November 18, 1994

Professor Nancy Leveson
Computer Science and
Engineering Department FR-35
University of Washington
Seattle, WA 98195

Dear Professor Leveson:

I very much enjoyed your article "High Pressure Steam Engines and Computer Software." After teaching "Bursting Boilers and the Federal Power" for a number of years in my Ethics in Engineering course and editing Cameron and Millard's "Technology Assessment: A Historical Approach, " I was, of course, very attuned to your approach.

Beyond that, with my colleague Ilene Burnstein who teaches software engineering in our Computer Science Department, and other colleagues, I have begun to formulate a collaborative research project on software engineering, professionalism, and accountability. The aim would be to derive guidelines for teaching as well as practice.

I have enclosed a draft of a concept paper for the project. In addition to Ilene Burnstein the project group would include Helen Nissenbaum (Princeton University), Michael Loui (University of Illinois), my colleague Michael Davis, and a Chicago lawyer, Barry Weiss. Each of them told me about you. We expect to add Donald Gotterbarn or another person at work on these issues in the professional societies, and perhaps someone from Carnegie Mellon. Perhaps the project will spark your interest.

I hope you will send criticisms and comments on the draft.

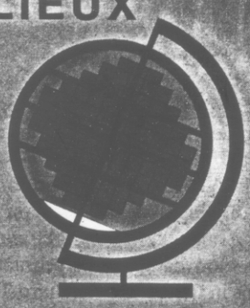
Sincerely,

Vivian Weil
Director

VW:rm

enc.

COMPUTING MILIEUX



High-Pressure Steam Engines and Computer Software

Nancy G. Leveson, University of Washington

How can we get the benefits of computers without increasing risk? Studying parallels in the early development of high-pressure steam engines and of software engineering can help.

Even though a scientific explanation may appear to be a model of rational order, we should not infer from that order that the genesis of the explanation was itself orderly. Science is only orderly after the fact; in process, and especially at the advancing edge of some field, it is chaotic and fiercely controversial.

— William Ruckelshaus[1]

The introduction of computers into the control of potentially dangerous devices has led to a growing awareness of the possible contribution of software to serious accidents. The number of computer-related accidents so far has been small due to the restraint shown in introducing computers into safety-critical control loops. However, as the economic and technological benefits of using computers become more widely accepted, their use is increasing dramatically. We need to ensure that computers are introduced into safety-critical systems in the most responsible way possible and at a speed that does not expose people to undue risk.

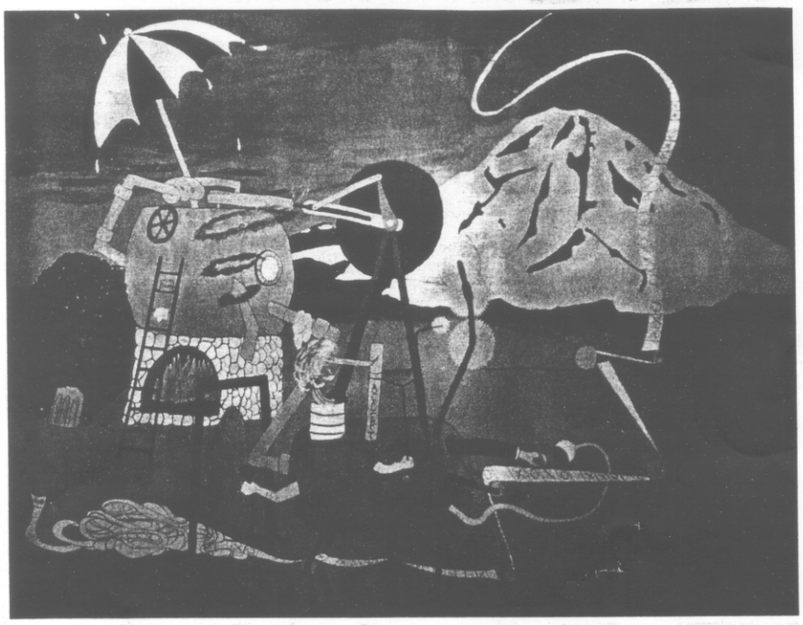
Risk induced by technological innovation existed long before computers; this is not the first time that humans have come up with an extremely useful new technology that is potentially dangerous. We can learn from the past before we repeat the same mistakes.

Problems of exploding boilers

Parallels exist between the early development of high-pressure steam engines and software engineering that we can apply to the use of computers in complex systems.

> Great inventions are never, and great discoveries are seldom, the work of any one mind. Every great invention is really either an aggregation of minor inventions, or the final step of a progression . . . Frequently an important invention is made before the world is ready to receive it, and the unhappy inventor is taught, by his failure, that it is as unfortunate to be in advance of his age as to be behind it. Inventions only become successful when they are not only needed, but when mankind is so advanced in intelligence as to appreciate and to express the necessity for them, and to at once make use of them.
>
> — Robert H. Thurston, *A History of the Growth of the Steam Engine* (1883)

0018-9162/ $4.00 © 1994 IEEE



Steam-powered Turing machine, a mural in the Department of Computer Science at the University of Washington.

Hero of Alexandria, who lived around 60 AD, conducted some of the first known investigations into the use of steam for power. But it was not until the 16th and 17th centuries that the problem of pumping water out of mines changed the search for steam power from a diversion to a necessity. Many inventors attempted to harness this source of power, but Savery is usually credited as the first to produce and sell a workable steam apparatus. Newcomen later designed a practical cylinder and piston engine around the year 1700 that was the forerunner of all subsequent steam engines.

Watt's contribution. In 1786, James Watt was working as an instrument maker at Glasgow University and was asked to repair a model of a Newcomen engine that was being used in a natural philosophy class. By one of those serendipitous coincidences of history, Watt had become friendly with several professors, including chemistry professor Joseph Black, who discussed with Watt his recent discovery of the phenomenon of latent heat. Watt was unique among the early steam engine inventors in having had direct and indirect contact with scientists who studied heat.[2]

Watt decided he could improve on the Newcomen engine and patented several important ideas, including the separate condenser and the design of an engine producing rotating motion, at the same time as the industrial revolution was generating a demand for power on an unprecedented scale. With a successful manufacturer named Matthew Boulton, Watt came up with a design for a steam engine that was the leading edge of technological change in the last two decades of the 18th century. The application of steam power transformed industry in terms of output and productivity and produced even more revolutionary changes in transportation when it was applied to locomotives and ships.

The Boulton and Watt machines used low-pressure steam (from 5 to 15 pounds per square inch), which limited both their efficiency and economy. Higher pressure (that is, above atmospheric pressure) would have permitted more powerful and economical engines, but Watt opposed it on the grounds that it increased the danger of explosion and constituted an unacceptable risk.

Although Watt and Boulton resisted making high-pressure steam engines, their patent expired in 1800, and such engines soon made their appearance. Oliver Evans in the US and Richard Trevithick in England almost simultaneously designed engines that dispensed with condensers and used steam directly to push a piston. These so-called high-pressure engines required greater than atmospheric pressure to work.

The first widespread application of the high-pressure engine — on steamboats — resulted in frequent and disastrous explosions: Passengers and crew were blown up, scalded to death, hit by flying fragments of iron, and blown off steamers to drown. Accidents were also common in industrial uses of high-pressure steam. The early steam engines used inferior materials and had low standards of workmanship; the mechanics lacked proper training and skills; and there were serious problems with quality control.[3]

In the US, there were calls for professionalization and standardization of the training of steam engineers, who typically had an informal and haphazard education. There was even a suggestion that the federal government establish an academy of steam technology. All of this came to naught, and engineers continued to be trained haphazardly for many years.

Watt's predictions about the danger of the new engine were correct. Cameron and Millard[3] wrote:

> As the technology of steam power advanced, Watt found himself in an increasingly difficult dilemma: The trend toward greater efficiency and power also increased the risk of explosion. The technology that he had created escaped his control and became increasingly dangerous to life and property. Watt expected more accidents and deaths would result from adoption of high-pressure steam. The threat to public safety now overshadowed the public utility of steam power. . .
>
> But what could Boulton and Watt do? They were in no position to stem the economic forces that demanded more and more power from the steam engine. If they refused to develop the technology, many other engineers — most of them untrained and poorly skilled — were willing to take

the risk of high-pressure steam. What they could do was to alert the public to dangers in the new technology and remind their fellow engineers of their special obligations to ensure public safety. Watt initiated the debate about the risks of the new technology and used his influence to press for safer, and better engineered, alternatives.

Watt's campaign against high-pressure steam along with some well-publicized accidents slowed its adoption in England. Trevithick[2] complained that his competitors had greatly exaggerated the risk and the accidents, writing:

> I believe that Mr. B. and Mr. Watt is abt to do mee every engurey in their power for the have don their outemost to repoart the exploseion both in the newspapers and private letters very differnt to what it really is.

A German supporter of high-pressure steam wrote in 1842 that the intense discussion of its defects and safety risks had clouded the issue of its advantages and had "disgusted the industrial community."[3]

Safety features. Public pressure did force the makers of high-pressure steam engines to incorporate safety features.[4] The risk from this type of machine came from the boiler and not from the engine itself: It was the boiler that was exploding and causing most of the casualties. The technological development of boilers lagged behind the rapid improvement of the engines. Engineers quickly amassed scientific information about thermodynamics, the action of steam in the cylinder, the strength of materials in the engine, and many other aspects of steam engine operation. But there was little scientific understanding about the buildup of steam pressure in the boiler, the effect of corrosion and decay, and the causes of boiler explosions.[2] High-pressure steam had made the current boiler design obsolete by producing excessive strain on the boilers and exposing weaknesses in the materials and the construction of the boilers.

To counter this, engineers introduced two types of safety features: safety valves to reduce steam pressure when it reached a dangerous level and fusible lead plugs that were supposed to melt when the temperature in the boiler grew too hot because of the overheating of the steam. But these much publicized technological fixes did not solve the problems, and the number of explosions continued to increase. The fixes were unsuccessful because engineers did not fully understand what went on in steam boilers: It was not until well after the mid-century that the dynamics of steam generation was understood.

A second reason for the number of accidents was that engineers had badly miscalculated the working environment of steam engines and the quality of the operators and maintainers. Most designs for engines and safety features were based on the assumption that owners and operators would behave rationally, conscientiously, and capably. But operators and maintainers were poorly trained, and economic incentives existed to override the safety devices in order to get more work done. Owners and operators had little understanding of the workings of the engine and the limits of its operation.

While operators certainly did contribute to the problems, they were not solely responsible for them. Nevertheless, owners or operators received most of the blame for explosions. Criticism was rarely leveled at the engineer who had designed the engine. As noted above, many of the engineers who took the risk of developing high-pressure steam technology were untrained and poorly skilled. Limited knowledge of the scientific foundations of their craft existed at the time. The personal standards of the inventor-engineer were the chief element in the safe operation of the engine, and Watt believed that engineers had a personal responsibility to ensure a safe and efficient steam engine and that they bore culpability in case of accidents.

Early opponents of high-pressure steam proposed regulations to limit its dangers by limiting the uses of the new technology. This idea met with little success. In the first half of the 19th century, governments were not disposed to interfere with private enterprise. The steam engine embodied the idea of success and was credited with "national progress almost unchecked, and of prosperity and happiness increased beyond all precedent."[3] Many engineers argued that the social and economic gains of steam power were an acceptable trade-off for the risk involved. Typical was the response of US Senator Thomas Hart Benton. Upon helping defeat legislation to reduce boiler explosions on steamboats, Benton remarked that masters and owners of steamboats were, with few exceptions, men of the highest integrity and that *he* had never met with any accident on a steamboat despite the fact that he traveled widely. When boarding a steamboat, he said he was always careful to inquire whether the machinery was in good order.[5]

But the dramatic increase in accidents that followed wide-scale introduction of steam engines was hard to ignore. The explosion of a steam-powered boat in England, followed by a series of industrial explosions, led to the creation of a select committee in 1817 to report on the dangers of high-pressure steam. The committee began its report by acknowledging the great contributions of steam power to national prosperity and the drawbacks to interfering with private business. However, it noted that when public safety was endangered by "ignorance, avarice, or inattention . . . it becomes the duty of Parliament to interpose."[5] The committee recommended frequent boiler inspections, but its recommendations were not put into effect. About the same time, the city council of Philadelphia was the first legislative body in the US to take notice of the accidents and attempt to investigate. A city council report was referred to the state legislature, where it died.

Accidents continued at an alarming rate during the 1830s and 1840s, prompting more government attempts to limit risk. In the US, the commissioner of patents estimated that from 1816 to 1848, a total of 233 steamboat explosions occurred, killing 2,562 persons, injuring 2,097, and causing more than $3 million in property losses. The Franklin Institute, which was founded in Philadelphia in 1824 to study and promote the "mechanical arts and applied science," began a six-year study of boiler explosions. The first research grant of a technological nature by the US government went to the insti-

tute to defray the cost of the apparatus required for experiments in this study. In this instance, an invention and the accidents associated with it were pushing science. The result was

(1) a series of reports that exposed errors and myths in popular theories on the nature of steam and the causes of explosions,
(2) guidelines for the design and construction of boilers, and
(3) a recommendation that Congress enact regulatory legislation, including requirements that engineers meet certain standards of experience, knowledge, and character.[5]

Attitudes begin to change. As a result of steamboat explosions, the prevailing bias against government regulation began to change. Laws were passed in England and the US requiring compensation for families of passengers killed in accidents due to neglect or default. However, no inspection criteria were included, nor were qualifications set for engineers. The prevailing belief was that putting qualifications for engineers into effect was too difficult and that enlightened self-interest of entrepreneurs would guarantee the public safety. These laws failed to reduce the number of explosions.

Hundreds of newspaper editorials on the subject expressed the increased frustration of the public. The social costs of high-pressure steam engines versus the economic benefits were even treated in literature. Charles Dickens wrote about them in *Household Words*, and in the novel *Gryll Grange* by Thomas Love Peacock, a character remarks that "High-pressure steam would not scatter death and destruction around them if the dishonesty of avarice did not tempt their employment, where the more costly low-pressure engine would ensure absolute safety."

Public pressure plus a series of marine disasters killing hundreds more people finally forced Congress to pass a law in 1852 that corrected the problems with steamboat boilers and reduced the number of steamboat accidents. This law was the first successful example of regulatory legislation in the US, and it created the first US agency to regulate private enterprise.[5] Unfortunately, similar legislation was not passed for locomotive and stationary boilers, and accidents involving the use of boilers in other than steamboats continued.

Watt and others were correct in their belief that new standards of precision and safety were essential in the design, manufacture, and operation of the engines. These high standards were finally enforced in Britain in the latter part of the 19th century, and boiler explosions in Britain fell dramatically. In 1905, Britain had only 14 deaths from boiler explosions, compared to 383 in the US. Eventually, a majority of Americans also realized the necessity of enforcing standards: Associations for the prevention of steam boiler explosions were formed; insurance companies were organized to ensure steam equipment that was manufactured and operated with the utmost regard for safety; and, through the efforts of the American Society of Mechanical Engineers, uniform boiler codes were adopted.[5]

The social costs of high-pressure steam engines versus the economic benefits were even treated in literature.

Exploding software?

We are now in the computer age, and we are again faced with a new technology for which there are great economic incentives to push the state of the art and to use this technology to control dangerous systems. Computers, like steam engines and electrical systems, give us the ability to accomplish things we could not accomplish before. And again, it appears that the risks could increase over time as computers take over more and more functions. One difference is the potential consequences of accidents: We are building systems — and using computers to control them — that have the potential for large-scale destruction of life and the environment. Even a few accidents may be disastrous.

It is therefore crucial that we use computers responsibly. Examining more closely the parallels from the past may provide some clues as to how to do this.

Parallel 1 — *The risk from steam engines came from the boiler and not from the engine itself; boiler technology lagged behind improvement in steam engines. Yet economic forces demanded more and more power from the steam engine. Many engineers argued that the social and economic gains of steam power were an acceptable trade-off for the risk involved.*

The use of digital computers allows us to build complex systems that otherwise would be impossible and provides the potential of great economic gain. Although computer hardware technology has advanced at an astounding rate, the development of software engineering has been slower. It has also been slower than required for the complex systems we want to build, like a space station or automatically controlled nuclear power plants. There appear to be two ways to cope with this shortfall.

The first is to fall back on a time-tested engineering principle: Keep things simple and slowly increase the complexity of what we are attempting to do as we learn from our experiences. For example, Ontario Hydro recently became the first utility in Canada to obtain a license for a completely computerized nuclear power plant shutdown system. The software contains about 6,000 lines of code and uses only the simplest, most straightforward coding techniques. Hardware fail-safe devices like watchdog timers and software self-checks are included to deal with some types of software errors. The software includes well-established safety design principles that were standard in the previous hardware shutdown systems. And because the software design is so simple, Ontario Hydro was able to apply formal and informal verification and safety techniques,[6,7] in addition to using standard testing techniques to develop confidence in the software.

In contrast, the first computerized shutdown system in England, under licensing evaluation for the Sizewell B reactor, has 100,000 lines of code, involves 300-400 microprocessors, and contains both control and shutdown functions.[8] This system is beyond our ability to apply sophisticated software verification techniques, is obviously more complex than necessary, and violates the basic nuclear reactor safety design principle that requires complete independence of control and safety devices.[9] Safety design criteria of this type represent knowledge accumulated by successes and failures in engineering over hundreds of years — computer scientists need to be aware of them, and engineers should think carefully before abandoning them.

A second way to cope with the gap between software and hardware technology development also requires us to dampen somewhat our enthusiasm and confidence in computers. Although mistrust of computers has led to the use of hardware backup and fail-safe devices in the most critical systems, this mistrust is fading. Increasingly, existing hardware safety mechanisms and interlocks are being eliminated, and computers are being substituted for monitoring and control. Engineers are deciding that the hardware safety interlocks and backups are not worth the expense or, in the case of aircraft, the extra weight; or they put more faith in software than in hardware reliability. This again violates a standard safety design principle that requires eliminating single-point failure modes, that is, the system should be built so that a single event (like a software error) cannot cause an accident.

The Therac-25 is an apt example. The designers of this radiation therapy machine eliminated the standard hardware safety interlocks for linear accelerators of this type when they introduced computer control, believing that the hardware devices were no longer necessary. Instead, the interlocks and safety checks were implemented in software. After seven accidents between 1985 and 1987 involving massive radiation overdoses and four deaths, the company that built the Therac-25s relented and put hardware safety devices on the machines.[10]

We can be cautious and impose self-discipline in our use of computers to control dangerous systems without unduly hampering technological progress. Watt campaigned against the use of high-pressure steam engines, yet he was only successful in somewhat delaying their use in Britain. In the 1880s, at the same time the industrial world was struggling to cope with the rapid introduction of steam technology, similar issues arose with the introduction of high-voltage electricity. Another inventor, Thomas Edison, criticized the use of high voltage because of its complexity, poor reliability, and threat to public safety and began a campaign to alert the public of the dangers and of his belief that the size and impact of the risk would increase over time. Edison argued for a safe low-voltage electrical system that could quickly achieve public acceptance. Like Watt, he was only partially successful.

Another inventor-engineer, Elihu Thomson, also opposed high-voltage current as too dangerous. But instead of condemning the system and campaigning for its elimination, Thomson attempted to find a technological fix. He believed that several safety devices would greatly reduce the risk of accidents and lobbied for the need to engineer safe high-voltage systems. Thomson's argument was that a program of safety engineering would have commercial advantages in a highly competitive market for those companies with a technological lead in the construction of the safety devices.

Watt and Edison attempted to limit risk by arguing against the introduction of technology with tremendous potential benefits. In contrast, Thomson argued that we can limit risk by using simple, safe designs rather than limiting the uses of our technology or drastically inhibiting technological development. The Thomson approach is the more practical and more likely to be successfully applied to the use of computers in safety-critical systems.

Parallel 2 — *Engineers quickly amassed scientific information about many aspects of steam engine operation, but there was little scientific understanding about the buildup of steam pressure in the boiler, the effect of corrosion and decay, and the causes of boiler explosions. High-pressure steam made the current boiler design obsolete by producing excessive strain on the boilers and exposing weaknesses in the materials and construction of the boilers.*

Like boilers, the scientific foundations of our field are still being developed. Changing from an art to a science requires accumulating and classifying knowledge. Although knowledge is being acquired, more effort is being expended on new inventions and on building tools for unproven techniques without rigorous scientific foundations. We need to carefully validate and assess our hypotheses using scientific principles.

Trial and error is a time-tested way of accumulating engineering knowledge. Engineers analyze the causes of failures and accidents and then take corrective measures to prevent or minimize their recurrence. The corrections eventually find their way into specifications, standards, codes, regulatory requirements, and what is considered to be good engineering practice. But this is a very slow way to accumulate knowledge.

Early in the trial-and-error process, engineers start to look for analytical approaches. The brisk pace of technological development today is possible because of the foundational knowledge that has been developed about such things as mechanics, materials, and structures: Engineers no longer have to evaluate their designs by building something and seeing whether it falls down over time.

There are two stages in the early years of a new technology:

(1) exploration of the space of possible approaches and solutions to problems (that is, invention) and
(2) evaluation of what has been learned by this trial-and-error process to formulate hypotheses that can be scientifically and empirically tested in order to build the scientific foundations of the technology.

Most of our emphasis so far has been in the first stage or invention; it is time now to give more attention to the second.

Invention is a worthy and necessary pursuit, but the most useful inventions are based on, or improved by, scientific knowledge. Invention produces products, techniques, and tools. Science produces the knowledge and ability to evaluate and

improve our products, techniques, and tools. Inventors use science to build better inventions, to know that they are better, and to compare them to what we already have. Although inventors had built steam engines for centuries, Watt was unique in having direct contact with scientists who were studying heat. The gradual development of scientific knowledge led to the important patents by Watt that produced a practical steam engine. Further enhancement of basic knowledge about steam engines and boilers allowed the production of more effective and safer engines. Although rudimentary knowledge allowed the production and use of low-pressure steam engines, safe high-pressure engines required a deeper scientific foundation.

Software engineering inventions have provided leverage in building our current software systems. I do not want to denigrate what we have accomplished: We are building extremely complex systems, many of which work remarkably well a great deal of the time. But we may be straining at the limits of what we can do effectively without better inventions based on known scientific and engineering principles. And our early rapid progress may be slowing as we reach the limits of what we can accomplish with brute force. As an example, the late 1950s and early 1960s saw the development of very clever ways of building parsers for programming languages. But with the development of formal theories of grammars, parser generators that eliminated the necessity of crafting a parser for each new compiler became possible.

Similar needs exist in software engineering. Our greatest need now, in terms of future progress rather than short-term coping with current software engineering projects, is not for new languages or tools to implement our inventions, but for more in-depth understanding of whether our inventions are effective and why or why not. For example, we have a greater need to develop and validate the underlying principles and criteria for designing specification languages than to create more languages. We have a greater need to develop and validate basic design principles and to understand conflicts and trade-offs between them than for more tools to specify designs. And we have a greater need to study the effects of different types of software development processes in real organizations and under different conditions than to create more languages for specifying processes.

Reseachers in some subfields of software engineering have been more conscientious in attempting to build their theoretical foundations. Testing is one such area, although it also has a long way to go. For example, testing researchers have defined theoretical ways of comparing testing strategies both in terms of cost and effectiveness, formal criteria for evaluating testing strategies, and axioms or properties that any adequacy criterion (rule to determine when testing can stop) should satisfy. In general, theoretical foundations can provide

(1) criteria for evaluation,
(2) means of comparison,
(3) theoretical limits and capabilities,
(4) means of prediction, and
(5) underlying rules, principles, and structure.

Our greatest need now is for more in-depth understanding of whether our inventions are effective and why or why not.

Like the exploding boilers, our ability to build safe software-controlled systems and to build effective software engineering tools to accomplish this will be enhanced by greater understanding of the scientific foundations of our craft.

Parallel 3 — *The safety features designed for the boilers did not work as well as predicted because they were not based on scientific understanding of the causes of accidents.*

Building a scientific foundation of software engineering requires both building mathematical models and theories and performing carefully designed experiments. In an abstract system, the elements are created by definitions and the relationships between them are created by assumptions (for instance, axioms and postulates). Many questions can be answered about abstract systems by using mathematics. In concrete systems (where some of the components are physical objects), establishment of the existence and properties of elements requires research with an empirical foundation, since our knowledge of the physical laws involved is almost always incomplete.

The great power of the computer is that it is a general-purpose machine that can be changed into a special-purpose machine by the addition of a set of instructions (data) to accomplish that purpose. Software is an abstract design of a special-purpose machine that becomes a concrete design as soon as it is executed on a computer. Therefore, software can be — and should be — evaluated both as an abstract design and a concrete design. Furthermore, software is both a mathematical object and a human product. We cannot build effective tools or design techniques to help humans construct software without understanding the human problem-solving behavior involved in building software.

The empirical aspects of our field imply the necessity for experimentation. As an example, formal methods have been proposed as a partial solution for the problems of ensuring safety, but there has been little validation of the hypotheses underlying these techniques. Does the use of formal methods result in fewer or different errors being made? Are the resulting programs more reliable? Are they safer? Are some techniques more effective than others? What type of training is necessary to use the techniques effectively? Is it more or less costly to use formal methods? Because the techniques must be employed by humans, it is not possible to answer these questions using only mathematical analysis; experiments involving humans will be necessary.

Intuition plays an important role in for-

mulating hypotheses. But sometimes our intuition is misleading; we cannot stop with generating hypotheses (as we too often do now) no matter how much confidence our intuition allows us to place in them. Currently, we are applying techniques and even mandating them without validating that these work or that the underlying hypotheses and assumptions are valid.

When a physicist makes an erroneous claim, such as in cold fusion, the idea may stay around for a while on the fringes of the field. However, the insistence on repeatability and careful experimentation allows such claims to be dismissed by the scientific majority within a relatively short period of time. We need to insist on the same level of evaluation and proof with regard to claims about software engineering techniques and tools. Unfortunately, this is rarely done, and our belief in silver bullets persists. What data is collected is often used in the service of advocacy rather than inquiry.

I am not advocating that everyone stop the research they are doing in software engineering and start testing hypotheses and building foundations. Invention is a very important part of progress in engineering. Tools and techniques are needed for the serious problems we face today. But inventions that are based on established principles will be more effective in solving the complex problems we are attempting to solve. We need to recognize the unproven assumptions and hypotheses underlying our current software engineering techniques and tools and evaluate them in the context of what has actually been demonstrated about these hypotheses instead of what we would like to believe. And we need to understand our problems before we can devise good solutions to them: Not enough experimentation or research is directed toward understanding our problems better as opposed to proposing solutions.

Like the exploding boilers, our ability to build safe software-controlled systems and to build effective software engineering tools to accomplish this will be enhanced by greater understanding of the scientific foundations of our craft.

Parallel 4 — *The introduction of safety devices for steam engines was inhibited not only by the lack of underlying scientific knowledge about boilers, but also by a narrow view of attempting to design a technological solution without looking at the social and organizational factors involved and the environment in which the device is used.*

A major airline, known for having the best aircraft maintenance program in the world, a few years ago introduced an expert system to aid their maintenance staff. The quality of maintenance fell. The staff began to depend on the computerized decision making and stopped taking responsibility and making their own decisions. When the software was changed to provide only information and only when requested, quality again rose. A similar example of this phenomenon has been found in aircraft: Hazardous situations have resulted when the introduction of computers increased pilot complacency and reliance and reduced situational awareness. The use of computers to enhance safety may actually achieve the opposite effect if human factors and the environment in which the computer will be used are not carefully considered.

Some people have suggested that the solution is to remove humans from critical loops completely. However, in doing this, they are placing unjustified reliance on the ability of programmers to foresee all eventualities and correctly predetermine the best solution under all circumstances. And even highly automated systems need humans for supervision, maintenance, and operation.

Another aspect of technological narrowness is the emphasis on technical solutions over organizational and managerial considerations. Nearly every major accident of the past 20 years (for example, Three Mile Island, Chernobyl, Challenger, Bhopal, and Flixborough) involved serious organizational and managerial deficiencies. Management that does not place a high priority on safety can defeat the best efforts by the technical staff. In each of the recent accidents noted, the organizations had sophisticated safety programs and safety devices. In each case, the potential effectiveness of the safety devices was canceled out by nontechnical factors. The concern, responsibility, and accountability for safety in an organization may be as important as, or more important than, technology.

Parallel 5 — *The operators of steam engines received most of the blame for accidents, not the designers or the technology.*

It is unfortunately very common to blame the operators for accidents when they have been put into a situation where human error is inevitable. This is as common today as it was a hundred years ago. And it is becoming a more serious problem as software engineers start to design human/machine interfaces without adequate knowledge about human factors and without the benefit of decades of gradual improvement of designs through experience.

As an example, although it is almost universally believed that pilot errors account for the majority of aircraft accidents, a US Air Force study of 681 inflight emergencies showed 659 crew recoveries for equipment and maintenance deficiencies with only 10 pilot errors. Other aerospace studies show that about 80 percent of aircraft-pilot-related accidents are due to poor training or neglect of human engineering in controls and instruments, not to stupidity or panic.[11]

Humans are effective in emergencies because of their ability to analyze a situation and come up with novel solutions. Humans work well when they have a deep understanding, a sound model of the world, that they can use to predict the results of their actions. Operators sometimes find it necessary to violate the rules in order to accomplish their tasks or to prevent or mitigate the consequences of accidents. The disruption that often occurs during a job action when employees "work to rule" demonstrates how necessary flexibility is. To make decisions during emergencies, operators must have an understanding of the system they are controlling and must be given proper information in a usable format.

Three Mile Island is a classic example of the misattributing of an accident to operators and the use of hindsight to label

operator action as erroneous. Operators are usually blamed for this accident although the accident sequence was initiated and compounded by equipment failure that was completely independent of operator action. Furthermore, the major errors of the operators could only be seen after the fact; at the time, there was not enough information about what was going on in the plant to make better decisions. In fact, the errors that the operators made have been labeled as inevitable given the existing instrumentation:[12] They were a direct function of the electromechanical system design. For example, the computer was hours behind in printing out alarms and information, although decisions had to be made in minutes, the instrumentation was unreadable under emergency conditions, and the wrong information was provided. Prior to the Three Mile Island accident, nuclear engineers took little interest in operator interface design. The Kemeny Commission's report on the accident concluded that the operator error was precipitated and compounded by basic flaws in system design.[13]

The Vincennes (Iranian Airbus) incident is well known, but many other less-publicized accidents have occurred due to poor design of the human/computer interface. At one chemical plant in Britain, a computer printed a long list of alarms when a power failure occurred. The design team had assumed that in such a situation the operator would immediately trip (shutdown) the plant. Instead, the operator watched the computer print the list of alarms and wondered what to do. The operator should not bear the responsibility alone here; if a person is overloaded with too much information, he or she is most likely to do nothing while trying to understand the situation.[14]

A basic understanding of human psychology and behavior is a prerequisite for user interface design that is commonly missing from software engineering education. A design, for example, that involves displaying data or instructions on a screen for an operator to check and verify by pressing the enter button will, over time and after few errors are found, result in the operator getting into the habit of pressing the enter key multiple times in rapid succession. Most of us have fallen into this trap ourselves.

The solution is obvious. Software engineers must take human factors more seriously, and human engineering experts must be involved in the design of safety-critical software interfaces.

Parallel 6 — *The early steam engines had low standards of workmanship, and engineers lacked proper training and skills.*

Building safety-critical software requires special skills and knowledge on the part of both developers and management. Like any quickly developing technology, demand for qualified personnel has outstripped the supply, and an appreciation of the skills and training necessary is often lacking.

People lacking in-depth knowledge of software engineering or the application area, and sometimes both, can be found building safety-critical software. All too typical is the man with a degree in nuclear engineering who told me that he builds software to control aircraft, although he does not really understand basic aeronautical principles (and, I suspect, software engineering principles). In addition, the education that *is* provided in software engineering too often narrowly focuses on computer skills without providing training in basic engineering skills.

Many government standards in the US require critical engineering projects to have at least one licensed Professional Engineer on their staff. System safety engineers have additional licensing requirements in many states. The standards do not usually require that every engineer on a project have a professional engineering or safety engineering license; however, a license is required for those holding certain positions on the project, such as lead engineer or system safety manager, along with requirements that they accept responsibility for assuring that the highest engineering standards and ethics are practiced. Nothing similar exists for any of the software engineers who are working on the same projects.

In his campaign against high-voltage electricity, Edison warned against the problems of poor workmanship and ignorance on the part of the majority of electrical contractors, just as Watt had emphasized the personal moral responsibility of the engineer to ensure a safe and efficient steam engine and the culpability of the engineer in case of accidents.[3] If we in software engineering do not ourselves insist on establishing minimum levels of competency and safety, then the government will step in and do it for us. The public expects, and has the right to expect, that dangerous systems are built using the safest technology available.

Watt, Edison, and other inventors of the 18th century campaigned to raise professional skills because they realized the potential harm of their inventions in the wrong hands. They anticipated the need for higher standards of safety and precision in the engineering of new technological systems, and they initiated the process of raising professional standards.[3] Edison and Watt believed that "engineers had a responsibility to produce competent work, including the utmost in safety."[3] Eventually, professional societies developed that took over the role of establishing safety and competency standards.

Such standards and licensing requirements must be carefully composed. The extensive regulation of high-voltage electricity distribution in Great Britain has been blamed for its slow adoption and the lag in electrical development compared to the US.[15] For example, regulations that set a minimum standard of insulation were stricter than was necessary and were blamed for the high cost of installation. But many British engineers argued that although the extensive regulation increased the cost, it also lessened the danger of fire and injury. As a group, British electrical engineers in the 1890s believed that lack of regulation in the US had helped the development of the electrical industry at the cost of more accidents, which were "so common as to be part and parcel of the system."[15] At the same time, British engineers were condemning Americans for their unsafe use and maintenance of steam boilers.

Just as overly strict regulations unnec-

essarily inhibited electrical-technology development in Britain in the last century, so poorly written standards can inhibit the development of computer technology. Worse, standards can inadvertently shift responsibility away from the manufacturers and developers to government agencies that have much less effective and direct control over the safety of the final product. And poorly written standards may have no effect or may even increase risk.

In our enthusiasm, we also do not want to impede progress by writing unachievable standards or inadvertently increase risk by implementing the wrong standards. As discussed earlier, we have not scientifically established the benefits and effectiveness of most of our software engineering techniques. Depending on a particular software engineering methodology to assure safety by assuming it will produce error-free or ultra-high reliability software is dangerous. In addition, mandating particular solutions, such as equating safety with the reliability of a protection system to recover from hazardous states, effectively eliminates the consideration of many potentially more effective techniques that eliminate hazardous states or minimize getting into them.[16] And as the technology progresses, standards that require the use of specific approaches often lag behind. Manufacturers may feel no ethical or legal duty to go beyond what is required in the standard.

Finally, manufacturers or those who will personally benefit financially from particular techniques being included or not included in the standards sometimes play a dominant role in the drafting process. The result may be watered-down requirements or the recommendation of techniques with more commercial than technical value.

The alternative is to construct flexible standards specifying general criteria for acceptability of a methodology instead of a specific methodology and ensuring that those building safety-critical software have the competency and personal responsibility to use the best approaches available at the time and for the particular project characteristics.

As Edison argued with respect to electricity, increased government regulation of our technology may not be to anyone's benefit; but it is inevitable unless we, as the technology's developers and users, take the steps necessary to ensure safety in the devices that are constructed and technical competence in those who construct them. ■

Acknowledgments

Several people provided helpful comments on earlier drafts of this article, including Daniel Berry, John Gannon, Susan Gerhart, David Notkin, David Parnas, Jon Reese, John Rushby, and Elaine Weyuker. It should not be assumed, however, that they necessarily agree with the points made in the article.

References

1. W.D. Ruckelshaus, "Risk, Science, and Democracy," *Readings in Risk*, in T.S. Glickman and M. Gough, eds., Resources for the Future, Washington, D.C., 1990, p. 108.
2. R.L. Hills, *Power from Steam: A History of the Stationary Steam Engine*, Cambridge University Press, Cambridge, UK, 1989.
3. R. Cameron and A.J. Millard, *Technology Assessment: A Historical Approach*, Kendall/Hunt, Dubuque, Iowa, 1985.
4. H.W. Dickinson, *A Short History of the Steam Engine*, Frank Cass, London, 1963.
5. J.G. Burke, "Bursting Boilers and the Federal Power," *Technology and Culture*, Vol. VII, No. 1, Winter 1966, pp. 1-23.
6. G.H. Archinoff et al., "Verification of the Shutdown-System Software at the Darlington Nuclear Generating Station," *Proc. Int'l Conf. on Control and Instrumentation in Nuclear Installations*, 1990.
7. W.C. Bowman et al., "An Application of Fault-Tree Analysis to Safety-Critical Software at Ontario Hydro," *Conf. on Probabilistic Safety Assessment and Management*, 1991.
8. S. Watts, "Computer Watch on Nuclear Plant Raises Safety Fears," *London Independent*, Oct. 13, 1991.
9. A. Aiken, "Fault Analysis," in *High-Risk Safety Technology*, A.E. Green, ed., John Wiley and Sons, New York, 1982.
10. N.G. Leveson and C.S. Turner, "An Investigation into the Therac-25 Accidents," *Computer*, Vol. 26, No. 7, July 1993, pp. 18-41.
11. W.G. Johnson, *MORT: Safety Assurance Systems*, Marcel Dekker, New York, 1980.
12. M.J. Brookes, "Human Factors in the Design and Operation of Reactor Safety Systems," in *Accident at Three Mile Island: The Human Dimensions*, D.L. Sills, C.P. Wolf, and V. Shelanski, eds., Westview Press, Boulder, Colo., 1982.
13. J.G. Kemeny, "The Need for Change: The Legacy of Three Mile Island," *Report of the President's Commission on Three Mile Island*, Pergamon Press, New York, 1979.
14. T. Kletz, "Wise After the Event," *Control and Instrumentation*, Vol. 20, No. 10, Oct. 1988, pp. 57-59.
15. A.J. Millard, *A Technological Lag: Diffusion of Electrical Technology in England 1879-1914*, Garland Publishers, New York, 1987.
16. N.G. Leveson, *Safeware: System Safety in the Computer Age*, Addison-Wesley, Reading, Mass., 1994.

Nancy G. Leveson is the Boeing Professor of Computer Science and Engineering at the University of Washington, Seattle. Her research interests are in formal modeling and analysis of software safety and reliability.

Leveson received her bachelor's degree in mathematics and a PhD in computer science from the University of California at Los Angeles. She is editor-in-chief of *IEEE Transactions on Software Engineering*. She is a member of the Board of Directors of the Computing Research Association, the National Research Council Commission on Engineering and Technical Systems, and the ACM Committee on Computers and Public Policy. She is a member of the IEEE Computer Society.

Readers can contact Leveson at the Computer Science and Engineering Department, FR-35, University of Washington, Seattle, WA 98195. Her e-mail address is leveson@cs.washington.edu.