Software engineering: "Wild West" or maturing discipline?

A look at standards and regulation in software engineering

by Joyce Rowlands, Senior Project Manager

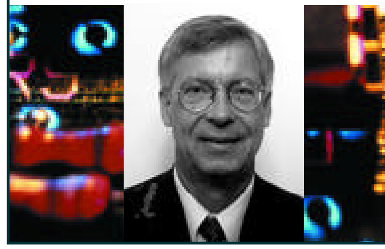
Engineering Dimensions spoke recently to several experts in the software field to obtain their thoughts on issues at the heart of the software debate: Is software engineering real engineering? Does the lack of standards and regulation in the software field pose a risk to public safety? Is there a need for demand-side legislation? Does the professional engineer trained as a software specialist have a unique (even essential) role to play in the design of safety-critical software? Here's what we learned.

“It's the wild west out there.” That's one view on the state of standards, regulation and accountability in the software field. The fact is: There are all kinds of software standards but, in some industries, little agreement on which ones to use. As for personnel, almost anyone can adopt the title “software engineer” (can't they?) with a few months training and the “chutzpah” to peddle oneself as a software specialist. Add superhuman deadlines and staff lock-ups to meet those targets, and standards often go out the window. In the words of one industry-watcher: “When schedules get tight, quality goes.”

At the opposite end of the spectrum, there's no doubt that an organization like Ontario Power Generation (formerly part of Ontario Hydro) spent years scrupulously designing safety standards and test methods for the software used in computerized shutdown systems at its Darlington nuclear generating station. Once the software was installed, tested and judged safe, a massive effort was mounted to document and standardize the development process, test methods and validation techniques, to smooth the process for future retrofits. That done, the organization rewrote and retested its software to fit those specs–all in all, an exhaustive process.

“The real thing you want to test is that [the software] not only does what it's supposed to do, but also that it doesn't do anything untoward or unusual and potentially unsafe. That's a much bigger question.”

Kurt Asmis, PhD, P.Eng., National Contingency Planning Group, Department of National Defence



Largely unregulated

Ontario Power Generation's diligence gives a measure of comfort. But the fact remains that the software field has been largely unregulated, subject only to voluntary standards (for the most part), and populated with all manner of self-taught practitioner. Further, one finds disclaimers instead of warranties on most software products. Yet despite all this, most experts agree that software science is evolving rapidly from the artisan world where individuals ply their personalized craft, to that of mature applied science.

Developments in academe and professional regulation support this contention: A number of universities are now offering (or are in the process of developing) *bona fide* engineering programs specializing in software, leading to eligibility for licensure as a professional engineer (first grads in 2001). And PEO has recently formalized a process to assess the academic and experience qualifications of applicants whose employment experience in software design doesn't correspond with their academic backgrounds, to enable them to become licensed.

Big issue

The big issue in the software field is the need for common practices and techniques in developing and writing software, so others can follow the embedded logic in order to review, test and validate the design. Without standard procedures, it's back to the world of the personal craftsperson, with no assurance that anyone but the originator can retrace the steps to fix bugs, or test safety features.

Kurt Asmis, PhD, P.Eng., now on assignment with the National Contingency Planning Group, Department of National Defence (year 2000 readiness team), is credited with recognizing the need to review Darlington's shutdown system software back in the late '80s. As director of the Atomic Energy Control Board's safety evaluation division at that time, he headed the Darlington review for the federal regulator.

Asmis believes software development for safety-critical applications should employ the rigour and discipline used in any other branch of engineering. “The same rules should apply as for a major bridge or electrical system. That means sign-off, checks and independent reviews. The real thing you want to test is that [the software] not only does what it's supposed to do, but also that it doesn't do anything untoward or unusual and potentially unsafe. That's a much bigger question.”

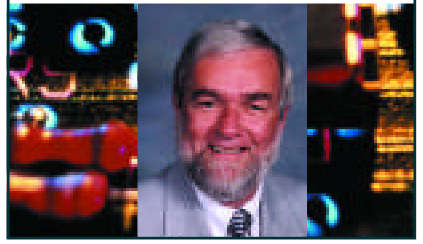
Essential ingredient

Professional engineers specializing in software design bring to the process something that other software practitioners lack: an understanding of the physical system or product impacted by the software. This, along with an appreciation of traditional engineering principles, is an ingredient some feel is essential in developing software for safety-critical systems. The ability to communicate readily with project engineers gives added value. “They're all involved in the safety chain, and they have to be able to communicate,” notes Asmis.

He argues that P.Eng. software specialists add a further dimension to safety. That's because software tends to be developed by one individual or group, who often work in relative isolation from other members of the project team. “Unlike other engineering fields, you don't hand the baton on to someone in a totally different trade, who adds value to the safety of a design with their own set of eyes and experience.”

“Now we're getting sophisticated enough that we can talk about the engineering of software, as opposed to throwing a bunch of hackers in a room and hoping that on Monday morning something comes out.”

Michael Bennett, PhD, associate chair, software engineering program, University of Western Ontario



Eyes opened

It seems the Darlington experience opened a lot of people's eyes to the need for standardized methods and review techniques in safety-critical software. David Parnas, PhD, P.Eng., director of McMaster University's software engineering program, and NSERC/Bell Industrial Research Chair in Software Engineering, was the outside expert brought in by Asmis to review Darlington's software back in 1986. It was his thumbs-down verdict that triggered the subsequent massive review project.

Today, Parnas is a vocal advocate for standards and professionalism in software design. In his mind there's no question that writing safety-critical software (or “software critical to safety,” his preferred term) is *real* engineering, requiring all the rigour of traditional disciplines.

He tells the story of his visit to the plant where the pacemaker embedded in his own chest was manufactured. “The company takes great pride in its design process. They showed us all the inspections they'd done, all the signatures on all the drawings and all the reviews,” he says. But when Parnas asked about their review of the pacemaker's software, he drew blank stares: “‘Do you have to review software,’ they asked? They didn't have any of the design documentation, they didn't have any of the things they had for any of the other engineering products–and they were very careful, very disciplined engineers.”

That simply isn't good enough as far as Parnas is concerned. “There's a whole body of knowledge on how to structure programs, so each part can be reviewed separately,” he says. “It all needs to be reviewed, tested and inspected, but most people don't know how to do those things. You have to follow a disciplined process, and reviews have to be part of it.”

Sparring professor

Parnas has also been known to spar with fellow professors (engineering colleagues), who insist that any engineer who writes software is *de facto* a software engineer. “That's like saying anyone with a degree in electrical engineering can do civil engineering or build a bridge,” he believes. He feels strongly that software engineering requires additional knowledge–a grounding in discreet mathematics, for instance. “I don't care whether you acquire it for-

“Software engineers need to know a lot of things other engineers need to know. If you want to shut down a nuclear plant, for example, you need someone who understands both the software and the dynamics.”

David Parnas, PhD, P.Eng.

mally or informally, but you need it," he insists.

Parnas is emphatic too in his conviction that, in certain situations, an engineering background, in addition to formal, software education, is essential to safety. "Software engineers need to know a lot of things other engineers need to know. If you want to shut down a nuclear plant, for example, you need someone who understands both the software and the dynamics," he says. "Computer scientists look at our program [at McMaster], see we have a course in thermodynamics and heat flow, and ask 'what's that got to do with software?'"

"There's a lot of software written for process control and critical applications. It might help if there were a single governance for safety applications."

Eric Samuel, P.Eng., systems design engineer and bid manager, CrossKeys Systems Corporation

Parnas supports PEO's recent steps to define the licensing process for software engineers, and sits on the task force that identified core knowledge requirements. "Software needs to be designed by qualified people," he says. "We need some sort of qualification, and there's no other organization I know of that assesses people's technical background. You need a licence to be a hairdresser, but not to be a programmer."

Bad software

He doesn't hesitate to point out the potential risk involved in having unqualified people writing software for safety-critical applications. According to Parnas and other experts, there are plenty of examples of failures that have resulted in death and destruction, though the finger of blame rarely points to the software. That's because, when a disaster occurs, invariably, more than one system fails—that being the nature of "failsafe" back-up systems.

"[In many industries], everything to do with software seems to be voluntary. And I don't see any regulator in the medical products area saying, 'we're going to inspect your software.'"

David Parnas, PhD, P.Eng., director, software engineering program, McMaster University

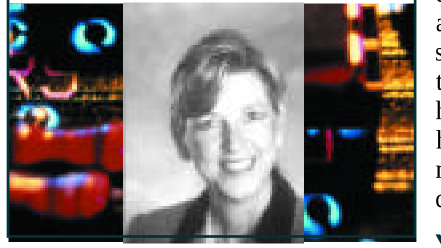
One well-known example in which software failure is documented is the radiotherapy machine that killed several patients by delivering heavy overdoses of radiation. In that case, Parnas was told, the person who wrote the software had trouble communicating with his fellow engineers: "His command of English and French was equally bad, so they put him on the software. They thought that was the place where he could do no harm, despite the fact that he had no education in software."

Industry standards

That attitude is changing, with broad recognition in the academic and professional engineering communities that specialized training, as well as standards and regulation are needed. But where does industry stand? The former Ontario Hydro (and its corporate progeny) and Atomic Energy of Canada Ltd. are the only industries Parnas knows of in which strict standards and inspection criteria for software development have been imposed by an outside regulator. In many industries, he says, "everything to do with software seems to be voluntary. And I don't see any regulator in the medical products area saying, 'we're going to inspect your software.' I've had phone calls from people asking 'what do we do?' but they don't follow through and actually do it."

"I think we have to tread very, very carefully and be very clear about what we want to regulate."

Claudette MacKay-Lassonde, PhD, P.Eng., chairman, Enghouse Systems Limited



Eric Samuel, P.Eng., is a systems design engineer and bid manager for CrossKeys Systems Corporation of Kanata, Ontario, a Newbridge-affiliated company that produces software for the telecommunications industry. A software designer himself, Samuel managed people doing software development for several years. CrossKeys doesn't write safety-critical software, but some segments of the industry do, and Samuel believes they should be regulated. "For some communications systems, you'd definitely want to use engineering safety principles," he says. "There's a lot of software written for process control and critical applications. It might help if there were a single governance for safety applications."

Qualified practitioners scarce

That said, there's a serious shortage of qualified software practitioners, and an even greater scarcity of licensed professional engineers trained in software development. So it's hard to pick and choose. But Samuel sees value in the professional ethic and discipline a P.Eng. brings to the task. "In my own case, having gone through the training and discipline, there have been times when it's helped me make the right choices–the correct professional decisions."

One who concurs with this sentiment is Michael Bennett, PhD, associate chair; software engineering, Department of Electrical and Computer Engineering, University of Western Ontario. The irony is that Dr. Bennett is not a P.Eng.—but soon hopes to be: "For 40 years I was in the software wilderness [unable to get licensed]. Now my application is on my desk, and I hope to send it off today."

Until recently, software engineering was not mature enough to qualify for licensed status, Dr. Bennett contends. "The problem with any engineering field is that you need a process in place so you can say, 'this is how we do this kind of engineering,'" he says. "Until five years ago, we really couldn't say that in software. But now we're getting sophisticated enough that we can talk about the engineering of software, as opposed to throwing a bunch of hackers in a room and hoping that on Monday morning something comes out."

Y2K lessons

For Bennett, the need for standards and regulation in the software field is underlined by the Y2K crisis. "There are some real safety issues out there that people aren't aware of," he believes. "One of the great benefits of Y2K is that it has brought to the ordinary person on the street what can happen if you don't engineer software properly–it can cause all kinds of catastrophe." He believes it's only a matter of time before people start to demand legislation to protect against inadequate software, "and we're going to see people suing companies if software fails."

Bennett believes there's value in the higher order accountability a licensed professional is held to–accountability that goes beyond responding to the demands of one's employer or project manager: "There's enormous value in that. It means you can stand up and say, 'as professionals, we can't do that.' Otherwise you're completely susceptible to the whims of the employer."

Tread carefully

Another view from industry comes from Claudette MacKay-Lassonde, PhD, P.Eng., chairman, Enghouse Systems Limited, and a former PEO president. She believes there's a need for control and regulation in areas that have direct safety impacts, but she urges a cautious approach "I think we have to tread very, very carefully and be very clear about what we want to regulate," she says. Otherwise, the profession will lose credibility, she believes, because it won't be able to enforce its position.

P.Eng. sign-off

Paul Joannou, M.Eng., P.Eng., is a 21-year veteran of Ontario Hydro, now manager of engineering standards for Ontario Power Generation's nuclear business, and formerly supervisor of software engineering standards and methods. OPG is currently updating all of its standards and procedures for engineering, and Joannou is at the centre of it. "One of the things we're looking at is making sure there are adequate requirements for P.Eng. sign-off, where necessary," he says.

Finding the right people to do the job is the problem, says Joannou. "When safety-critical software is being developed, there's a need to apply specialized techniques that are not common knowledge," he explains. "Not applying proper techniques can have disastrous results–clearly there's a public safety issue. I think a software engineering background, as I've seen it defined, is a necessary component. There are definitely people pushing [for standards and regulation] in different areas. Things are happening legislatively, in standards committees, academically, in professional engineering associations; so it's all heading in the right direction. This is all part of the lifecycle of a new discipline." ◆

"Not applying the proper techniques can have disastrous results–clearly there's a public safety issue. I think a software engineering background, as I've seen it defined, is a necessary component."

Paul Joannou, P.Eng., manager of engineering standards, Ontario Power Generation

Miles to go...

PEO President-elect Peter Devita, P.Eng.

Defining a core body of knowledge for software engineering and licensing software practitioners are important and necessary steps in closing the regulatory gap between traditional engineering disciplines and today's high-tech world. But PEO still has a long way to go in coming to grips with new and emerging technologies.

That's the view of Peter DeVita, P.Eng., PEO's President-elect, a computer engineer whose firm specializes in building customized computers for unique applications. During his term as president, DeVita intends to highlight the need for regulation of emerging engineering disciplines–and he'll have a lot to say on enforcement.

"If you're going to issue a licence that has real meaning, you have to define an exclusive scope of practice–and PEO must drive that," stresses DeVita. "We do this through a series of initiatives that includes direct enforcement–by that I mean [law]suits by PEO, lobbying for protective demand-side legislation, influencing government and industry to use engineers properly, and informing and educating the public on the role of engineering in our society. We have an [engineering] culture that evolved from a group who had an exclusive scope of practice. But we forgot about the fact that technology keeps on growing.

"If you're going to issue a licence that has real meaning, you have to define an exclusive scope of practice–and PEO must drive that," stresses DeVita.

DeVita explains that civil engineers, who have played a major role in PEO's history since its formation, have had an exclusive scope of practice since 1937. "If you look at mechanical engineering, those doing the standard stuff–pressure vessels and so on–they're covered," he says. "But when you get into the sophisticated stuff–robotics, etc.–these people aren't covered. Look at electrical and you see the same thing. Power is covered, but not the newer stuff." DeVita notes that, currently, anyone can do the work that he does–a fact he sometimes has to point out to make the case that emerging areas aren't being regulated effectively.

DeVita has equally strong views on the underutilization of engineers in this country, and our dangerous drift toward a branch plant economy .

Stay tuned...