

Software Engineering: Moving From Craft to Profession

Richard E. (Dick) Fairley, Professor
Department of Computer Science and Engineering
OGI School of Science and Engineering, OHSU
dfairley@cse.ogi.edu

Leonard L. Tripp, Technical Fellow
The Boeing Company
leonard.l.tripp@boeing.com

Abstract

There are many opinions concerning the nature of software engineering. Should the practice of software engineering be regarded as an art, a science, or a professional activity? In many organizations, the practice of software engineering is more similar to a medieval craft than to art, science, or profession. However, societal demands for software-intensive systems of high quality require that software engineering be grounded by professional underpinnings. Ongoing developments are building the framework necessary for software engineering to become a recognized profession. This paper provides a status report on various activities, past and present, which provide the basis for transforming software engineering from craft to profession.

1. Introduction

In the recent past, there have been multiple objections to use of the term “software engineering” as a descriptor of work activities, educational programs, and job titles for software developers. These objections are based on the perception that the software engineering discipline does not possess the necessary attributes of an engineering profession. The goal of this paper is to demonstrate that various activities, past and present, provide the basis for transforming software engineering from a craft into profession.

A profession is “An occupation requiring considerable training and specialized skill.” A professional is “a skilled practitioner” [1]. In modern societies, professional workers are generally regarded as practitioners having specialized skill who provide valued services to the citizens of those societies. Modern-day professions are characterized by several attributes: codified bodies of knowledge,

programs of education and training, accreditation of educational programs, professional societies, archival publications, codes of ethics, objective certification procedures, and (in some cases) licensure [2]. Crafts, on the other hand, are characterized by apprenticeship (learning by doing), idiosyncratic approaches, and production of handmade artifacts. Ingenuity of the craftsman is prized over systematic approaches and individualism is prized over teamwork. Craftspeople typically have in-depth areas of specialization and narrow expertise is prized over broad skill sets.

Many aspects of software engineering, as currently practiced, require craft-like skills. Given the current state of software technology, craft skills are necessary for the successful practice of software engineering but they are not sufficient to satisfy the ever-increasing demands for high quality software, developed in a timely and economical manner. Factors that dictate the need for professional engineering skills include the size and complexity of modern software-intensive systems, the variety of skills required to build and maintain these systems, the rate of technological innovation, and the ever-expanding realms of applicability of software with attendant requirements for trustworthiness (safety, security, reliability, and availability). Satisfaction of these needs requires the skills of professional practitioners.

The next section of this paper provides a brief historical overview of some seminal elements in the evolution of the software engineering profession. The subsequent section describes current activities in various areas. The final section of the paper provides some concluding remarks.

2. Foundations for a Software Engineering Profession

The term “software engineering” is credited to Fredrick Bauer of the University of Munich. He and his colleagues first used “software engineering” as a descriptor for a 1968 NATO-sponsored workshop. The reports from the 1968 and 1969 workshops had significant impact in making visible the technical and economic aspects of an emerging software discipline. The following paragraphs describe some of the major events in the journey from craft to profession.

2.1 Professional Societies

Software engineering is served by two professional societies: The IEEE Computer Society (IEEE-CS) and the Association for Computing Machinery (ACM). IEEE-CS is one of 36 special-interest societies of the Institute of Electrical and Electronic Engineers. The IEEE Computer Society traces its origins to the 1946 formation of the Committee on Large-Scale Computing of the American Institute of Electrical Engineers (AIEE). The AIEE and Institute of Radio Engineers merged in 1963 to form the Institute of Electrical and Electronic Engineers. IEEE-CS is dedicated to “advancing the theory, practice, and application of computer and information processing technology” [3]. ACM was founded in 1947 for the purpose of “advancing the skills of information technology professionals and students worldwide” [4]. ACM sponsors 34 special interest groups, including the special interest group for software engineering (SIGSOFT), founded in 1985.

2.2 Conferences and Workshops

The International Conference on Software Engineering was first held in 1975 and continues to be the premier conference on software engineering, under joint sponsorship of ACM-SIGSOFT and IEEE-CS. ACM-SIGSOFT also sponsors an annual conference on the Foundations of Software Engineering. In addition, both organizations sponsor numerous special-interest workshops and seminars on a variety of software engineering topics, some jointly sponsored and some sponsored individually.

2.3 Archival Publications

Software engineering is well-served by two refereed, archival publications: the *IEEE Transactions on Software Engineering* and the *ACM Transactions on Software Engineering and*

Methodology. These journals were founded in 1972 and 1985, respectively. The ICSE conference proceedings also provide a historical record of advances in software engineering. In addition, IEEE SOFTWARE magazine and the ACM Software Engineering Notes newsletter provide timely articles and notifications. Several other journals provide archival records of progress in software engineering. They include Software Practice and Experience, the Journal of Systems and Software, the International Journal of Information Technology, among others.

2.4 Standards

The first software engineering standards committee was formed in 1976 to develop a standard for software quality assurance. Today, IEEE-CS has a Software Engineering Standards Committee (consisting of volunteers), whose purposes are to: “2.1 Codify the norms of professional software engineering practices into standards. 2.2 Promote use of software engineering standards among clients, practitioners, and educators. 2.3 Harmonize national and international software engineering standards development.” [5]. There are currently 39 approved standards covering a wide variety of topics in software engineering. On-going activities are concerned with creating new standards and updating existing ones.

2.5 Education Programs

The distinctions between educational programs in software engineering and computer science are similar to the distinctions between educational programs in electrical engineering and physics. In both cases, scientific programs of research and education are concerned with discovering of basic principles and conveying an understanding of them, whereas the engineering programs are concerned with applying basic principles to solve problems of interest to society and with developing new techniques to solve new problems as they arise.

The first Masters programs in software engineering were introduced around 1980, at Seattle University and the Wang Institute. In the United States, the first undergraduate degree program in software engineering was introduced at Rochester Institute of Technology in 1996 and the first PhD program was introduced at Carnegie Mellon University in 2000. In addition, many organizations, both academic and private, engage in professional development, offering

training courses on numerous topics in software engineering.

2.6 Documented Work Processes

Because software engineering is a people-intensive discipline, process models that specify work activities by which people can work more effectively to produce large, complex software-intensive systems have had a significant impact on the field of software engineering. So-called “capability maturity models” (CMMs) have been developed by a number of organizations; most notably by the Software Engineering Institute (SEI), which was founded by the U.S. Department of Defense in 1983 and is located at Carnegie Mellon University. SEI exists “to help others improve their software engineering capabilities by advancing the state of the practice of software engineering” [6]. A major activity of SEI has been, and continues to be, development of Capability Maturity Models of various kinds that focus on the work processes of software engineering, system engineering, and related work processes. The first CMM for software was released by SEI in 1991. The International Standards Organization is completing work that will culminate in approval of ISO Standard 15504 (the SPICE model) for Software Process Improvement and Capability dEtermination [7].

2.7 Code of Ethics

The *Software Engineering Code of Ethics and Professional Practice, Version 5.2* was adopted in 1999 by the ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices. The code exists in both short and long versions. The preamble to the short version is as follows: “Software engineers shall commit themselves to making the analysis, specification, design, development, testing, and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public, software engineers shall adhere to the following Eight Principles: . . .” The eight principles specify ethical considerations relating to: the public, client and employer, product, judgment, management, profession, colleagues, and self. The code has been jointly approved by ACM and the IEEE-CS as the standard for teaching and practicing software engineering [8].

3. Current Activities

The seminal activities described in the previous section are summarized in Table 1. It is

apparent that the field of software engineering has, over a period of years, made significant progress toward professional status. This section expands on the topics introduced above by presenting recent advances and current activities in various areas. The goal of these efforts is to build on the foundations described above in order to place the software engineering discipline on a solid basis of professional practice.

3.1 A Codified Body of Knowledge

Efforts currently underway to codify a body of knowledge for software engineering include the SWEBOK project and the work of the Software Engineering Standards Committee. The SWEBOK (SoftWare Engineering Body Of Knowledge) effort is concerned with documenting the “generally accepted” body of knowledge in software engineering. “Generally accepted knowledge” is taken to mean knowledge and practices applicable to most projects most of the time, and for which there is widespread consensus about their value and usefulness. SWEBOK incorporates the 10 knowledge areas listed in Table 2. Trial Version 1.0 of SWEBOK can be found at [9]. This trial version is currently in a two-year comment period. Review comments are welcomed.

The Software Engineering Standards Committee of the IEEE Computer Society has the following charter: “Our vision is a family of products and services on software engineering standards for use by practitioners, organizations, and educators to improve the efficiency and effectiveness of their software engineering processes, to improve communication between acquirers and suppliers, and to improve the quality of delivered software and systems containing software.” [10].

Taken together, SWEBOK and the Software Engineering Standards provide impressive documentation of the body of knowledge for software engineering. In addition, journals, conference proceedings, and the many textbooks now available on virtually every aspect of software engineering further codify the software engineering body of knowledge.

3.2 Model Curricula

A Joint IEEE Computer Society/ACM Task Force on “Model Curricula for Computing” (CC) is creating several volumes of curriculum recommendations, including an Overview Volume, a Computer Science Volume, a Computer Engineering volume, a Software

Engineering volume, and an Information Systems volume [11]. A final draft of the Computer Science Volume (CCCS) was approved by the CC-2001 Steering Committee in December, 2001. The other three volumes are currently in preparation. When completed the Software Engineering Volume (CCSE) will provide guidance for implementation and accreditation of software engineering curricula.

Knowledge content for CCSE is being developed by the SEEKA project (Software Engineering Education Knowledge Areas). The goal of SEEKA is to identify an appropriate subset of software engineering knowledge that should be conveyed to undergraduate students in software engineering. The current draft of SEEKA encompasses eight knowledge areas: Foundations, Requirements, Design, Construction, Maintenance, Process, Quality, and Management [12].

It should be noted that SWEBOK and SEEKA have related, but different goals. SWEBOK is intended to provide a guide to the body of software engineering knowledge. SEEKA is intended to specify the knowledge areas that should be covered in an undergraduate curriculum in software engineering. SEEKA is both more and less than SWEBOK; more in the sense that more topics are typically included in an undergraduate program of study than are covered by SWEBOK but less in the sense that complete mastery of the SWEBOK topical areas is not expected of an undergraduate student.

3.3 Accreditation of Education Programs

The Accreditation Board for Science and Engineering (ABET) develops criteria and accredits educational programs in engineering and computer science in the United States and some international locations. From the ABET Web site: "In the United States, accreditation is used to assure quality in educational institutions and programs. Accreditation is a voluntary, non-governmental process of peer review. It requires an educational institution or program to meet certain, defined standards or criteria. Accreditation is sometimes confused with certification. In general, institutions and programs are accredited, and individuals are certified. There are two types of accreditation -- institutional and specialized. Institutional accreditors, such as those referred to as "regional" accreditors, examine the college or university as a whole educational institution. Specialized accreditors evaluate specific

educational programs. Professional accreditors, such as those for medicine, law, architecture and engineering, fall into this category. The Accreditation Board for Engineering and Technology (ABET) is a professional accrediting organization that accredits programs, not institutions. State licensing boards and certification programs may require graduation from an ABET-accredited program as the first step in the registration or certification process for professional practice." [13].

The Computing Accreditation Commission (CAC/ABET) is one of four ABET accreditation commissions. CAC is responsible for accreditation of undergraduate programs in computer science and software engineering [14]. Five universities have requested CAC/ABET accreditation visits to their undergraduate programs in software engineering in Fall, 2002. These will be the first programs to be evaluated using the CAC/ABET guidelines for software engineering programs.

3.4 Certification

Certification is an acknowledgment that an individual has demonstrated mastery of a subject area or a set of skills. Certificates are issued by a certifying agency, based on some objective criteria. Certification is not the same as accreditation; people are certified, academic programs (and other things) are accredited. During the past three years, IEEE-CS has developed a process to certify software practitioners as Certified Software Development Professionals (CSDPs) [15].

CSDP certification includes five components: 1) education: a bachelors degree, 2) experience: approximately five years (9000 hours) spread over 6 of 11 specified areas with some of that experience occurring in at least two of the past four years, 3) examination: 180 computer-delivered questions in a 3.5 hour time limit, 4) review of the code of ethics: see above, and 5) recertification. In order for certificate holders to maintain their proficiency in software engineering as knowledge, methods, and techniques evolve, they must be actively engaged in professional practice, seek opportunities to stay aware of and learn emerging practices, and develop and enhance their skills throughout the duration of their careers. Individuals can retain their CSDP certification by documenting educational activities, publishing, presentations, technical/professional services, self-study, relevant employment, and other activities.

Orientation materials and application procedures for CSDP certification can be found at [15]. Materials on that Web site include recommended references, sample questions, a guide to software engineering terminology, and a resource guide. As of August 2002, 218 software practitioners have taken the examination. 178 have passed and are certified by the IEEE Computer Society as Certified Software Development Professionals.

3.5 Licensure

Accreditation is not the same as certification and certification is not the same as licensure. Programs are accredited and people are certified. A license conveys official or legal permission to do or to own a specific thing. Licensing procedures are typically implemented by governmental agencies. A license to do or to own a certain thing is required when the activity or thing becomes a matter of concern to the interest of public health, safety, or welfare. Governmental agencies that currently require some form of licensure for software engineers include some governmental agencies of the United States, Texas, Canada, British Columbia, and Ontario.

It is a matter of speculation as to whether or when licensing of software engineers will become a pervasive requirement for engaging in the practice of software engineering. It is encouraging to observe that the efforts described in this paper provide a solid foundation for development of licensing criteria for software engineers, should it ever be required.

4. Conclusion

An additional hallmark of a professional discipline is that of providing service, of a specialized nature, to clients. Software engineering is a service-oriented discipline: the products of software engineering are sometime developed for individual clients and sometimes for markets that include many clients, but without clients there would be no software engineering discipline. In any case, the efforts of software engineers have widespread and growing impacts on modern society. It is thus fitting that there be public concern about the professional status of software engineers, and additionally fitting that software engineers are taking the necessary steps to provide the framework elements for a profession of software engineering. As demonstrated by this paper, and

summarized in Table 3, these steps are well underway.

5. References

- [1] *The American Heritage College Dictionary*, Third Edition, Houghton Mifflin Company, 1993.
- [2] Ford, G. & Gibbs, N. *A Mature Profession of Software Engineering*, SEI Report CMU/SEI-96-TR-004, Carnegie Mellon University, 1996.
- [3] www.computer.org: home page of the IEEE Computer Society.
- [4] www.acm.org: home page of the Association for Computing Machinery.
- [5] standards.computer.org/sesc/sesc_pols/SESCCharter.htm: charter page of the Software Engineering Standards Committee of IEEE-CS.
- [6] www.sei.cmu.edu: home page of the Software Engineering Institute of Carnegie Mellon University.
- [7] www.squ.gu.edu.au/spice/contents.html: SPICE documents, V 2.0.
- [8] computer.org/certification/ethics.htm: short and long forms of the ACM/IEEE-CS code of ethics.
- [9] www.swebok.org: home page of the Software Engineering Body of Knowledge.
- [10] standards.computer.org/sesc/: Home page of the Software Engineering Standards Committee of IEEE-CS.
- [11] www.computer.org/education/cc2001: home page for the Model Curricula for Computing series.
- [12] sites.computer.org/ccse/artifacts/KADescriptions.htm: home page for the Software Engineering Education Knowledge Area descriptions.
- [13] www.abet.org/about_ac.html: home page for ABET accreditation guidelines.
- [14] www.abet.org/cac1.html: home page for the Computing Accreditation Commission of ABET.
- [15] computer.org/certification: home page for the Certified Software Development Professional program of the IEEE Computer Society.

Table 1. Evolution of professional activities in software engineering

DATE	EVENT
1968	Term “software engineering” coined
1975	First International Conference on Software Engineering
1976	First software engineering standards committee formed
1980	First Masters programs in software engineering
1982	IEEE introduces Transactions on Software Engineering
1983	IEEE introduces SOFTWARE magazine
1985	ACM introduces Transactions on Software Engineering and Methodology
1985	ACM introduces Software Engineering Notes newsletter
1991	Version 1.0 of the SEI Software Capability Maturity Model introduced
1996	First undergraduate degree program in software engineering
1999	ACM & IEEE-CS jointly endorse a software engineering code of ethics
2000	First Ph.D. degree program in software engineering
2001	Trial Version 1.0 of Software Engineering Body of Knowledge published for comments
2001	Committee formed to define Software Engineering Education Knowledge Areas
2002	First ABET/CAC accreditation visits for undergraduate software engineering degree programs
2002	First examinations conducted to certify software development professionals

Table 2. The ten knowledge areas of SWEBOK

Software Requirements
Software Design
Software Construction
Software Testing
Software Maintenance
Software Configuration Management
Software Engineering Management
Software Engineering Process
Software Engineering Tools and Methods
Software Quality

Table 3. Foundational elements of a software engineering profession with examples

Elements	Examples
Professional societies	ACM-SIGSOFT, IEEE-CS
Conferences and workshops	ICSE, FSE, numerous workshops
Archival publications	IEEE TSE, ACM TOSEM
Standards	IEEE-CS SESC
Magazines and newsletters	IEEE SOFTWARE, ACM SEN
Educational programs	U/g, Masters, and PhD
Documented work processes	CMMs, SPICE
Code of ethics	ACM / IEEE
Codified body of knowledge	SWEBOK
Model curricula	SEEKA / CCSE
Accreditation of educational programs	ABET / CAC
Certification	CSDP
Licensure	Some agencies of U.S. Gov't., Texas, & Canada