

# LICENSING SOFTWARE ENGINEERS IN CANADA

~ BY DAVID LORGE PARNAS

**SUCH** diverse professions as medicine, law, and engineering are self-regulating. In each of these fields:

- Practitioners deal directly with the general public, most of whom cannot determine whether or not a self-proclaimed expert is qualified to provide the required service.
- Proper practice requires the practitioner understand an organized body of knowledge and have learned how to apply that knowledge through both book knowledge and supervised practice.
- Practitioner errors can cause a great deal of harm to clients or the public.
- There are established codes of practice that must be enforced. The enforcers must be able to stop an unqualified or negligent professional from offering services to the public.



Because regulating these professions requires expertise, many legislatures have created governing bodies whose primary duty is protecting the public. These bodies are not to be confused with advocacy groups, which advance the interests of the professionals, or with technical societies, such as ACM or IEEE, which promulgate scientific knowledge. Professional regulatory authorities exist to restrain practitioners in the interest of the public. The legislation requires them to control a title by examining the qualifications of potential practitioners; licensing all who are qualified to practice; and removing the license of those who do not practice properly.

Like all human institutions, these bodies are far from perfect. It can be difficult to distinguish between poor practices and innovative ones; the line between protection of the public and advancing the interests of the professionals can be fuzzy. The jurisdictions of two professions may appear to overlap. Nonetheless, the overall effect of self-regulation has been positive. When I walk into a doctor's office, I may not get the world's best doctor, but I am very

unlikely to get a completely unqualified and negligent practitioner. I do not have to study my doctor's credentials and quiz her to determine whether or not she took the appropriate courses. The license on the wall, or a call to the medical association, assures me that she is qualified to practice and aware of her legal and ethical obligations.

## Engineering as a Self-Regulating Profession

Legislation to regulate engineering was introduced in response to exploding boilers and collapsing bridges. All Canadian provinces, and most U.S. states, have regulatory bodies charged with making sure that qualified engineers are easy to identify; making sure that (licensed) engineers remain qualified; and regulatory bodies removing the license of engineers who violate laws or codes of practice. They have been granted trademarks and other legal means to prevent people from identifying themselves as engineers unless they have a license.

Advances in science, technology, and mathematics have made it impossible for one person to be qualified in all areas of engineering. Consequently, specialties or disciplines within engineering, such as electrical engineering, have been identified. For each such specialty, there is broad agreement on a core body of knowledge required to practice in that discipline.

For critical applications, such as the construction of bridges and other structures, there is additional legislation, known as demand-side legislation, requiring that design documents bear the seal of a Professional Engineer (P.Eng.). In other cases, an employer or customer is free to employ unlicensed professionals.

Software has become a critical component in medical devices, manufacturing plants, buildings, and aircraft. It has replaced mechanical, electrical, and electronic components in traditional engineering products, and is essential for many services. Moreover, software is used to design critical products; the safety of those products can depend on the correctness of that software. Today, the conditions listed at the beginning apply as much to software development as they do to any other professional activity.

Unfortunately, in spite of four decades of calls for software development to be viewed as engineering, regulation of software development professionals has been neglected by both the engineering authorities and the computer science community.

- The engineering regulators were so accustomed to thinking about physical products they did not recognize that software was replacing other technologies in critical products. Only recently have authorities recognized their legal mandate, and their duty to enhance public safety, required they regulate the use of engineering titles by software developers.
- Computer science was identified as a research area by people who were trained in science and mathematics, not engineering. They seem to have been unaware of the structure of the engineering profession and were unaware of the need to identify a “core body of knowledge” and to establish educational and ethical standards for developers. Computer scientists have treated “software engineering” as an area of research, not as a profession. Most scientific fields are not regulated because scientists rarely provide services directly to the public the way engineers do.
- The public has suffered from this neglect. There are books full of stories about the damage caused by software errors. Delayed and canceled projects are rampant. In most of these cases, the failures can be associated with the failure to apply simple, well-accepted design principles. In many cases, the programmer had never heard of those principles and in other cases, they knew the principles but did not have the discipline to apply them. As long as anyone is free to identify his/herself as a “software engineer,” these problems will remain.

### **The Canadian Dispute**

An internal jurisdictional dispute about “software engineering” at a Newfoundland university developed into a Canada-wide legal dispute. When the computer science department started a “Software Engineering” program without considering the fact that “Engineering” was a restricted title, the Engineering faculty appealed to the regulators; the regulators determined that both students and the public could be misled into believing (incorrectly) that graduates of the program

would be qualified to work as P.Engs and asked that the program be renamed. The right of the CS department to teach what it wanted was never questioned; the only issue was whether a program whose graduates were not qualified to practice engineering should be labeled “Engineering.” A dispute about a name was misrepresented as a battle for academic freedom and a flock of white knights rushed to defend the university against the regulators.

This dispute was fueled by ignorance on both sides. The CS side, apparently unaware of the nature of self-governing professions, chose to argue against any form of regulation rather than work to make sure the licensing criteria were appropriate. With their limited knowledge of computer science, engineers were not in a position to identify the core body of knowledge, but proceeded without the help of scientists who were experts in the field.

Not recognizing how deeply the philosophies differed, a panel proposed a joint accreditation body for software engineering programs and then another committee drafted standards. Predictably, the standards—drawn up by engineers who knew little about software, and computer scientists who opposed strict regulation—pleased nobody. Many computer scientists complained that the standards were too restrictive and the regulatory authorities found them too weak.

The two sides are no closer to understanding each other than they were when the dispute began. The computer science community continues to believe they are defending academic freedom—ignoring the fact that as a professor of engineering, I have as much freedom to speak my mind as they have. The engineering profession still does not recognize they need the help of computer scientists to do the job properly.

### **The Accredited Programs**

In June 2001, software engineering programs at three Canadian universities were accredited by the Canadian Engineering Accreditation Board (CEAB). The graduates of these programs will be eligible for licensing as professional engineers after they gain supervised experience and pass the usual examinations on law and ethics. The three programs differ greatly. Two were developed with the help of computer scientists; one is very close to a computer engineering program. At each of these institutions, students have a choice between unaccred-

---

Instead of a productive technical discussion about licensing criteria, we have had a political battle about the very idea of regulated professions.

ited computer science programs and accredited software engineering programs.

In September 2001, a fourth<sup>1</sup> program was accredited by a body recently created by the Canadian Information Processing Society. This accreditation has no legal implications and graduates will not be qualified to become licensed engineers. Although the standards used by this accreditation body are notably weaker than those of the CEAB, the public will not be aware of the differences and will be misled by the title.

This situation is obviously unsatisfactory. There are now two disjoint sets of "accredited" software engineering programs; some institutions now offer two different programs with the same name. It is difficult to imagine who can benefit from this confusing situation.

The issue of the core body of knowledge for software engineering has not been satisfactorily resolved (see [1] for details).

### Licensing

The Ontario authorities have licensed about 300 engineers who specialize in software. Other provinces have been actively licensing software developers as well. Most of the new licensees are engineers trained in related fields (for example, electrical or computer engineering) who have crossed into software. Few have computer science training.

This situation is also unsatisfactory. Many of those licensed have not mastered the computer science knowledge that I consider essential for software engineering. Until a core body of knowledge for software

engineering is clearly identified and made a prerequisite for licensing, the probability of unqualified licensed practitioners doing harm remains high.

No progress has been made on the issue of demand-side legislation. It is not clear when a software development project must be under the control of licensed software engineers.

### Regrets

The dispute between the computer science community and the engineering regulatory authorities has been destructive. What is needed is a joint effort with the computer scientists helping to identify the knowledge required for licensing and the regulatory authorities applying the legal mechanisms and expertise gained in regulating other engineering disciplines. Instead of a productive technical discussion about licensing criteria, we have had a political battle about the very idea of regulated professions.

I hope that some computer scientists will offer to help the regulatory authorities and that those authorities will recognize they need expertise to do the job properly. At the moment, I see no movement on either side. ■

### REFERENCES

1. Parnas, D.L. Why software developers should be licensed. *Dimensions* 22, 3. Professional Engineers Ontario, (May/June 2001), 36-39. Reprinted in *Engineering Times* 24, 1 National Society of Professional Engineers (Jan. 2002), 6.

---

DAVID LORGE PARNAS (P.ENG) (parnas@mcmaster.ca) is now an SFI Fellow and Director of the Software Quality Research Laboratory at the University of Limerick, Ireland.

---

<sup>1</sup>Strangely, the Web site of this institution identified its program as the "first" accredited software engineering program.

# SOFTWARE ENGINEERING REQUIRES INDIVIDUAL PROFESSIONALISM

~ BY GORD MCCALLA

**HERE,** I argue strongly against licensing for software engineers. I will not be talking in purely philosophical terms. Instead, I will draw concrete lessons from the Canadian experience, where there is a fierce battle under way over the issue of licensing software engineers as professional engi-

neers. This battle has revealed much about licensing and its implications.

First, let me set the Canadian context. Engineering in Canada has for years been a self-governing profession, given exclusive right to practice engineering under provincial and territorial Engineering Acts. Software engineering in Canada (as everywhere in the world) has for years been a central area of computer science. It has *not* been a professional