# ACM TASK FORCE ON LICENSING OF SOFTWARE ENGINEERS WORKING ON SAFETY-CRITICAL SOFTWARE

***Draft Report - July, 2000***


| | |
|---|---|
| John Knight | *University of Virginia* |
| Nancy Leveson | *Massachusetts Institute of Technology* |
| Lori Clarke | *University of Massachusetts* |
| Michael DeWalt | *Certification Services, Inc.* |
| Lynn Elliot | *Guidant, Inc.* |
| Cem Kaner | *Florida Institute of Technology* |
| Bev Littlewood | *City University, London, UK* |
| Helen Nissenbaum | *Princeton University* |


# ACM TASK FORCE ON LICENSING OF SOFTWARE ENGINEERS WORKING ON SAFETY-CRITICAL SOFTWARE

## BACKGROUND AND SCOPE OF REPORT

Two years ago, ACM was approached by the Texas Professional Engineers Licensing Board for help in defining performance criteria for software engineering licensing examinations to be administered in Texas. At the time, ACM agreed to expand its on-going work with the IEEE Computer Society and SWECC to include recommendations related to licensing examinations. This agreement was reached despite the fact that some members of ACM Council had reservations about whether licensing software engineers was "in the best interests of the field of computing and the public."

In 1998, Barbara Simons, the president of ACM, formed an ACM Advisory Panel on Professional Licensing of Software Engineers headed by Fran Allen and Paula Hawthorne and composed of Barry Boehm, Fred Brooks, Jim Browne, Dave Farber, Sue Graham, Jim Gray, Ken Kennedy, Nancy Leveson, Dave Nagel, Peter Neumann, Dave Parnas, and Bill Wulf. Although the committee could not reach consensus about licensing, the majority recommended against. The final report of the advisory panel is included as an appendix to this report.

At the ACM Council Meeting in May 1999, after reviewing the advisory panel's report and following a lengthy discussion, the ACM Council passed the following motion:

> *"ACM is opposed to the licensing of software engineers at this time because ACM believes it is premature and would not be effective at addressing the problems of software quality and reliability.*
>
> *ACM is, however, committed to solving the software quality problem by promoting R&D, by developing a core body of knowledge for software engineering, and by identifying standards of practice."*

The ACM Council decided to remain a member of SWECC. To determine how much support ACM should be providing to licensing activities, in the summer of 1999 Barbara Simons created two blue-ribbon task forces: (1) to evaluate the Software Engineering Body of Knowledge Activities (SWEBOK) and (2) to determine ways in which ACM and the profession might improve the robustness and quality of safety-critical software and to evaluate licensing activities in this context. This report describes the initial findings and recommendations of the latter committee. A more complete report will follow later.

Nancy Leveson and John Knight agreed to head the second committee and selected members who had particular knowledge and expertise in various facets of the issue: Lori Clarke,

 

University of Massachusetts, Amherst; Michael DeWalt, Certification Services Inc. and former National Resource Specialist for Software, FAA; Lynn Elliot, Vice President, Guidant Corporation; Cem Kaner, consultant, software testing expert and lawyer; Bev Littlewood, Center for Software Reliability, City University, London; and Helen Nissenbaum, Princeton University, an expert in professional ethics.

Barbara Simons asked the task force to look at licensing in more depth than was possible by the previous advisory panel and to focus on implications for safety. Before deciding on a particular approach to ensuring safety within software engineering (such as licensing), the task force chairs (Leveson and Knight) felt that a careful evaluation of all the alternatives for achieving this goal as well as their potential effects on our profession should be undertaken. These alternatives include voluntary and mandatory licensing, government regulation and oversight, voluntary product certification, insurance, standards, codes of practice, independent inspection (IV&V), ethical standards, liability and legal remedies, accreditation of educational institutions, and specification of a standard curriculum but without formal accreditation.

In April 2000, the task force held a fact-finding meeting in Washington D.C. At that meeting, the task force had discussions with and got information from a group of invited experts. The presenters were: John Calvert, U.S. Nuclear Regulatory Commission; Paul Jones, U.S. Food and Drug Administration, Center for Devices and Radiological Health; Patrick Natale PE, Executive Director National Society of Professional Engineers (NSPE); Arthur Schwartz, Deputy Executive Director & General Counsel, National Society of Professional Engineers; John Adams PE, National Council of Examiners for Engineering and Surveying (NCEES); and James Moore, Mitre Corp, member of SWECC and SWEBOK.

Although the investigations are not yet complete and the final task-force report will not be finished until later in 2000, Barbara Simons asked the task force to provide an update of its activities and findings for the May 2000 meeting of the ACM Council. We include in this report only the findings, conclusions, and recommendations related to PE licensing and the software-body-of-knowledge activities.

# THE PROBLEM AND THE ORGANIZATION OF THE REPORT

Software is increasingly being used in ways that affect public safety, where safety is defined in the general sense as involving the possibility of unacceptable loss (including human life or injury, property damage, environmental pollution, societal disruption, large monetary loss, etc.). While the first such software was carefully crafted by experts and was limited in its functionality, more extensive use and increased complexity of the software being used is leading to increasing numbers of software-related accidents and losses. Accidents almost always have multiple causal factors, but software design has played a role in accidents involving, for example, medical devices, aircraft and other transportation systems, space vehicles, and defense systems.

Some people building software to control such systems are limited in experience and qualifications (although there is no data as to what percentage of developers this might be), and this has led to suggestions by some government entities, independent licensing bodies, and

 

individuals to certify or license software engineers. Before rushing down a path that might not solve the problem, however, or might create even worse ones, we need to consider carefully the various solutions and determine how best to protect the public without unduly affecting engineering progress, the economy, or individual rights to practice a profession or hold a job.

This initial report contains the findings and recommendations of the task force resulting from our internal discussions as well as interviews and meetings with experts about licensing and the software body of knowledge activities. Our findings and recommendations about alternatives to licensing, i.e., government regulation and oversight, product certification and warranties, insurance, standards, codes of practice, independent inspection, ethical standards, liability and legal remedies, accreditation of educational institutions, and specification of a standard curriculum but without formal accreditation, will be contained in the final report of the task force.

# LICENSING

A Professional Engineer (PE) is an individual who has been granted by a governmental jurisdiction the right to use that title and offer professional engineering services (i.e., has been licensed) based on a model process that includes a four-year degree from a university program accredited by the Engineering Accreditation Committee (EAC) of the Accreditation Board for Engineering and Technology (ABET); an eight-hour examination on fundamentals of engineering taken usually in the senior year of college; four years of acceptable experience; a second examination on principles and practice; written recommendations from other professional engineers; and, in many states, mandatory continuing education/development.

The practical implications of the PE license and the percentage of engineers licensed varies by engineering subdiscipline. Few engineers actually are licensed as PEs, with the majority being in civil engineering (about 40% of civil engineers) and the second largest being mechanical engineers. The lowest percentage is electrical engineers. There is currently a decreasing percentage of mechanical and electrical engineers going through the licensing process while the percentage of civil engineers is holding steady.

It is important to note that the PE license in most jurisdictions says “Professional Engineer” and does not distinguish between subdisciplines. A PE can, however, be disciplined by the state (fined or lose their license) if they violate state rules of professional conduct or practice beyond their area of competence. Licensed professional engineers are accountable for their activities and assume personal legal liability.

Licensing is a state regulated activity. The National Council of Examiners for Engineering and Surveying (NCEES) provides a model law and The National Society of Professional Engineers (NSPE) lobbies legislatures to adopt licensing laws and regulations to protect the public health and safety. Every state currently has an engineering licensure law. Who is an engineer is defined by “practice acts,” i.e., defined by what the person does and not simply what they call themselves, although “title acts” also exist but only in California. Using the title “engineer” or practicing as an engineering professional in a state and in a manner for which licensing is

 

required is illegal. The rules for the licensing of PEs vary from state to state. Enforcement also varies. For example, some states (and the model law) require that faculty teaching engineering design classes must be licensed but this part of the law is often not strictly enforced. However, it could apply potentially to any faculty member teaching a class in software engineering.

A professional engineer in a practice for which licensing is required must be licensed in every state in which he or she practices. Mandatory licensing is usually required for those providing services directly to the public and those involved in the design of facilities, roads, transportation and construction where drawings etc. must be submitted to state agencies to be approved and which must be signed and sealed by the professional engineer. There are sometimes exceptions to required licensing, mostly notably if the person works in industry (for a company) and does not provide direct services to the public and if the person is an employee of the federal government.

Licensing activities within each state are overseen by a state licensing board(s) whose members are usually appointed by the Governor often upon the recommendation of the professional engineering societies. These boards act as a state agency and are supported by state appropriations. Most states charge licensing fees, ranging up to $200 per year. In many states, these fees go into the state's general appropriations budget. Persons licensed in multiple states must pay the fees for each state in which they practice their profession. Some states allow granting licenses on the basis of being licensed in a different state whose requirements "meet or exceed" those of the state in question.

Current PE activities with respect to software are limited to a very few states, most notably Texas. Some legitimate concern among engineers and a few other states about licensing software developers appears to be driven by such activities as those by Novell and Microsoft to issue a certification of "Novell Certified Network Engineers" and "Microsoft Certified System Engineers." In these cases, most people would agree that the use of the terms "network engineer" and "system engineer" are clearly being abused and might be confusing to the public.

The extension of concern beyond these limited cases is more problematic.

**FINDING:**

*Presently, licensing as PEs would be impractical for software engineering.*

1. As part of the PE licensing process, everyone must take an eight hour Fundamentals of Engineering examination (FE) composed of a morning general examination designed to match the ABET criteria for the first two years of an engineering degree and a discipline-specific afternoon examination that covers the material taught in the last two years of an engineering degree program.

   The morning general examination (that must be taken by everyone) covers:

   - *Chemistry:* Acids and bases; equilibrium; equations; electrochemistry; inorganic chemistry; kinetics; metals and nonmetals; nomenclature; organic chemistry; oxidations and reduction; periodicity; states of matter; solutions; and stoichiometry.

 

- *Computers:* Algorithm flowchart; spreadsheets; pseudocode; and data transmission and storage.
- *Dynamics:* Force, mass, and acceleration; friction; impulse and momentum; kinematics; vibrations; and work and energy.
- *Electrical Circuits:* AC circuits; diode applications; DC circuits; electric and magnetic fields; capacitance and inductance; ideal transformers; Fourier and Laplace transforms; and operational amplifiers (ideal).
- *Engineering Economics:* Annual cost; breakeven point; benefit-cost analysis; future worth or value; present worth; and valuation and depreciation.
- *Ethics:* Relations with clients, peers, and the public.
- *Fluid Mechanics:* Flow measurement; fluid properties; fluid states; impulse and momentum; pipe and other internal flow; and similitude and dimensional analysis.
- *Material Science/Structure of Matter:* Atomic structure; crystallography; corrosion; diffusion; materials; binary phase diagrams; properties; and processing and testing.
- *Mathematics:* Analytic geometry; differential equations; differential calculus; difference equations; integral calculus; linear algebra; Laplace transforms; probability and statistics; roots of equations; and vector analysis.
- *Mechanics of materials:* Beams; bending; columns; combined stresses; shear; stress and strain; tension and compression; and torsion.
- *Statics:* 2-dimensional equilibrium; 3-dimensional equilibrium; centroid of area; concurrent force systems; friction; moment of inertia; and vector forces.
- *Thermodynamics:* 1st and 2nd law; availability-reversibility; cycles; energy, heat, and work; ideal gases; mixture of gases; phase changes; enthalpy, entropy, and free energy properties; and thermodynamic processes.

The afternoon, discipline-specific examination is offered in five disciplines and one general, non-specific discipline. The general, non-specific examination covers the same topics as in the morning examination but in more depth. A discipline-specific examination may be taken in chemical, civil, industrial, electrical, or mechanical engineering. Because of the requirements for there to be at least 100 ABET-accredited departments offering a degree in a subject to be included as a discipline-specific examination, software engineering could not be included for a long time, if ever, as an afternoon discipline-specific examination. At the time of writing, there are no ABET-accredited schools offering an undergraduate degree in software engineering although at least one school is working towards an accredited program in software engineering. By contrast, there are over 100 accredited undergraduate degree programs in computer engineering (or similar).

Most PEs take the FE examination during their senior year of college or immediately upon graduation and the PE examination after four years of experience. It has been found that it is very difficult to pass the FE examination unless it is taken right before or after graduation from an engineering school. This would exclude those who change to software engineering jobs after working in other areas.

 

In addition, the FE examination requirement for PE licensing is inappropriate for those receiving degrees from computer science departments that are not in Engineering Schools. Even computer science students in engineering schools are rarely trained in all the areas of engineering tested by the FE examination. Studying all those subjects would leave little time for studying computer science and software engineering.

After the practical experience period, an applicant for a PE must pass a second examination, this one on principles and practice of engineering in a discipline-specific topic. The process of deciding what topics should be covered by the examination is long and rigorous. The SWEBOK effort and process would not suffice as it does not satisfy the requirements for a rigorous job analysis and a large and scientific demographic survey of randomly selected practitioners.

The Texas Board of Professional Engineers presented and the NCEES passed a motion that the "Council (i.e., the NCEES) express its intention to add software engineering as a new discipline to Principles and Practice of Engineering (PE) examination status at such time as the provisions of the Examination Policy 7 (EP 7) are met." As of yet, the three provisions of EP 7 have not been met. One key provision is that no discipline shall be added to the examination program unless there is an EAC/ABET-accredited curriculum in the discipline. Though EAC/ABET is in the process of creating such a curriculum, none has been yet accredited.

The Texas motion further states that in "anticipation of the ABET action on a software engineering curriculum, Council staff will be authorized to provide assistance and advice to aid the joint efforts of the Association of Computing Machinery (ACM) and the Institute of Electrical and Electronics Engineers-Computer Society (IEEE-CS) in the development of appropriate national examinations in software engineering." In other words, NCEES staff has been available to ACM and IEEE-CS in an advisory role. Staff has met with IEEE-CS on several occasions and ACM on one occasion to give advice to those organizations while they work through the particulars of the exam development process. While NCEES has expressed the intention to add software engineering as a discipline when the provisions of EP 7 are met, the timeline for EAC/ABET accreditation is uncertain. No NCEES money has been allocated nor spent on this exam development endeavor, nor is the NCEES promoting the development of a software exam in any other way than in an advisory stance. The task force was told that the IEEE Computer Society might be interested in funding the development of this examination (at a cost of approximately $100,000). Note that this examination would be in addition to the general FE exam described above. Without such an examination, software engineers would be required to take an in-depth examination in another engineering field to be licensed as a PE.

2. Degree programs whose graduates seek licensing as a PE require ABET accreditation but few computer science departments are accredited by ABET. ABET does include computer engineering in their accreditation activities. However, relatively few practicing software engineers graduate from computer engineering departments compared to computer science departments, physics departments, and others.
   The Computing Sciences Accreditation Board, Incorporated (CSAB) provides accreditation of computer science programs. At the time of writing, the Computer Science Accreditation

 

Commission (CSAC) of the CSAB and ABET are negotiating an agreement that will lead to the integration of CSAC into ABET, and this might affect the accreditation situation.

3. PEs are licensed in individual states and there are variations among states in requirements, enforcement, etc. Large software efforts often span multiple states and involve hundreds of software engineers. It would be quite impractical to require software engineers to be licensed in each state in which a company for which they work has branches and offices. It would also be inconsistent with the other engineering disciplines where responsibility frequently lies with a single PE within the organization. The other engineers involved in a project are not required to be licensed.

4. The rate of technology change is slow in most fields and the NCEES process is very rigorous and fair. But the typical three year cycle to update a licensing examination will not be practical for many aspects of information technology, where changes are rapid and continual. The examination would most likely be out-of-date most of the time.
   In addition, licenses in most states are good for life. Is a software engineer who got a degree in 1970 and has not updated his or her knowledge since that time still qualified to create safety-critical software? About 15-20 states require some type of continuing education for licensed engineers, but this raises its own problems.

5. The PE examinations are being changed to be multiple choice to ensure consistency of grading. There must be one answer that is correct, and the rest must be demonstrably incorrect. Psychometric issues require this format. It is unlikely that any reasonable test of software engineering skill can be put into this format.

6. The breadth of people who are somehow involved in the production of safety-critical software would make licensing all of them impractical and not particularly helpful. Who would be included: requirements writers? designers? coders? software librarians? QA and testers? reviewers of the software? those who select the computer hardware, operating systems, and provide the system software? managers? Is there a common set of knowledge for all these different jobs?
   Much of current embedded software is created not by software engineers but by other engineers who use graphical programming languages and systems such as MatrixX. Are they practicing software engineering? By the standards of the PE process, they would not be required to be licensed or tested in any field but their major engineering discipline (such as mechanical engineering). What about a person who uses a spreadsheet or configures an SAP system?

**FINDING:**

*Licensing software engineers as PEs would have no or little effect on the safety of the software being produced.*

1. Licenses are usually only required for those engineers (and others such as lawyers, physicians, architects, accountants, etc. and even electricians, many child-care providers, and hairdressers) who provide services directly to the public. Thus consultants and teachers are usually covered but employees who work for companies that produce safety-critical products (and those working for the government) are, in almost all states, exempted from licensing requirements. Therefore, licensing is unlikely to have any affect on safety as virtually all

 

safety-critical systems are built by large teams of people who work for a company. There are few instances where safety-critical software is produced by an individual and sold directly to the public. Most software engineers would not fall under any licensing regulations and would not be required to be licensed nor find much benefit in voluntary licensing.

2. The current PE examination, as is the case for all licensing examinations, is aimed at a pass point determined by "minimal competence". This level of competence is unlikely to affect the quality of safety-critical software. Past minimal competency examinations in computer science areas, such as the DPMA, have been widely ignored and are essentially irrelevant.
3. The typical approach in many large companies is to have a single PE sign off on a complete system. Clearly, for large and complex software systems often written, tested, and reviewed by hundreds of people, one individual could not fully understand the entire system and oversee every part of the process. They must rely on the expertise of others to provide assurance. On the positive side, this does create a line of accountability for critical systems and components. But does licensing provide the enabling mechanism for this process? Are PEs better in any way than other responsible people who do not have a PE license?
4. The PE process licenses everyone as an "engineer"—it is up to the individual to determine whether they are qualified to practice a particular subdiscipline. Therefore, requiring a software engineer who builds or approves safety-critical projects to be a licensed PE is unlikely to solve any problems related to engineers who have limited software expertise.

**Conclusion:**

*The licensing of software engineers as PEs at best would be ignored and at worst would be damaging to our field. It would have no or negligible effect on safety.*

# SOFTWARE BODY OF KNOWLEDGE AND CURRICULUM ACTIVITIES

A preliminary set of findings have been generated about the body of knowledge activities and are included in this draft report because of their importance for the ACM Council decision regarding ACM participation in SWECC and SWEBOK.

1. There is no generally agreed body of knowledge (BoK) for software engineering at this time.
2. The rate of change of the field suggests that there is little chance of developing a documented BoK that would not be obsolete for much of its existence.
3. Safety-critical software and systems have special needs and knowledge that are excluded from the IEEE SWEBOK effort (for example, real-time systems are excluded). Only universally required and applicable knowledge is being included. Thus it will have little relevance for safety-critical systems and may dangerously exclude the most important knowledge related to competence to build these systems or imply that having the knowledge included will make a person qualified to build safety-critical systems.
4. Because little scientific evaluation of software engineering techniques has been done, it will be difficult to get consensus on what is effective and should be included or excluded and the differentiation between knowledge and fad.

 

5. Few textbooks or programs exist that provide adequate instruction on building safety-critical, real-time software. Standard software engineering textbooks and classes do not provide adequate instruction in these special aspects of software engineering and thus do not prepare students to work on such systems. This is particularly true for application software.

# GENERAL CONCLUSIONS AND RECOMMENDATIONS

## General Findings and Conclusions:

With respect to *safety-critical* software:

1. An effective approach to ensuring safety in software-intensive systems will require a systems approach and will not involve simply dealing with the software in isolation.
2. Each industry will need to determine an appropriate mix of approaches that work together to solve their particular problems and fit within the cultural context of that industry. There are no simple and universal fixes that will solve the problem of ensuring public safety. Effective approaches will involve establishing accountability (including corporate and not simply individual accountability), competency within application areas and job responsibilities, liability, regulation where appropriate, standards, and industry-specific requirements.

## Recommendations:

With respect to *safety-critical* software:

1. The ACM should withdraw from efforts to license software engineers as professional engineers.
2. The ACM should take a stand against government efforts to require the licensing of software engineers as impractical, potentially ineffective with respect to safety-critical projects, and potentially detrimental with respect to economic and other societal and technological factors.
3. Textbooks and other educational programs should be developed that focus on software engineering in real-time, safety-critical systems.
4. The ACM should not support the SWEBOK activities but should consider supporting other efforts to validate and codify basic knowledge in various aspects of software engineering.

 