For many programmers, software development consists of hacking. As we mature, it is time to follow the example of other professional disciplines, to put the *engineering* in software engineering. The guest editors look at what has been done and what still needs to be done to establish our profession.

Professional Software Engineering: Fact or Fiction?

Steve McConnell, Construx Software
Leonard Tripp, The Boeing Company

The most common approach to software development today is code-and-fix programming—hacking. In this approach, a development team begins with a general idea of what they want to build. They might have a formal specification, but probably not. They use whatever combination of informal design, code, debug, and test methodologies suits them. Programmers write a little code and run it to see whether it works. If it doesn't work, they change it until it does. The code-and-fix approach is far from the state of the art in software development. It costs more, takes longer, and produces lower-quality software than other approaches; its main advantage is that it requires little technical or managerial training. Leading organizations have known and used effective software development practices for decades, but the gap between average practice and best practice in software is enormous.

If one end of the software development competency spectrum is occupied by

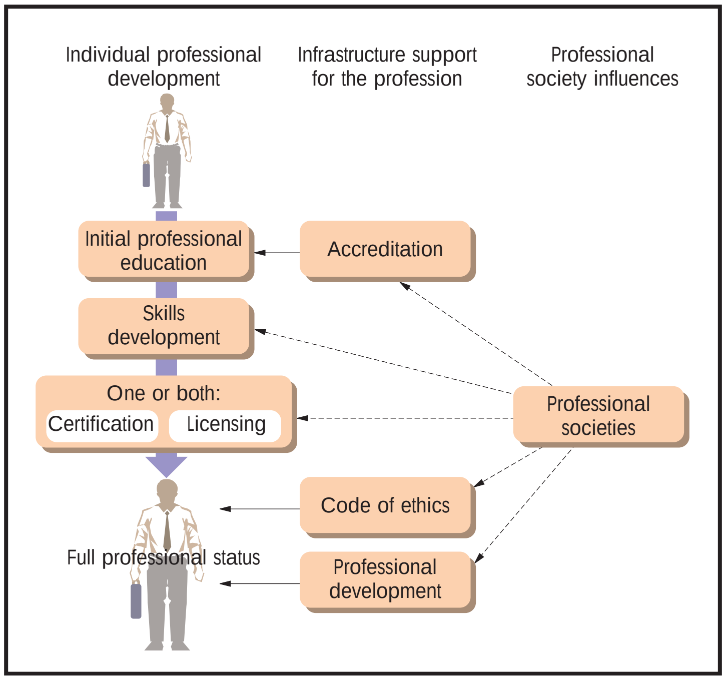


Figure 1. Professional development follows most or all of these basic steps in all well-established professions.

code-and-fix development, the other end is occupied by software engineering—"the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software," as defined by *IEEE Standard 610.12*.

Recently, the software engineering community has seen encouraging developments related to establishing, supporting, and disseminating a higher standard of conduct for practicing software developers. This special issue focuses on recent developments that support a true profession of software engineering.

Elements of a Profession

In 1996, Gary Ford and Norman E. Gibbs published a report titled *A Mature Profession of Software Engineering*. They studied several well-established professions, including medicine, law, engineering, and accounting. They observed that professionals in other fields follow a professional-development path that is fairly similar, regardless of their specific discipline. Figure 1 shows these characteristics.

Mature professions include the following elements:

- Initial professional education
- Accreditation
- Skills development
- Certification
- Licensing
- Professional development
- Professional societies
- Code of ethics

Initial professional education

Professionals generally begin their careers by completing a university program in their chosen field, such as medicine, engineering, or law.

The predominate form of education in software has been an undergraduate degree in computer science. Recently, progress has been made in defining undergraduate programs in software engineering. The term *software engineering* is often misunderstood, and one way to clarify it is to differentiate software engineering from computer science. In "Software Engineering Programs Are Not Computer Science Programs," reprinted in this issue, David L. Parnas describes the differences between the two, presenting the clearest explanation of software engineering we have read.

Accreditation

University programs are accredited by oversight bodies that determine whether each program provides adequate education. This assures that, as long as professionals graduate from accredited programs, they will start their professional lives with the knowledge they need to perform effectively.

Accreditation is becoming increasingly important as software engineering programs proliferate. Since the Rochester Institute of Technology initiated the first undergraduate software engineering program in the US in 1996, many universities have initiated similar programs including Auburn University, the Milwaukee School of Engineering, Monmouth University, and Montana Tech. In Canada, Concordia University, McMaster University, Memorial University of Newfoundland, and the University of Ottawa offer bachelor's programs. Several other North American universities are actively considering adding programs. At least 13 universities in the UK and six more in Australia offer undergraduate programs.

Accreditation of these programs is not yet in place. Gerald Engel describes what accreditation criteria will look like in "Program Criteria for Accreditation of Academic Programs in Software Engineering." The work Engel describes will significantly influence the education and the skills that the first wave of undergraduate software engineering students will take into industry.

Skills development

For most professions, education alone is not sufficient to develop full professional capabilities. Nascent professionals need practice applying their knowledge before they are prepared to take primary responsibility for performing work in their field. In the US, physicians generally have a three-year residency. Certified Public Accountants (CPAs) must work one year for a board-approved organization before receiving their licenses. Professional engineers must have at least four years of work experience. Requiring some kind of apprenticeship assures that people who enter a profession have practice working at a satisfactory competence level.

What knowledge is important to a professional software engineer? The IEEE Computer Society and the ACM have been working to define that knowledge. Pierre Bourque, Robert Dupuis, Alan Abran, James W. Moore, and Leonard Tripp describe the work in progress in "The Guide to the Software Engineering Body of Knowledge." This project is tremendously important, because the body of knowledge will affect university curricula and standards for licensing and certification exams.

Certification

After completion of education and skills development, a professional must pass one or more exams that assure he or she has attained a minimum level of knowledge. Doctors take board exams. Accountants take CPA exams. Professional engineers take a Fundamentals of Engineering exam at college graduation and then take an engineering specialty exam about four years later. Some professions require recertification from time to time. Certification is a voluntary process that helps the public determine who is fully qualified to participate in a profession and who isn't.

In the UK, the Institution of Electrical Engineers conducts a software engineering certificate program and the British Computer Society has a professional development scheme. The Australian Computer Society offers a certification program in information technology with a subspecialty in software engineering. In the US, there is no certification program in software engineering per se. The Institute for Certification of Computing Professionals offers a Certified Computing Professional designation. The American Society for Quality offers programs for Certified Software Test Engineer and Certified Software Quality Engineer. And numerous companies including Microsoft, Novell, and

RESOURCES

PRINT RESOURCES

Gary Ford and Norman E. Gibbs, *A Mature Profession of Software Engineering*, Software Engineering Inst., Carnegie Mellon Univ., Tech. Report CMU/SEI-96-TR-004, 1996. This technical report contains a detailed study of the elements that make up mature professions and an analysis of software engineering's maturity. This report is downloadable from the SEI's Web site at http://www.sei.cmu.edu.

Steve McConnell, *After the Gold Rush: Creating a True Profession of Software Engineering*, Microsoft Press, Redmond, Wash., 1999. This book is a set of essays that argue in favor of treating software development as an engineering discipline. It discusses the state of the art in software engineering and the work needed to establish software engineering as a profession.

WEB RESOURCES

Software Engineering Coordinating Committee: http://computer.org/tab/swecc. This Web site contains current information related to the IEEE Computer Society and ACM initiative to define software engineering as a profession.

Software Engineering Body of Knowledge: http://www.swebok.org. This Web site describes the work in progress on the software engineering body of knowledge project.

Software engineering code of ethics: http://computer.org/tab/swecc/code.htm. This Web site contains the text of both the short and long versions of the Software Engineering Code of Ethics and Professional Conduct adopted by the IEEE Computer Society and ACM.

Association of Professional Engineers and Geoscientists of British Columbia Software Engineering: http://www.apeg.bc.ca/cse/cse.htm. This site contains white papers related to British Columbia's software engineering educational and licensing initiatives.

The Texas Board of Professional Engineers software engineering licensing page: http://www.main.org/peboard/sofupdt.htm. This describes Texas's program for licensing professional engineers in software.

ACM's position on software licensing: http://www.acm.org/serving/se_policy. This site contains the ACM's position on licensing of software engineers.

University of Texas's professional-ethics site: http://www.cs.utexas.edu/users/ethics/professional.html. This site contains numerous links related to ethics, standards, education, and professionalism in software engineering and computer science.

ACADEMIC PROGRAMS

Rochester Institute of Technology: http://www.se.rit.edu. This site describes RIT's program, including detailed class listings and descriptions.

McMaster University: http://www.cas.mcmaster.ca/cas/undergraduate/SEprogrammes.html. This site describes McMaster's program, including detailed class listings and descriptions.

How You Can Participate

IEEE Computer Society Web site: http://computer.org. This site contains links to committees, conferences, and other opportunities to be professionally active in software engineering through the IEEE Computer Society.

ACM Web site: http://www.acm.org. This site contains links relating to participating in software engineering through the ACM.

Construx Software's software engineering professionalism site: http://www.construx.com/profession. This site contains information about how you can participate in the emerging profession of software engineering. It contains pointers to current information about university courses, state-by-state licensing initiatives, and other resources. It also contains detailed reading lists for self-study programs in software engineering.

Apple Computer provide various certifications related to their products.

The IEEE Computer Society began developing a certification program for software engineering practitioners in June 1999; it expects certification to be available in the third quarter of 2000.

Licensing

Licensing is similar to certification except that it is mandatory and it is administered by a government authority. Many exciting developments are occurring in this area. Texas began licensing professional software engineers in 1998. (For more information, see *IEEE Software*'s report "License to Code," by John Charles (Sept./Oct. 1998) and his recent update "Software Engineering Licensing Weathers Challenge" in Sept./Oct. 1999.) In June 1999, British Columbia began licensing professional engineers in software, and in September 1999, Ontario followed suit.

What are the implications of licensing? Is it a good idea? Licensing evokes strong reactions ranging from "Of course!" to "Hell, no!" For one view, see John Speed's article, "What Do You Mean I Can't Call Myself a Software Engineer?" Speed is a leader in promoting the licensing of professional engineers with a specialty in software engineering in the US. The article provides key information about the licensing process in the US. The basic principles of licensing apply in other countries. The article sidebars describe licensing policies in British Columbia, Ontario, and the United Kingdom. For other views of licensing, see this issue's Point/ Counterpoint by Dennis Frailey and Tom DeMarco. Frailey sees licensing as a necessary and valuable step in assuring professional competency; DeMarco compares it to a Soviet bureaucracy.

Professional development

Ongoing professional education maintains or improves workers' knowledge and skills after they begin professional practice. Requirements for professional development tend to be strongest in professions that work with a rapidly changing body of technical knowledge. Medicine is perhaps the most notable because of the constant improvements in drugs, therapies, medical equipment, and diagnosis and treatment procedures. After a professional's initial education and skills development are complete, ongoing education requirements help to assure a minimum competency level throughout the professional's career.

One aspect of professional development is learning appropriate standards of practice. The IEEE has been active in developing software engineering standards for more than 20 years, and collectively the IEEE standards make up a kind of practice standard for software engineers. James W. Moore describes the IEEE's recent efforts in "An Integrated Collection of Software Engineering Standards." Moore's article is a useful introduction to these tremendously valuable development resources, and it includes an interesting update about the IEEE's recent efforts to unify its software engineering standards.

Professional societies

Professionals see themselves as part of a community of like-minded individuals who put their professional standards above their individual self-interest or their employer's self-interest. In the beginning, professional societies usually promote the exchange of knowledge. Over time, their function evolves to include defining certification criteria, managing certification programs, establishing accreditation standards, and defining codes of ethics and disciplinary action for violations of those codes.

The IEEE Computer Society and the ACM have been active in defining the profession of software engineering. This work is orchestrated by the Software Engineering Coordinating Committee (SWECC). For more details, see the sidebar "Software Engineering Coordinating Committee."

Code of ethics

Each profession has a code of ethics to ensure that its practitioners behave responsibly. The code states not just what its practitioners actually do, but what they should do. Professionals can be ejected from their professional societies or lose their license to practice for violating the code of ethics. Adherence to a recognized code of conduct helps professionals feel they belong to a well-regarded community, and enforcement of ethics standards helps maintain a minimum level of conduct.

In 1998, the IEEE Computer Society and the ACM adopted a Software Engineering Code of Ethics and Professional Practice. Don Gotterbarn describes the code in "How the New Software Engineering Code of Ethics Affects You." This code supports a mature profession of software engineering by explicitly defining the ethical obligations of a software engineer.

Onward and Upward

Developments underway will affect all software developers directly or indirectly. Are they positive? In our opinion, they are both positive and necessary to raise the level of software development competence. But many of these developments are still in their early stages, and interested software practitioners can contribute to defining the body of knowledge, establishing curriculum standards, shaping licensing policies, and many other areas.

IEEE Software's mission is to "Build the Community of Leading Software Practitioners." We hope that you will help us in this mission by taking an active role in development of software engineering as a profession, including sending us your comments about the initiatives described in this issue (software@computer.org). ❖

Call for Papers

The Eleventh International Symposium on Software Reliability Engineering

8th October - 11th October 2000

San Jose, California, USA

IEEE COMPUTER SOCIETY

In Collaboration with the International Conference on Software Maintenance

Sponsored by the IEEE Computer Society

Organized by The Committee on Software Reliability Engineering of the IEEE Computer Society and theTechnical Council on Software Engineering

Detailed information on both conferences may be found at:

http://www.rstcorp.com/conferences/issre2000/

http://www.rstcorp.com/conferences/icsm2000/

Submissions to ISSRE'2000 should report promising new research breakthroughs or advances to existing literature. Topics of interest include:

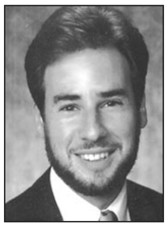
Practical applications of reliability modeling	Fault-tolerant and robust software
Measurement for software reliability assessment	Reliability of distributed systems
Software reliability models	SRE tools, education, and technology transfer methods
Software testing and verification	Software reliability standards and legal issues
Software safety	Building high-integrity mobile code systems

Electronic submission is encouraged. Electronic submissions may be in PDF or portable PostScript. Detailed submission information can be found at http://www.cs.uidaho.edu/~ISSRE2K/submission. All electronic submissions should be sent to: issre2k@cs.uidaho.edu Hardcopy submissions (6 copies) should be sent to one of the Program Co-Chairs:

Dr. John C. Munson	**Prof. Fevzi Belli**	**Prof. Mitsuru Ohba**
Computer Science Department	University of Paderborn	Computer Science Department
University of Idaho	Dept. of Electrical and Electronics Engineering	Hiroshima City University
Moscow, ID 83844-1010	33095 Paderborn, Germany	Hiroshima, Japan 731-31
jmunson@cs.uidaho.edu	belli@adt.uni-paderborn.de	ohba@cs.hiroshima-cu.ac.jp
1.208.885.7789		

Relevant Dates: Paper Abstracts **(01FEB00)**, Papers **(01MAR00)**,Industry Practice Abstract, Panel Proposals , Tutorial Proposals, Tools Fair Entries**(01APR00)**

About the Authors



Steve McConnell is president and chief software engineer of Construx Software, where he divides his time between leading custom software projects, teaching classes, and writing books and articles. He is also Editor-in-Chief of *IEEE Software*. He is the author of *Code Complete*, *Rapid Development*, and *Software Project Survival Guide*. In 1998, readers of Software Development magazine named him one of the three most influential people in the software industry, along with Bill Gates and Linus Torvalds. His most recent book is *After the Gold Rush: Creating a True Profession of Software Engineering* (Microsoft Press, 1999). He is on the panel of experts that advises the Software Engineering Body of Knowledge project and is a member of the IEEE and ACM. He can be reached at stevemcc@construx.com.

Leonard Tripp is a technical fellow in software engineering at The Boeing Company. He is the 1999 president of the IEEE Computer Society. His interests include engineering practices, standards, and tools for safety-critical airborne digital systems. He is the chair of the Industry Advisory Board for the Software Engineering Body of Knowledge project and the IEEE Computer Society and ACM Software Engineering Coordinating Committee (SWECC). He served as head of the US delegation to the international committee for software engineering standards from 1993 to 1998. His IEEE Computer Society awards include the 1996 Hans Karlsson Award for outstanding leadership in standards development, a Meritorious Service Award, and an Outstanding Contribution Award. He can be reached at l.tripp@computer.org.

The Software Engineering Coordinating Committee

SWECC (the official name is "SWEcc") was formed 1 January 1999 to foster the evolution of software engineering as a professional computing discipline. This committee replaced a prior ad hoc committee, the Joint IEEE Computer Society and ACM Steering Committee for the Establishment of Software Engineering as a Profession, formed by the two societies in January 1994. The committee's purpose is "to establish the appropriate sets(s) of criteria and norms for professional practice of software engineering upon which industrial decisions, professional certification, and educational curricula can be based."

Since 1993, the IEEE Computer Society and the ACM have been actively promoting software engineering as a profession, notably through their involvement in the IEEE Computer Society and ACM, the original joint committee, and now SWECC. Achieving consensus by the profession on a core body of knowledge is a key milestone in all disciplines and has been identified by the SWECC as crucial for the evolution of software engineering toward a professional status. The development of the Guide to the Software Engineering Body of Knowledge will follow a complete survey to clarify the knowledge, skills, and scope of responsibilities required of software engineers of various types and seniority, with designated knowledge area specialists responsible for identifying the topics within the assigned knowledge area and identifying reference material for each topic. Visit http://www.swebok.org for information and recent developments.

The Joint Task Force on Software Engineering Ethics and Professional Practice, under the auspices of the original joint committee, developed a Software Engineering Code of Ethics and Professional Practice that documents software engineers' ethical and professional responsibilities and obligations. Both the IEEE Computer Society and ACM approved the code in 1998. The task force in charge of the document has circulated it and is collecting comments from software engineering professionals around the world. Visit http://www.computer.org/tab/swecc/SWCEPP.htm for more information on this document and its task force.

In 1997, the Software Engineering Education Project was created, also under the auspices of the original joint committee, with a mandate to develop curriculum recommendations for software engineering programs. The project chose to work on undergraduate education first and began by developing accreditation criteria appropriate for undergraduate programs. The intent was to use accreditation criteria as a specification for developing a model undergraduate program. This effort was completed in mid 1998, and the IEEE Computer Society and ACM approved the accreditation criteria in November 1998. For more information and a summary of the project results, visit http://www.computer.org/tab/swecc/sweep.htm.

A not-too-surprising lesson learned by all volunteers involved in the first few years of this committee's activity was that things take a lot more time and effort than predicted. These early achievements are just a small but important step in the definition of a new profession, the full definition of which will take a few more years to achieve. The committee's future plans include continued and expanded work on the software engineering body of knowledge, education, and the code of ethics. To get involved or find more information, visit the IEEE Computer Society and ACM SWECC Web site, http://www.computer.org/tab/swecc.