence – In addition to being small, the members of the second working group started with a common determination to produce a complete, manageable product. The group members were persistent in working toward this goal and were willing to make the effort and adjustments necessary to develop a basis for working together productively.

Degree Content and Structure

The MSSE draws on the strengths of each of the sponsoring colleges to provide a curriculum that is both broad and deep. This scope creates a degree that is appropriate for students interested in a wide range of application domains.

It is important to note that the MSSE is multidisciplinary. It encompasses behavioral, managerial, and technical aspects of software engineering and attempts to synthesize -- rather than differentiate -- disciplinary paradigms and themes.

All students in the MSSE take a core curriculum that spans the scope of disciplinary areas relevant to the degree. This core provides a common foundation for all students in the program. Students also elect an area of concentration for their later work on the degree. This track area allows students to take a cohesive, more specialized set of courses that build on the core to support each student's particular career interests.

The first proposal for the MSSE anticipated having a professional degree consisting entirely of course work and a research degree that would include a thesis phase. Later versions of the proposal focused on the professional degree and that is the degree Drexel has implemented.

Admission requirements

Applicants for the MSSE must meet a variety of general admission requirements to establish likelihood of success in a graduate program. These include satisfactory performance in prior academic programs, Graduate Record Exams, and the TEOFL English language exam (for non-English speakers) as well as letters of recommendation. In addition, there are several requirements designed to ensure adequate knowledge at entry to the program. In general the goal is to accept students with a solid foundation in computing and substantial experience with large-scale software systems. The requirements include:

- Applicants must have a prior undergraduate or graduate technical degree. Appropriate undergraduate majors include, but are not limited to, computer science, engineering, information systems, management science, and mathematics.
- Applicants must have knowledge equivalent to an undergraduate course in each of the following:
  - Systems analysis and design or software engineering
  - Data structures and algorithms
  - Discrete mathematics
- Applicants must have advanced capability to program in a block-structured or an object-oriented programming language.

- Applicants must provide evidence of an understanding of the development of the industrial-strength software applications. This requirement may be met by having at least two years experience working directly with software system development, or (with permission of an advisor) extensive software intensive course work.

After consultation with an academic advisor, applicants who are deficient in one or more of the above areas may be required to take up to three foundation courses to prepare them for admission to the MSSE program.

Core Required Courses

There are six core courses in the MSSE covering topics essential for the practicing software engineer. These courses draw on strengths of the sponsoring colleges and cover the general topics of networks and computer hardware (from Engineering), formal software design methods and testing techniques to develop reliable systems (from Computer Science), and human and organizational issues (from Information Science and Technology). The core courses are as follows:

Fundamentals of Computer Hardware – This course provides fundamental knowledge about the organization and architecture of computers. It also introduces the students to the concepts of embedded systems and modeling of systems.

Computer Network Design I – This course is expected to familiarize the students with the art and science of computer networking.

Requirements Engineering and Management – This course presents a life cycle view of requirements issues including: requirements modeling for information process re-engineering, methods for constraint and risk analysis, requirements prioritization and trade-off analyses. The course also examines specific issues in the requirements management process including: requirements agreements, risk management, and requirements change management.

Software Project Management – This course focuses on first-line management of software system development. The major themes of the course are estimation, planning, and execution. Estimation includes software cost factors, estimation models, and risk management. Planning includes work breakdown, scheduling, staffing, resource allocation, and creation of a project plan. Execution includes team building, leadership, process tracking, and communication within and external to the project.

Software Design – This course introduces techniques and notations with formal underpinnings for specifying and documenting structural, architectural, and behavioral properties of software systems. These systems are studied at various levels of abstraction from architectures to subsystem decompositions to module and class interfaces and dependencies. Students learn to analyze, synthesize, and express software designs using a variety of special-purpose design notations.

Dependable Software Systems – In this course, students learn how to formulate specifications using formal and informal notations. They are introduced to programming techniques for reliable programming and to programming languages that facilitate reliable programming through mechanisms such as exception handling. Finally, students learn how to test the implementations of software against their specification using a variety of testing strategies and techniques.

Concentration Tracks

In addition to the required core courses, students in the MSSE complete the requirements in one of three track areas. As with the core, the tracks are designed to draw on strengths of the sponsoring colleges.

Information Science and Technology - Students in this track apply software engineering to information systems problems in commercial organizations and other settings. The principle focus is the process by which user and system requirements are converted into cost-effective, maintainable software systems. This is complemented by a concern for defining, creating, understanding, and evaluating the full range of software life cycle products. The track places particular emphasis on systems values, such as the human-computer interface, front-end user requirements analysis, modeling and validation, and the use of off-the-shelf tools and components to assist in software processes.

Computer Science - Students in this track pursue technical topics pertaining to the development of software systems such as databases, networks, operating systems, graphics and animation systems, compilers, expert systems, and systems for scientific computing. Students use languages and apply techniques to specify, design, implement, test, and maintain software systems.

Engineering - Students in this track pursue techniques to model engineering problems and offer software solutions. The courses in this track emphasize problems facing engineering industries spanning electrical, mechanical, environmental, chemical and others. Systems modeling and simulation techniques are used to solve these problems.

New Developments

Activity that affects software engineering education has accelerated in the last few years. Texas has become the first state to license professional engineers in software engineering [1]. Other states are likely to follow this lead. In addition, ABET, the accrediting body for engineering education, has approved criteria for accrediting engineering programs in software engineering.

A series of committees chartered by the ACM and IEEE-CS related to software engineering education have produced several important products including a draft body of knowledge for software engineering [2] and a standard

curriculum. There is also an SEI-sponsored group working on particular implementation issues for software engineering curricula [4].

All these developments are likely to increase interest in software engineering degrees. They will also provide a level of credibility and fixed reference point not available at the start of the Drexel project. While this will not remove the organizational barriers or need to develop common understanding across faculty groups, it should help to reduce the time needed for this learning process.

References

[1] Bagert, Donald J. "Texas Board Votes to License Software Engineers." *ACM Software Engineering Notes*. 23, 5, p. 7. September 1998.

[2] Dupuis, Robert. *Guide to the Software Engineering Body of Knowledge: A Straw Man Version*. IEEE Computer Society. September 1998.

[3] Ford, Gary, Norman E. Gibbs. "A Master of Software Engineering Curriculum." *IEEE Computer*. September, 1989.

[4] Hilburn, Thomas, et al. "Software Engineering Across Computing Curricula." *Proceedings, Conference on the Teaching of Computing*. Dublin, Ireland. August, 1998.

[5] Naur, P. and B Randall, (eds.) *Software Engineering: A Report on a Conference Sponsored by the NATO Science Committee*. NATO. 1969.

[6] Naveda, J. Fernando, and Michael Lutz. "Crafting a Baccalaureate Program in Software Engineering." *Proceedings, 10th Conference on Software Engineering Education and Training*. IEEE CS Press. 1997.

[7] Parnas, David L. "Education for Computing Professionals." *IEEE Computer*. Pp. 17 – 22. January, 1990.

[8] Shaw, Mary. "Prospects for an Engineering Discipline of Software." *IEEE Software*. 7, 11, pp. 15 – 24. November, 1990.

[9] Tucker, A. B., ed. Computing Curricula 1991: Report of the ACM/IEEE-CS Joint Curriculum Task Force. IEEE Computer Society Press, 1991.