Alexander Charns, *Cloak and Gavel: FBI Wiretaps, Bugs, Informers, and the Supreme Court,* University of Illinois Press, 1992.

David Chaum, "Achieving Electronic Privacy," *Scientific American,* August 1992, pp. 96–101.

Dorothy E. Denning, "The Case for 'Clipper'," *Technology Review,* July 1995, pp. 48–55. Denning makes the case in favor of the Clipper Chip. For another article by Denning see www.cosc.georgetown.edu/~denning/crypto/Future.html.

Dorothy E. Denning et al, "To Tap or Not To Tap," *Communications of the ACM,* March 1993, 36:3, pp. 25–44. This debate on wiretapping and encryption policy includes an article by Denning and responses from a variety of points of view.

David Flaherty, *Protecting Privacy in Surveillance Societies,* University of North Carolina Press, 1989.

Robin Hanson, "Can Wiretaps Remain Cost Effective?" *Communications of the ACM,* December 1994, 37:12, pp. 13–15. An economic analysis of the costs and benefits of the wiretap act.

Lance J. Hoffman, editor, *Building In Big Brother: The Cryptographic Policy Debate,* Springer Verlag, 1995.

Lance J. Hoffman, Faraz A. Ali, Steven L. Heckler, Ann Huybrechts, "Cryptography Policy," *Communications of the ACM,* September 1994, 37:9, pp. 109–17.

S. Kent et al, "Codes, Keys and Conflicts: Issues in US Crypto Policy, Report of a Special Panel of the ACM US Public Policy Committee," June 1994 (http://info.acm.org/reports/acm_crypto_study.html).

Edith Lapidus, *Eavesdropping on Trial,* Hayden Book Co., 1974. Contains history of wiretapping and the relevant sections of the Omnibus Crime Control and Safe Streets Act of 1968.

National Research Council, *Cryptography's Role in Securing the Information Society,* National Academy Press, 1996 (www2.nas.edu/cstbweb).

Bruce Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C,* John Wiley & Sons, Inc., 1994.

Alan F. Westin, *Privacy and Freedom,* Atheneum, 1968. Contains history of wiretapping and other means of surveillance.

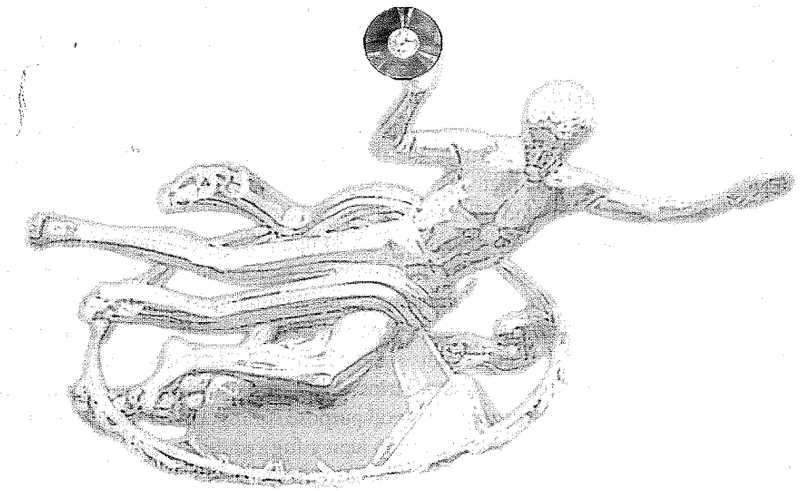
Philip R. Zimmermann, *The Official PGP User's Guide,* MIT Press, 1995. Includes discussion of the legal, ethical, and political issues surrounding PGP.

Philip R. Zimmermann, *PGP Source Code and Internals,* MIT Press, 1995.

Kapitel 4 aus: Sara Baase: A Gift of Fire.
Social, Legal, and Ethical Issues in Computing.
1997, Prentice-Hall, New Jersey, S. 113-164.

4

CAN WE TRUST THE COMPUTER?



4.1 WHAT CAN GO WRONG?

4.1.1 Questions About Reliability and Safety

"Data Entry Typo Mutes Millions of U.S. Pagers"
"Software Errors Cause Radiation Overdose"
"IRS Computer Sends Bill For $68 Billion in Penalties"
"Software Glitch Clogs AT&T Telephone Lines"
"Robot Kills Worker"
"DMV Spent $44 Million on Failed Computer Project"
"Man Arrested Five Times Due to Faulty FBI Computer Data"
"High-Tech Baggage System 'Eats' Luggage"
"Computer Predicts We Will Run Out of Copper by 1985"

What can go wrong when we use computers? Almost anything. Most computer applications, from consumer software to systems that control airplanes and telephone networks, are so complex that it is virtually impossible to produce a program with no errors. In this chapter, we will describe a variety of mistakes, problems, and failures involving computers—and some of the factors responsible for them. Some errors are minor; for example, a word processor might incorrectly hyphenate a word that does not fit at the end of a line. (The system I used for this book broke the word "robots" into "robot" and "s.") Some form letter systems begin letters with "Dear Mr. Association" because "Association" is the last word in the first line of an address on a mailing list. Some incidents are funny; some are tragic; some cost millions of dollars. All of the examples can teach us something. We will look at one case in depth (in Section 4.2): the Therac-25. This computer-controlled radiation treatment machine had a large number of flaws that resulted in the deaths of several patients. Section 4.3 looks at a few factors related to computer failures in more depth and describes some approaches to reducing problems. Section 4.4 puts the risks of computer systems in perspective by considering risks (and ways to reduce them) in other systems. In Sections 4.5–4.7 we consider the reliability of complex analysis and predictions made by computers based on mathematical models.

The headlines above and the examples we will describe raise many questions. Are we risking major disasters from breakdowns in computerized banking and communication systems? Are computer-controlled medical devices, factory automation systems, and airplanes too unsafe to use? Are we too dependent on computers?

Or, like many stories on the evening news, do the headlines and horror stories emphasize the dramatic and unusual events—the bad news? Car crashes are reported on the news, but we do not hear that 200,000 car trips were completed safely in our city today. Although most car trips *are* safe, there is a good purpose for reporting crashes on the news: It teaches us what the risks are (e.g., driving in heavy fog), and it reminds us to be responsible and careful drivers.

Just as car crashes can be caused by many factors (faulty design, sloppy manufacturing or servicing, bad road conditions, a careless or poorly trained driver, confusing road signs, etc.), computer glitches and system failures also have a myriad of causes, including faulty design, sloppy implementation, careless or insufficiently trained users, and poor user interfaces. Often more than one factor is involved. Sometimes, no one did anything clearly wrong, but an accident occurs. Occasionally, the irresponsibility of software developers is comparable to driving while very drunk.

Although millions of computers and software programs are working fine every day, it is crucial that we understand the risks and reasons for computer failures. How much risk must or should we accept? If the inherent complexity of computer systems means they will not be perfect, how can we distinguish between errors to accept as trade-offs for the benefits of the system and errors that are due to inexcusable carelessness, incompetence, or dishonesty? How good is good enough? When should we, or the government, or a business decide that a computer is too risky to use? We cannot answer these questions completely, but this chapter provides some background and discussion that can help you in forming conclusions.

This chapter should help us understand computer-related problems from the perspective of several of the roles we play, specifically,

- **A computer user.** Whether we use a personal computer at home or a sophisticated, specialized system at work, we should understand the limitations of computers and the need for proper training and responsible use. We must recognize that, as in other areas, there are good products and bad products.
- **A computer professional.** Automotive engineers study car crashes to determine their cause and build safer cars in the future. Studying computer failures should help you become a better computer professional (system designer, programmer, or quality assurance manager, e.g.) if that is your career direction. (In Chapter 10, we discuss aspects of professional ethics that relate to the quality and safety of computer systems.)
- **An educated member of society.** There are many personal decisions and social, legal, and political decisions that depend on our understanding of the risks of computer system failures. We may be on a jury. We may be an active member of an organization lobbying for legislation. We may be deciding whether or not to try an experimental computer-controlled medical device or whether to fly in a new computer-controlled airplane. One goal of this chapter is to provide some perspective and analysis to help us evaluate the reliability and safety of various computer applications and of computer technology in general.

Computer errors and failures can be categorized in several ways, for example, by the cause, by the seriousness of the effects, or by the application area. In any scheme used to organize the discussion, there will be overlap in some categories and mixing of diverse examples in some. For the remainder of this section I will use three categories: problems for individuals, usually in their roles as consumers; system failures that affect large numbers of people and/or cost large amounts of money; and problems in safety-critical applications where people may be injured or killed.

The incidents described here are a sampling of the many that occur. In most cases, by mentioning specific companies or products, I do not mean to single those out as unusual

offenders. One can find many similar stories in newspapers and magazines—and especially in the Risks Forum (the comp.risks newsgroup on Usenet) organized by Peter Neumann. Neumann has collected thousands of reports describing a wide range of computer-related problems.

4.1.2 Problems for Individuals

Many people are inconvenienced and/or suffer losses from errors in billing systems and in databases containing personal data. Users of home computers confront frustrating bugs in operating systems and applications software.

Billing errors

The first few errors we look at are relatively simple ones whose negative consequences were relatively easily undone.[1]

- A woman was billed $6.3 million for electricity; the correct amount was $63. The cause was an input error made by someone using a new computer system.
- In 1993, the IRS modified their programs with the intent of not billing Midwest flood victims. Instead, the computer generated erroneous bills for almost 5000 people. One Illinois couple received a bill for a few thousand dollars in taxes—and $68 billion in penalties.
- The auto insurance rate of a 101-year-old man suddenly tripled. Rates depend on age, but the program was written to handle ages only up to 100. It mistakenly classified the man as a teenager.
- Hundreds of Chicago cat owners were billed by the city for failure to register dachshunds, which they did not own. The city was using computer matching with two databases to try to find unlicensed pets. One database used DHC as the code for domestic house cat, and the other used the same code for dachshund.

The first three problems came from errors in the design and/or implementation of the programs. Some errors could have been avoided with more care. Some problems resulting from the errors could have been avoided if the programs had included tests to determine if the amount was outside some reasonable range or changed significantly from previous bills. In other words, because programs may contain errors, good systems have provisions for checking their results. If you have some programming experience, you know how easy it would be to include such tests and make a list of cases for someone to review. These errors are perhaps more humorous than serious. When mistakes are as big as these, they are obvious, and the bills are corrected. They are still worth studying because the same kinds of design and programming errors can have more serious consequences in different applications. In the Therac-25 case (Section 4.2) we will see that including tests for inconsistent or inappropriate input could have saved lives.

In the fourth example above, it was not errors in the software or the databases that caused the incident; the city staff did not know enough about the systems they used.

Database accuracy problems

In Chapter 2, we discussed privacy issues related to large government and private databases containing personal information. We deferred the problem of accuracy to this chapter. If the information in a database is not accurate, we can suffer inconvenience or serious harm. When information is entered automatically by other computer systems, mistakes that might be obvious to a human can be overlooked. Even if an error is corrected, the problems may not be over for the person affected. Computer records are copied easily and often; copies of the incorrect data may remain in other systems.

When interest rates dropped in the early 1990s, hundreds of people applying for new mortgages discovered that their credit reports mistakenly listed a late payment for their current mortgage. The error had occurred when one bank, call it Bank A, bought another, Bank B. It took more than a month for Bank A to transfer all the old Bank B mortgage accounts to Bank A's system. Payments made by thousands of Bank B customers were recorded as being paid when the transfer was completed—more than a month after they were actually paid. The "late" payments were automatically reported to the credit bureaus. This error did not have serious consequences. Bank A quickly realized what had caused the problem and sent correction letters to all three credit bureaus. Cases of errors in credit bureau files do not always end this well. Credit bureaus have received heavy criticism for incidents where incorrect information has caused people to lose their homes, cars, jobs, or insurance. Thousands of residents of New England were listed incorrectly in TRW records as not having paid their local property taxes. The problem was attributed to an input error. People were denied loans before the scope of the problem was identified and it was corrected. (TRW paid damages to many of the people affected.)[2]

A county agency used the wrong middle name in a report to a credit bureau about a father who did not make his child support payments. Another man in the same county had the exact name reported; he could not get credit to buy a car or a house. A woman in Canada could not get her tax refund because the tax agency insisted she was dead. Her identification number had been mistakenly reported in place of her mother's when her mother died. Note that although computerized records were used in these cases, computers did not cause the problem. The source of the problems was incorrect data being entered into records; they might have been as likely to occur with paper forms.

A 14-year-old boy in his first year of high school was excluded from football and some classes without explanation. He eventually learned, almost by accident, that school officials thought he had been using drugs while in junior high school. The two schools used different disciplinary codes in their computerized records. The boy had been guilty of chewing gum and being late.[3] This case is very similar to the case of the dachshund/cat confusion described earlier—except that the consequences were more significant. Both cases illustrate the problems of relying on computer systems without taking responsibility to learn enough about them to use them properly.

The Medical Information Bureau (MIB) maintains records on roughly half a million people; it estimates that 3–4% of its records contain errors. Some may be minor and not have serious consequences, but people have charged that their records contained incorrect reports of Alzheimer's disease, heart attacks, and drug and alcohol abuse. Such information can result in denial of jobs or insurance. (According to MIB rules, companies are not supposed to make decisions solely based on the MIB report, but privacy advocates believe that some do.)[4]

Some of the cases we mentioned can be extremely disruptive and damaging to people's lives and financial situations. When the errors are in databases used by law enforcement agencies, the consequences can include arrest at gunpoint, strip searches, and being jailed, often with violent criminals. Studies of the FBI's National Crime Information Center (NCIC) database in the 1980s found that roughly 11% of the arrest warrants listed in it were inaccurate or no longer valid. People are arrested when a check of the database shows a warrant for them—or for someone with a similar name. I will mention a few NCIC cases; the news media and government studies have reported many more.

An adoption agency ran a routine check on an applicant and found that he had been convicted of grand larceny. In fact, he had been involved in a college prank, stealing a restaurant sign, years before, and the charges had been dropped after he apologized and paid for the damage. The error could have caused the agency to deny the adoption. A Michigan man was arrested for several crimes, including murders, committed in Los Angeles. Another man had assumed his identity after finding his lost wallet (or a discarded birth certificate; reports varied). It is understandable that the innocent man was listed in NCIC as wanted. However, he was arrested four more times within 14 months. (After repeatedly asking the city of Los Angeles to correct the records, he sued and won a judgment against the city.) A man was imprisoned at a military base for five months because NCIC mistakenly reported that he was AWOL. A college professor returning from London was arrested and jailed for two days after a routine check with NCIC at Customs showed that he was a wanted fugitive. NCIC was wrong—for the third time about this particular man. Similar problems occur with local police systems: An innocent driver was stopped by police and frisked because his license plate number was incorrectly listed as the license number of a man who had killed a state trooper. The computer record did not include a description of the car.[5]

There are several factors in causing the severity of the problems that result from errors in databases: a large population (where many people have identical or similar names); the fact that most of our financial interactions are with strangers; automated processing without human common sense or the ability to recognize special cases; overconfidence in the accuracy of data stored on computers; and a lack of accountability for failure to update information and correct errors.

Consumer hardware and software

> *Why did Intel name its new chip the Pentium? –Because when it added 100 to 486, it got 585.994257.*
>
> –One of many jokes about the Pentium chip flaw.

Several major operating systems and applications software packages for personal computers have had serious errors in their first releases. Computer scientists and cypherpunks (computer users with expertise in cryptography) discovered significant security flaws in Netscape, the popular World Wide Web browser program. Software is routinely sold with known bugs. Bugs in popular tax preparation programs sold by Intuit caused the wrong tax to be computed. The company was aware of at least one bug but shipped the program without warning customers. After the problems were publicized, Intuit promised to pay interest and penalties caused by the errors.[6] The calculator in some versions of Microsoft Windows shows the result of 2.01–2 as 0.00.*

Personal computer hardware also has flaws. Computer chips are increasingly complex. Sophisticated algorithms, some that used to be implemented in software, are now implemented on chips. Intel received a flurry of intensely negative publicity when a mathematician discovered that the Pentium chip had a bug in the process used for division of numbers represented in floating-point format.† Intel was "flamed" on the Internet, criticized by competitors, and ridiculed by numerous widely circulated jokes about the Pentium. The bug was of most concern in scientific applications, which use floating-point computations extensively. Home users of PCs were not likely to experience a problem, and sales of Pentium machines were strong during and after the controversy. What is interesting about the Pentium incident is that a flaw in a chip is not unusual. Intel's 386 and 486 chips had math flaws. The floating-point unit in some of Motorola's PowerPC chips did not work as specified. During the Pentium controversy, Compaq was recalling notebook computers that had memory problems and were described as "plagued by bugs." Some Hewlett-Packard workstations crashed and corrupted data because of a chip flaw. What angered so many people about the Pentium was that Intel knew of the problem but did not tell customers. Intel's response, after the bug was disclosed, was perceived by many people as being customer-hostile, and led to more negative publicity. Eventually the negative publicity prompted an apology from the company and a new policy of replacing the chip for any customer who asked.[7]

The mathematician who disclosed the Pentium flaw stated that "microprocessors [have] become so complex that it is no longer possible to completely debug them, or even to determine every bug that exists in one."[8] The same, of course, is true for software. Manufacturers make trade-offs between additional debugging and getting a product to market sooner. Technically oriented customers grumble but make do. Software sellers say that as more ordinary people begin using computers, they will be less tolerant of problems and glitches. Bugs in Walt Disney Co.'s "Lion King" CD-ROM, for example, led to thousands of complaints from parents of disappointed children. (As in the Pentium case, consumer anger was increased by Disney's initial unhelpful response to the problems.) Consumers are angered by dishonesty (i.e., by companies selling products with known serious flaws without telling customers about them) and by denials of problems and lack of adequate response to complaints. Some businesses will continue to try to hide problems, whereas others are recognizing that honesty and customer service are good business. Software producers expect market pressure to force improvement in consumer software. One commented, "As the consumer starts discriminating on what they'll buy, you have to get better."[9]

---

*The bug is apparently in the display, not the internal result; subsequent calculations using the result treat it as 0.01.

†Floating point format is used for very large numbers and for numbers with a fractional, or decimal part, for example, 1.5.

4.1.3 System Failures

No matter how carefully designed and operated a system is, it can still fail grotesquely.

–PETER G. NEUMANN, *Software Engineering Notes*[10]

Modern communications, banking, and financial systems depend heavily on computers. The computers do not always function as planned. The costs of failures can include millions of dollars and virtually complete shutdown of basic services.

Communications

Nationwide AT&T telephone service for voice and data was disrupted for nine hours in January 1990 because of a software error in a four-million line program. The disruption was variously described as a "slowdown," a "shutdown," and a "virtual paralysis." It prevented roughly 50 million calls from getting through. AT&T's official report stated, "While the software had been rigorously tested in laboratory environments before it was introduced, the unique combination of events that led to this problem couldn't be predicted."[11] In June and July 1991, telephone networks in several major east coast and west coast cities failed. The cause was a three-line change in a two-million line telecommunications switching program. The program had been tested for 13 weeks, but was not retested after the change—which contained a typo. In November 1991, a four-hour telephone outage in New England occurred when a technician changed a piece of disk equipment. Flights at Logan Airport in Boston were delayed or canceled because the communication systems used by the controllers and the pilots is connected to the AT&T system that failed. Commenting on the failure of millions of pagers, alluded to in one of the headlines at the beginning of this chapter, a company official said, "We designed our system architecture expressly so this couldn't happen. Of course, we *thought* it couldn't happen." The problem occurred when someone typing codes into a database forgot to hit the enter key in a line of data.[12]

Business and financial systems

The NASDAQ stock exchange was virtually shut down for two and a half hours in July 1994 because of a problem with new communications software that had been installed earlier in the week. A backup system also failed. Another computer failure caused an hour-long shutdown a year later. More than 5000 stocks are traded on NASDAQ, which operates over 200,000 terminals around the country. In a few incidents, large investment firms and newspapers reported incorrect or out-of-date stock prices; they blamed computer errors.[13] Other stock and commodities exchanges have halted trading because of computer problems. So far, the costs of these kinds of errors have been primarily lost business to the stock exchanges and losses for individual investors (both of which can be substantial). There is concern that future errors in stock, banking, or other financial systems could trigger a recession or a serious world-wide economic disruption.

In Chapter 1 we mentioned the ATM system that doubled the amount withdrawn from customer accounts. At another major bank a software error disrupted processing of

transactions; the bank had to borrow almost $24 billion overnight to cover its shortfall—and pay about $5 million in interest.[14]

A computer error in a contest sponsored by Pepsi Cola in the Philippines caused 800,000 winning numbers to be generated instead of the intended 18. The face value of the winnings would have been $32 billion. Pepsi paid nearly $10 million to customers with winning numbers to maintain "good will," but still faced hundreds of lawsuits and criminal complaints.[15] Pepsi Cola is large enough to absorb a $10 million expense; smaller companies have been destroyed by computer errors.

Destroying businesses

Once the fourth largest carpet distributor in the U.S., Kane Carpet Company went out of business 17 months after installing a new computerized inventory control system. The company estimated its losses at $37 million and blamed it all on the inventory system. A few dozen companies that bought another inventory system called Warehouse Manager blame the system for disastrous losses; one previously successful company saw its income decline by about half and laid off half its employees. The specific complaints were numerous. One company could not get the system to place a purchase order for several weeks; it claimed the backlog in orders cost $2000 per day. Processes that were supposed to take seconds, such as printing invoices, took several minutes while customers waited in long lines. Both systems gave incorrect information about inventory. Clerks were told that products were in stock when they were not, and *vice versa.* Both errors led to dissatisfied customers and lost sales. According to users of Warehouse Manager, the system reported incorrect prices to clerks. A part that cost $114 was listed for sale at 54 cents. A $17 part was listed for sale at $30. One error means lost money for the company; the other means lost customers who find a better price elsewhere. When two clerks tried to access the computer from their terminals simultaneously, the terminals locked up. Some companies said the system erased information needed for accounting and tax reports.[16]

What was responsible for the problems in Warehouse Manager? The program was sold by NCR Corporation, but it was developed by another company. It was originally designed for and implemented on a different computer and operating system. It appears that there were unexpected problems when the program was rewritten for NCR's machines and its ITX operating system. According to the *Wall Street Journal,* internal memos at NCR reported that the system had been inadequately tested and was performing badly in real business settings. NCR salespeople told prospective customers that Warehouse Manager was running successfully at 200 installations, but most of them were installations using the machine for which the program was originally designed. Several users claim that although NCR was receiving complaints of serious problems from many customers, the company told them the problems they were having were unique.

NCR blamed the problems on the company that wrote Warehouse Manager and modified it for ITX. Eventually NCR agreed it "did not service customers well" and the program should have been tested more. The company settled most of the few dozen lawsuits out of court, with confidentiality agreements about the terms.

The sources of the problems in this case included technical difficulties (converting software to a different system), poor management decisions (inadequate testing), and, according to the customers, dishonesty in promoting the system and responding to the problems.

Delayed and abandoned systems

Software glitches delayed the opening of a highly automated packing plant for Ben & Jerry's ice cream company. A bug in a new checkout scanner program developed for Walgreen, a supermarket chain, occasionally caused an incorrect price to be used. Walgreen delayed introduction of a new inventory control system for six months while the problem was solved. An $810 million project to upgrade the radar systems at Canada's major airports was delayed because of software errors. Among other problems, the program showed some airplanes flying backwards.[17]

Many systems are so fundamentally flawed that they are junked after wasting millions of dollars. The California Department of Motor Vehicles, for example, abandoned a $44 million computer system that never worked properly. A consortium of hotels and a rental car business spent $125 million on a comprehensive travel industry reservation system, then canceled the project because it did not work.[18]

There are many more such examples. One infamous case of a delay caused by a faulty computer system is the Denver International Airport.

The Denver Airport baggage system

> *What is the difference between the new Denver Airport and the White House?*
> *–You can land a plane at the White House.*
>
> –A joke about the delays in opening the Denver International Airport*

I saw an odd sight when I flew past the huge new Denver International Airport and the miles of wide highway leading to it. The airport covers 53 square miles, roughly twice the size of Manhattan. I saw nothing moving at the airport and no cars on the road—10 months after the $3.2 billion airport was to have opened in October 1993. The opening was rescheduled at least four times until the actual opening in 1995. The delay cost more than $30 million per month, or over one million dollars a day, in bond interest and operating costs. Most of the delay has been attributed to the now infamous computer-controlled baggage-handling system, which cost $193 million.[19]

The plan for the baggage system was quite ambitious. Outbound luggage checked at ticket counters or curbside counters was to be delivered to any part of the airport in less than 10 minutes via an automated system of cars traveling up to 19 miles per hour on 22 miles of underground tracks. Similarly, inbound luggage was to be delivered to terminals or transferred directly to connecting flights anywhere in the airport. Each bag is put into a car bar-coded for its destination. Laser scanners throughout the system track the 4000 cars and send information about their locations to computers. The computers use a database of flights, gates, and routing information to control motors and track switches to route the cars to their destinations. The complete system includes about 100 computers.

It did not work as planned. During tests of the system over several months, cars crashed into each other at track intersections; luggage was misrouted, dumped, and flung

*In 1994 a pilot crashed a small plane on the White House lawn.

about; and cars that were needed to move luggage were mistakenly routed to waiting pens.

Both the specific problems and the general underlying causes are instructive. Some of the specific problems were as follows:

- **Real-world problems.** Some of the scanners got dirty or were knocked out of alignment and could not detect cars going by. This was related to the car crashes.
- **Problems in other systems.** The airport's electrical system could not handle the power surges associated with the baggage system; the first full-scale test blew so many circuits that the test had to be halted. Faulty latches on the cars caused luggage to be dumped on the tracks between stops.
- **Software errors.** For example, the routing of cars to waiting pens when they were actually needed was attributed to a software error.

No one expects software and hardware of this complexity to work perfectly the first time it is tested. In real-time systems,* especially, there are numerous interactions and conditions that may not be anticipated. It is not surprising that problems would be encountered during development. Mangling a suitcase is not embarrassing if it occurs during an early test and if the problem is fixed. It is embarrassing if it occurs after the system is in operation or if it takes a year to fix the problems. What led to the extraordinary delay in the Denver baggage system? There seem to have been two main problems:

- **The time allowed for development and testing of the system was insufficient.** The only other baggage system of comparable size is at Frankfurt Airport in Germany. The company that built that system spent six years on development and two years testing and debugging. BAE Automated Systems, the company that built the Denver system, was asked to do it in two years. Some reports indicate that because of the electrical problems at the airport, there were only six weeks for testing.
- **Significant changes in specifications were made after the project began.** Originally, the automated system was to serve United Airlines, but Denver officials decided to expand it to include the entire airport, making the system 14 times as large as the automated baggage system BAE had installed for United at San Francisco International Airport.

PC Week's reporter said, "The bottom-line lesson is that system designers must build in plenty of test and debugging time when scaling up proven technology into a much more complicated environment."[20] Some observers criticize BAE for taking on the job when the company should have known that there was not enough time to complete it. Others blame the city government for poor management, politically motivated decisions, and proceeding with a grandiose but unrealistic plan.

*Real-time systems are systems that must detect and control activities of objects in the real world within time constraints.

4.1.4 Safety-Critical Applications

There are many examples of problems in safety-critical computer systems in military applications, power plants, aircraft, trains, automated factories, medical applications, and so on. Most of the deaths that have occurred because of computer-related problems were in aviation and medical applications.[21] We will look briefly at a few aviation cases, then at one medical instrument case in depth in the next section.

Computers in the air

The A320 Airbus airplane was the first fully "fly-by-wire" airplane. The pilots do not directly control the plane; their actions are inputs to computers that control the aircraft systems. Between 1988 and 1993, four A320s crashed. Although the official cause for some of the crashes was ruled "pilot error," pilots and some observers fault the fly-by-wire system. Pilots have complained that the airplane does not respond as expected, that it seems to have "a mind of its own" and may suddenly behave in unexpected and inappropriate ways. In the 1992 crash, the pilots specified a rate of descent of 3300 feet per minute instead of the normal 800 feet per minute. The official report on the crash indicated that reasons for the error probably included the pilots' lack of familiarity with the A320 automation equipment and confusing design of the controls and displays. The crew left the "vertical navigation" entirely to the automatic systems although there were indications that the descent rate was too high. Perhaps they had too much confidence in the computer's ability to detect and correct mistakes. In the 1993 crash, the computer did not recognize that the plane had landed; it prevented the pilot from reversing engine thrust to brake the airplane. Pilots and human-factors specialists emphasize the need for an easy way to override the computer and easy transfer between automatic and manual control.[22]

While there has been much concern about the possibility of crashes caused by computerizing pilot functions, the lack of computer automation was considered a factor in a 1995 crash that killed 160 people. According to an FAA official, computer automation has reduced or eliminated some types of pilot errors while introducing new ones.[23]

The Traffic Collision Avoidance System (TCAS) is intended to detect a potential in-air collision and direct the airplanes to avoid it. The first version of the system had so many false alarms that it was unusable. TCAS II still has a high rate of false alarms in some situations; more improvements are being made in the software. Some pilots complained that the system directed them to fly toward a nearby airplane rather than away from it, potentially causing a collision rather than avoiding one. (To its credit, TCAS has helped avoid some collisions.)[24]

Several crashes of U.S. Air Force Blackhawk helicopters, killing nearly two dozen people, were eventually attributed to radio interference with the computer system that controlled the helicopter.[25]

Air traffic control

There were more than a dozen computer breakdowns at the air traffic control center for the Chicago area in 1995. The result was long delays in flights and increased risk of collisions. The problem is that the system uses computers that are 25–30 years old, so old that spare parts and service are not available from the manufacturers any longer. The Federal Aviation Administration, which operates the air traffic control system, is cutting

hiring and training of technicians who repair and maintain the equipment. The problem here, though a "computer breakdown," has little to do with any inherent weakness in computers. Equipment ages, and the number of aircraft miles flown has almost doubled in the past 25 years. The source of the problem is political: The federal government funds many foolish, unimportant, or special-interest programs, but it has not made a priority of the safety of the millions of people who fly.

4.2 CASE STUDY: THE THERAC-25

4.2.1 Therac-25 Radiation Overdoses

The Therac-25 was a software-controlled radiation therapy machine used to treat people with cancer. Between 1985 and 1987, Therac-25 machines at four medical centers gave massive overdoses of radiation to six patients. In some cases, the operator repeated an overdose because the machine's display said that no dose had been given. Medical personnel later estimated that some patients received between 13,000 and 25,000 rads,* where the intended dose was in the 100–200 rad range. These incidents caused severe injuries and the deaths of three patients.

What went wrong?

Studies of the Therac-25 incidents showed that many factors were involved in causing the injuries and deaths. The factors include lapses in good safety design, insufficient testing, bugs in the software that controlled the machines, and an inadequate system of reporting and investigating the accidents. (Articles by computer scientists Nancy Leveson, Clark Turner, and Jonathan Jacky are the main sources for this discussion.[26])

To understand the discussion of the problems, it will help to know a little about the machine. The Therac-25 is a dual-mode machine; that is, it can generate an electron beam or an X-ray photon beam. The type of beam to be used depends on the tumor being treated. The machine's linear accelerator produces a high-energy electron beam (25 million electron volts) that is dangerous. Patients are not to be exposed to the raw beam. The computer monitors and controls movement of a turntable on which three sets of devices are mounted. Depending on whether the treatment is electron or X-ray, a different set of devices is rotated in front of the beam to spread it and make it safe. It is essential that the proper protective device be in place when the electron beam is on. A third position of the turntable may be used with the electron beam off, and a light beam on instead, to help the operator position the beam in precisely the correct place on the patient's body. There were several weaknesses in the design of the Therac-25 that contributed to the accidents (including some in the physical design that we will not mention here).

4.2.2 Software and Design Problems

Design flaws

The Therac-25, developed in the late 1970s, followed earlier machines called the Therac-6 and Therac-20. It differed from them in that it was designed to be fully computer

*A rad is the unit used to quantify radiation doses. It stands for "radiation absorbed dose."

controlled. The older machines had hardware safety interlock mechanisms, independent of the computer, that prevented the beam from firing in unsafe conditions, for example, if the beam-attenuating devices were not in the correct position. Many of these hardware safety features were eliminated in the design of the Therac-25. Some software from the Therac-20 and Therac-6 was reused in the Therac-25. This software was apparently assumed to be functioning correctly. This assumption was wrong. When new operators used the Therac-20, there were frequent shutdowns and blown fuses, but no overdoses. The Therac-20 software had bugs, but the hardware safety mechanisms were doing their job. Either the manufacturers did not know of the problems with the Therac-20, or they completely missed their serious implications.

The Therac-25 malfunctioned frequently. One facility said there were sometimes 40 dose rate malfunctions in a day, generally underdoses. Thus operators became used to error messages appearing often, with no indication that there might be safety hazards.

There were a number of weaknesses in the design of the operator interface. The error messages that appeared on the display were simply error numbers or obscure messages ("Malfunction 54" or "H-tilt"). This was not unusual for computer programs in the 1970s when computers had much less memory and mass storage than they have now. One had to look up each error number in a manual for more explanation. The operator's manual for the Therac-25, however, did not include any explanation of the error messages. Even the maintenance manual did not explain them. The machine distinguished between the severity of errors by the amount of effort needed to continue operation. For certain error conditions, the machine paused, and the operator could proceed (turn on the electron beam) by pressing one key. For other kinds of errors, the machine suspended operation and had to be completely reset. One would presume that the one-key resumption would be allowed only after minor, not safety-related, errors. Yet this was the situation that occurred in some of the accidents in which patients received multiple overdoses.

Investigators studying the accidents found that there was very little documentation produced during development of the program concerning the software specifications or the testing plan. Although the manufacturer of the machine, Atomic Energy of Canada, Ltd. (AECL), a Canadian government corporation, claimed that it was tested extensively, it appeared that the test plan was inadequate.

Bugs

Investigators were able to trace some of the overdoses to two specific software errors. Because many readers of this book are computer science students, I will describe the bugs. These descriptions illustrate the importance of using good programming techniques. However, some readers have little or no programming knowledge, so I will simplify the descriptions.

After treatment parameters are entered by the operator at a control console, a software procedure, Set-Up Test, is called to perform a variety of checks to be sure the machine is positioned correctly, and so on. If anything is not ready, the routine schedules itself to be executed again so that the checks are done again after the problem is resolved. (It may simply have to wait for the turntable to move into place.) The Set-Up Test routine may be called several hundred times while setting up for one treatment. When a particular flag variable is zero, it indicates that a specific device on the machine is positioned correctly. To ensure that the device is checked, each time the Set-Up Test routine runs, it increments the

variable to make it nonzero. The problem was that the flag variable was stored in one byte. When the routine was called the 256th time, the flag overflowed and showed a value of zero. (If you are not familiar with programming, think of this as an odometer rolling over to zero after reaching the highest number it can show.) If everything else happened to be ready at that point, the device position was not checked, and the treatment could proceed. Investigators believe that in some of the accidents, this bug allowed the electron beam to be turned on when the turntable was positioned for use of the light beam, and there was no protective device in place to attenuate the beam.

Part of the tragedy in this case is that the error was such a simple one, with a simple correction. No good student programmer should have made this error. The solution is to set the flag variable to a fixed value, say 1, when entering Set-Up Test, rather than incrementing it.

In a real-time system where physical machinery is controlled, status is determined, and an operator enters—and may modify—input (a multitasking system), there are many complex factors that can contribute to subtle, intermittent, and hard-to-detect bugs. Programmers working on such systems must learn to be aware of the potential problems and to program using good techniques to avoid them. In some of the accidents, a set of bugs allowed the machine to ignore changes or corrections made by the operator at the console. When the operator typed in all the necessary information for a treatment, the program began moving various devices into place. This process could take several seconds. The software was written to check for editing of the input by the operator during this time and to restart the set-up if editing was detected. However, because of bugs in this section of the program, some parts of the program learned of the edited information while others did not. This led to machine settings that were incorrect and inconsistent with safe treatment. According to the later investigation by the Food and Drug Administration (FDA), there appeared to be no consistency checks in the program. The error was most likely to occur if the operator was experienced and quick at editing input.

4.2.3 Why So Many Incidents?

There were six known Therac-25 overdoses. You may wonder why the machine continued to be used after the first one.

The Therac-25 had been in service for up to two years at some clinics. It was not pulled from service after the first few accidents because it was not known immediately that it was the cause of the injuries. Medical staff members considered various other explanations. The staff at the site of the first incident said that one reason they were not certain of the source of the patient's injuries was that they had never seen such a massive radiation overdose before. The manufacturer was questioned about the possibility of overdoses, but responded (after the first, third, and fourth accidents) that the patient injuries could not have been caused by the machine. According to the Leveson and Turner investigative report, they also told the facilities that there had been no similar cases of injuries.

After the second accident, AECL investigated and found several problems related to the turntable (not including any of the ones we described). They made some changes in the system and recommended operational changes. They declared that the safety of the machine had been improved by five orders of magnitude, although they told the FDA that they were not certain of the exact cause of the accident; that is, they did not know if they had

found the problem that caused the accident or if they had just found other problems. In making decisions about continued use of the machines, the hospitals and clinics had to consider the costs of removing the expensive machine from service (in lost income and loss of treatment for patients who needed it), the uncertainty about whether the machine was the cause of the injuries, and later, when that was clear, the manufacturer's assurances that the problem had been solved. After some of the later accidents, machines were removed from service. They were returned to service after modifications by the manufacturer, but the modifications had not fixed all the bugs.

A Canadian government agency and some hospitals using the Therac-25 made recommendations for many more changes to enhance safety; they were not implemented. After the fifth accident, the FDA declared the machine defective and ordered AECL to inform users of the problems. The FDA and AECL spent about a year (during which the sixth accident occurred) negotiating about changes to be made in the machine. The final plan included more than two dozen changes. The critical hardware safety interlocks were eventually installed, and most of the machines remain in use with no new incidents of overdoses since 1987.[27]

4.2.4 Overconfidence

In the first overdose incident, when the patient told the machine operator that she had been "burned," the operator told her that was impossible. This was one of many indications that the makers and some users of the Therac-25 were overconfident about the safety of the system. The most obvious and critical indication of overconfidence in software was the decision to eliminate the hardware safety mechanisms. A safety analysis of the machine done by AECL years before the accidents suggests that they did not expect significant problems from software errors. In one case where a clinic added its own hardware safety features to the machine, AECL told them it was not necessary. (None of the accidents occurred at that facility.)

The hospitals using the machine assumed that it worked safely, an understandable assumption. Some of their actions, though, suggest overconfidence, or at least practices that should be avoided, for example, ignoring error messages because the machine produced so many of them. A camera in the treatment room and an intercom system enabled the operator to monitor the treatment and communicate with the patient. (The treatment room is shielded, and the console used by the operator is outside the room.) On the day of an accident at one facility, neither the video monitor nor the intercom was functioning. The operator did not see or hear the patient try to get up after an overdose; he received a second overdose before he reached the door and pounded on it. This facility had successfully treated more than 500 patients with the machine before the accident.

4.2.5 Conclusion and Perspective

From design decisions all the way to responding to the overdose accidents, the manufacturer of the Therac-25 did a poor job. Minor design and implementation errors might be expected in any complex system, but the number and pattern of problems in this case, and the way they were handled, suggests irresponsibility that merits high awards to the families of

the victims and possibly, some observers believe, criminal charges. This case illustrates many of the things that a responsible, ethical software developer should not do. It illustrates the importance of following good procedures in software development. It is a stark reminder of the consequences of carelessness, cutting corners, unprofessional work, and attempts to avoid responsibility. It reminds us that a complex system may work correctly hundreds of times with a bug that shows up only in unusual circumstances, hence, the importance of always following good safety procedures in operation of potentially dangerous equipment. This case also illustrates the importance of individual initiative and responsibility. Recall that some facilities installed hardware safety devices on their Therac-25 machines. They recognized the risks and took action to reduce them. The hospital physicist at one of the facilities where the Therac-25 overdosed patients spent many hours working with the machine to try to reproduce the conditions under which the overdoses occurred. With little support or information from the manufacturer, he was able to figure out the cause of some of the malfunctions.

Even if the Therac-25 case was unusual,* we must deal with the fact that the machine was built and used, and it killed people. There have been enough accidents in safety-critical applications to indicate that significant improvement is needed. Should we not trust computers for such applications at all? Or, if we continue to use computers for safety-critical applications, what can be done to reduce the incidence of failures? We will discuss some approaches in the next section.

To put the Therac-25 in some perspective, it is helpful to remember that failures and other accidents have always occurred and continue to occur in systems that do not use computers. Two other linear accelerator radiation-treatment machines seriously overdosed patients. Three patients received overdoses in one day at a London hospital in 1966 when safety controls failed. Twenty-four patients received overdoses from a malfunctioning machine at a Spanish hospital in 1991; three patients died. Neither of these machines had computer controls. Two news reporters reviewed more than 4000 cases of radiation overdoses reported to the U.S. government. The Therac-25 incidents were included, but most of the cases did not involve computers. Here are a few of the overdose incidents they describe. A technician started a treatment, then left the patient for 10–15 minutes to attend an office party. A technician failed to carefully check the prescribed treatment time. A technician failed to measure the radioactive drugs administered; she just used what looked like the right amount. In at least two cases, technicians confused microcuries and millicuries.† The general problems were carelessness, lack of appreciation for the risk involved, poor training, and lack of sufficient penalty to encourage better practices. In most cases, the medical facilities paid small fines or were not fined at all. (One radiation oncologist severely injured five women. He was eventually sued.)[28]

Some of these problems might have been prevented by good computer systems. Many could have occurred even if a computer were in use. None excuse the Therac-25. They suggest, however, that individual and management responsibility, good training, and accountability are more important factors than whether or not a computer is used.

---

*Sadly, some software safety experts say the poor design and lack of attention to safety in this case are *not* unusual.

†A curie is a measure of radioactivity. A milicurie is one thousand times as much as a microcurie.

4.3 INCREASING RELIABILITY AND SAFETY

4.3.1 What Goes Wrong?

Computer programs have tens of thousands, hundreds of thousands, or millions of lines of code. (Microsoft's Windows 95 has more than 11 million lines.) There is plenty of room for errors. Figure 4.1 lists common factors in computer errors and system failures. Most of them are illustrated in examples we have described. Some are technical issues, and some are managerial, social, legal, and ethical issues.

Overconfidence

Overconfidence, or an unrealistic or inadequate understanding of the risks in a complex computer system, is a core issue. When system developers and users appreciate the risks, they will then be more motivated to use the techniques that are available to build more reliable and safer systems and to be responsible users. How many PC users never backed up their files until after they had a disk crash and lost critical data or months of work?

- The complexity of real-time, multitasking systems.
- "Non-linearity" of computer software. This means that, whereas a small error in an engineering project may cause a small degradation in performance, a single typo in a computer program can cause a dramatic difference in behavior.
- Failing to plan and design for unexpected inputs or circumstances.
- Interaction with physical devices that do not work as expected.
- Incompatibility of software and hardware, or of application software and the operating system.
- Inadequate management.
- Insufficient testing.
- Carelessness.
- Business and/or political pressure to get a product out quickly.
- Misrepresentation, hiding problems.
- Inadequate response when problems are reported.
- Inadequate attention to potential safety risks.
- Data-entry errors.
- Inadequate training of users.
- Errors in interpreting results or output.
- Overconfidence in software.
- Lack of market or legal incentives to do a better job.

FIGURE 4.1: Some factors in computer system errors and failures.

Some safety-critical systems that failed (e.g., systems that control airplanes and trains) had supposedly "fail-safe" computer controls. In some cases the logic of the program was fine, but the failure resulted from not considering how the system interacts with real users (such as pilots) or real-world problems (such as loose wires or fallen leaves on train tracks).

Can the risks of failure in a system be analyzed and quantified? Yes, but the techniques for developing estimates of failure rates must be used carefully. For example, the computers on the A320 airplane each have redundant software systems designed by separate teams of programmers. The redundancy is a safety feature, but how much safety does it provide? The failure rate was supposed to be less than one failure per billion flight hours. It was calculated by multiplying the estimated failure rates of the two systems, one in 100,000 hours. The calculation is reasonable if the systems are independent. But safety experts say that even when programmers work separately, they tend to make the same kinds of errors, especially if there is an error, ambiguity, or omission in the program specifications.[29]

Unrealistic reliability or safety estimates can come from genuine lack of understanding, or carelessness, or intentional misrepresentation. People without a high regard for honesty sometimes give in to business or political pressure to exaggerate or to hide flaws, avoid unfavorable publicity, and avoid the expense of corrections or lawsuits. The manufacturer of the Therac-25 declared that changes in the system increased safety by five orders of magnitude (a factor of 100,000). It is hard to guess how they arrived at that figure.

Political pressure to produce inflated safety predictions is, of course, not restricted to computer systems. In 1986 the Challenger space shuttle broke apart, killing the seven people aboard. The investigation by Nobel Prize winner Richard Feynman sheds interesting light on how some risk estimates are made. Feynman found that NASA engineers estimated the chance that an engine failure would terminate a flight to be about one in 200–300. Their boss gave the official NASA estimate of the risk: one in 100,000. The document that justified this unbelievable (in Feynman's judgment) estimate, calculated it from failure estimates for various components. Feynman concluded that the failure rates for the components were chosen to yield the prechosen result of one in 100,000.[30] One lesson from the Therac-25 and Challenger is to be skeptical about numbers whose magnitude may seem unreasonable to common sense.

4.3.2 Professional Techniques

Software engineering and professional responsibility

The many examples of computer system errors and failures suggest the importance of using good software engineering techniques at all stages of development, including specifications, design, implementation, documentation, and testing. Although complex systems will not be perfect, there is a wide range between poor work and good work, as there is in virtually any field. Professionals, both programmers and managers, have the responsibility to study and use the techniques and tools that are available. Professional responsibility includes knowing or learning enough about the application field and the software or systems being used to understand potential problems and to do a good job. Obviously, this is especially important in safety-critical applications. (There was a case

where a programmer at a medical facility discovered that on the system he was using a process could fail to execute on time while a window was being moved on the screen. The system controlled a patient's respirator.[31])

The programming team for the Clearinghouse Interbank Payment System, which transfers about one trillion dollars a day among various banks, spent years working on the specifications for upgrading an earlier program; then they spent six months on the programming. They developed and carried out a realistic and extensive set of tests simulating a day with a trillion dollars of transactions. Clearly this is a system where reliability is crucial, both to individual customers and to the functioning of the economy.[32] Unfortunately, many software developers tend to skimp on the planning, specification, and design phases of a project; get quickly to the programming; then deliver the product with minimal testing. In fact, programming, or coding, is a relatively small part of a well-designed system.

A subfield of computer science focusing on design and development of safety-critical software is growing. Safety specialists emphasize that safety must be "designed in" from the start. There are techniques of hazard analysis that help system designers identify risks and protect against them. Software engineers who work on safety-critical applications should have special training. Software safety expert Nancy Leveson emphasizes that we can learn much from the experience of engineers in building safe electromechanical systems. "One lesson is that most accidents are not the result of unknown scientific principles but rather of a failure to apply well-known, standard engineering practices. A second lesson is that accidents will not be prevented by technological fixes alone, but will require control of all aspects of the development and operation of the system."[33]

Software developers need to recognize the limitations of software. As computers have become more capable, software monitoring and control of machinery have become more common. In Chapter 1 we mentioned several computer systems being developed to take over some of the tasks involved in driving a car. The risks of turning control over to computers must be weighed carefully. Most software today is simply not safe enough for safety-critical applications. Hardware safety mechanisms, as used by engineers in pre-computer systems, still have an important role; they should not be omitted without extremely strong justification.

User interfaces and human factors

Well-designed user interfaces can help avoid many computer-related problems. Principles and practices for doing a good job are known.[34] System designers and programmers need to learn from psychologists and human factors experts. As an illustration of some principles that can help build safer systems, consider automated flight systems. An expert in this area emphasizes the following points:[35]

- **The pilot needs feedback to understand what the automated system is doing at any time.** This is critical when the pilot must suddenly take over if the automation fails or must be turned off for any reason. One example is having the throttle move as a manually operated throttle would, even though movement is not necessary when the automated system is operating.
- **The system should behave as the pilot (or, in general, experienced user) expects.** Pilots tend to reduce their rate of climb as they get close to their desired altitude. On the McDonnell Douglas MD-80, the automated system maintains a climb rate that is up to eight times as fast as pilots typically choose. Pilots, concerned that the plane might overshoot its target altitude, made adjustments, not realizing that their intervention turned off the automated function that causes the plane to level out when it reaches the desired altitude. Thus because the automation behaved in an unexpected way, the airplane climbed too high—exactly what the pilot was trying to prevent. (The incidence of the problem was reduced with more training, but the human factors approach is to design the automation to suit the human, not *vice versa.*)
- **A workload that is too low can be dangerous.** Clearly, if an operator is overworked, mistakes are more likely. One of the goals of automation is to reduce the human workload. However, a workload that is too low can lead to boredom, inattention, or lack of awareness of current status information that might be needed in a hurry when the pilot must take over.

Redundancy and self-checking

Redundancy and self-checking are two techniques important in systems on which lives and fortunes depend. We already mentioned redundancy in the A320 airplane. Similarly, the space shuttle uses four identical but independent computer systems that receive input from multiple sensors and check their results against each other. If one computer disagrees with the other three, it is taken out of service. If one of the three remaining is judged by the other two to be faulty, it is taken out of service, and the rest of the flight is canceled. In case of a more serious problem, perhaps caused by a common flaw in the software, there is a fifth computer, made by another manufacturer and programmed by different programmers, that can control the descent of the shuttle.[36] This degree of redundancy is expensive and is not used in many applications, but it illustrates the kinds of precautions that can be taken for systems that operate in dangerous physical environments where human lives are at stake.

Complex systems can collect information on their own activity for use in diagnosing and correcting errors. After Chemical Bank's ATMs mistakenly doubled the amount of customers' withdrawals, the bank was able to correct the balances in the affected accounts. In Section 2.6 we mentioned that an audit trail (i.e., a record of access and modifications to a database) can help detect and discourage privacy violations. Audit trails are vital in financial systems. A detailed record of transactions helps protect against theft as well, and, as in this case, it helps trace and correct errors. The bank was able to prevent an inconvenience caused by a software bug from becoming a huge problem that could have cost customers and the bank (in lawsuits) millions of dollars.

AT&T's telephone system handles roughly 100 million calls a day. The very complex software for the system is developed and extensively tested by experienced programmers. The system is designed to constantly monitor itself and correct problems automatically. Half of the computing power of the system is devoted to checking the rest for errors. When a problem is detected in a switching component, the component automatically suspends use of the switch, informs the rest of the network that it is out of service temporarily and should not receive calls, activates recovery routines that take a few seconds to correct the problem, then informs the network that the component is functioning again. But wait a minute! This is the same system that failed a few years ago, disrupting phone service for

hours. In fact, it was this very part of the system that caused the breakdown. There was a bug in the routine that processed recovery messages from switches that had failed and recovered. The same software operated in each switch. Thus, each switch that received the message failed, then recovered and sent a recovery message. A chain reaction of failures occurred. The bug was in a software upgrade that had been running for about a month.[37] Even when the best professional practices are followed, even with extensive testing, we cannot be guaranteed that such complex systems do not have bugs.

Testing

It is difficult to overemphasize the importance of adequate, well-planned testing of software. Testing is not arbitrary; there are principles and techniques for doing a good job. Unfortunately, many programmers and software developers see testing as a dispensable luxury, a step to be skimped on to meet a deadline or to save money. This is a common, but foolish, risky, and often irresponsible attitude.

In his Challenger investigation, Richard Feynman concluded that the computer systems used on board the shuttle were developed with good safety criteria and testing plans. Ironically, he was told that because the shuttle software usually passed its tests, NASA management planned to reduce testing to save money. Fortunately, instead, as a result of studies done after the loss of the Challenger, NASA instituted a practice called independent verification and validation (IV&V).* That means that the software is tested and validated by a company other than the one that developed the program and other than the customer. (Testing and verification by an independent organization is not practical for all projects, but many software developers have their own testing teams that are independent of the programmers who develop a system.) The IV&V team acts as "adversaries" and tries to find flaws. After a few years, NASA planned to eliminate IV&V, but switched direction again. In response to several studies, including an extensive one done by software safety experts in 1992, NASA decided to make IV&V a permanent part of the program.[38] This example illustrates a common ambivalence about testing.

4.3.3 Law and Regulation

Criminal and civil penalties

Legal remedies for faulty systems include suits against the company that developed or sold the system and criminal charges when fraud or criminal negligence occurs. Families of Therac-25 victims sued; the suits were settled out of court. A bank won an $818,000 judgment against a software company for a faulty financial system that caused problems described as "catastrophic" by a user. A company that supplied critical parts for the Navy's F-18 jets pleaded guilty to routinely failing to perform required tests and falsifying test reports. The company was fined $18.5 million and may have to pay millions more in civil penalties.[39] The latter is not a computer-related case, but the issues and potential penalty would be similar if tests of a safety-critical computer system were falsified.

*The destruction of the Challenger was caused by seals that failed because of the cold weather, not by software error. Studies were done on many aspects of safety afterwards.

Many contracts for business computer systems limit the amount the customer can recover to the actual amount spent on the computer system. Customers know, when they sign the contract, that losses incurred because the system did not meet their needs for any reason are generally not covered. Such contract limitations have been upheld in court, and they should be. If people and businesses cannot count on the terms of a contract being upheld by the legal system, contracts would be almost useless; millions of business interactions that take place daily would become more risky and therefore more expensive. Because fraud and misrepresentation are not, or course, part of a contract, some companies that suffer large losses allege fraud and misrepresentation by the seller in an attempt to recover some of the losses, whether or not the allegations are firmly grounded.

Well-designed liability laws and criminal laws—not so extreme that they discourage innovation, but clear and strong enough to provide incentives to produce safe systems—are important legal tools for increasing reliability and safety of computer systems, as they are for other industries. After-the-fact penalties do not undo the injuries that occurred, but paying for mistakes and sloppiness is incentive to be responsible and careful. It compensates the victim and provides some justice. An individual, business, or government that does not have to pay for its mistakes and irresponsible actions will make more of them.

Unfortunately, liability law in the U.S. is very flawed. Multimillion dollar suits are often won when there is no scientific evidence or sensible reason to hold the manufacturer or seller responsible for accidents that occur with use of a product. Abuse of the liability lawsuit system virtually shut down the small airplane manufacturing industry in the U.S. It is difficult enough for jurors to evaluate the scientific evidence relating to silicone breast implants, for example. It will be at least as difficult for jurors to decide whether a bug in a computer program should have been detected, or whether it was responsible for an accident, or whether the damage was a risk the buyer must reasonably take. The newness and complexity of large computer systems make designing liability standards difficult, but this task needs to be done.

Regulation

Is there legislation or regulation that can *prevent* life-threatening computer failures? A law saying that a radiation machine should not overdose a patient would be silly. We know that it should not do that. No legislator or regulator knew in advance that that particular computer application would cause harm. We could ban the use of computer control for applications where an error could be fatal, but such a ban is ill advised. In many applications the benefits of using computers are well worth the risks.

A widely accepted option is regulation, possibly including specific testing requirements and requirement for approval by a government agency before a new product can be sold. The FDA has regulated drugs and medical devices for decades. Extensive testing, huge quantities of documentation, and government approval are required before new drugs and some medical devices can be sold. Arguments for such regulation, for both drugs and for safety-critical computer systems are the following:

- The profit motive may encourage businesses to skimp on safety; the government has a responsibility to prevent that from happening.

- It is better to prevent a bad product from being used than to rely on after-the-calamity remedies.
- Most potential customers and people who would be at risk (patients, airplane passengers) do not have the expertise to judge the safety or reliability of a system.
- It is too difficult and expensive for ordinary people to successfully sue large companies.

If the FDA had thoroughly examined the Therac-25 before it was put into operation, the flaws might have been found before any patients were injured. However, the weaknesses and trade-offs in the regulatory approach should be noted.[40]

- The approval process is extremely expensive and time-consuming. The delays caused by the regulation and requirements for government review cost many lives. In some cases companies abandon useful products because the expense of meeting FDA requirements is too high.
- Regulations that require specific procedures or materials discourage or prevent the use of newer and better ones that were not thought of by the people who wrote the rules.
- The goal of the regulation, be it safety, privacy, accuracy, less pollution, or whatever, tends to get lost in the details of the paperwork required. One writer on software safety commented, "The whole purpose of [following good software development techniques and documenting the steps] is to ensure that the necessary planning and design is performed, but regulatory agencies tend to focus on the visible products of the effort: the documents."[41]
- The approval process is affected by political concerns, including influence by competitors and the incentive to be overcautious. (Damage caused by an approved product results in bad publicity and possible firing for the regulator who approved it. Deaths or losses caused by the delay or failure to approve a good new product get little publicity.)

Leveson and Turner, in their Therac-25 article, summarize some of these dilemmas:

> The issues involved in regulation of risky technology are complex. Overly strict standards can inhibit progress, require techniques behind the state of the art, and transfer responsibility from the manufacturer to the government. The fixing of responsibility requires a delicate balance. Someone must represent the public's needs, which may be subsumed by a company's desire for profits. On the other hand, standards can have the undesirable effect of limiting the safety efforts and investment of companies that feel their legal and moral responsibilities are fulfilled if they follow the standards. Some of the most effective standards and efforts for safety come from users. Manufacturers have more incentive to satisfy customers than to satisfy government agencies.[42]

We have focused so far on legal approaches to protecting against business system failures and dangers in safety-critical applications. What about the problem of accuracy of information in databases maintained by businesses and government agencies? Detailed regulation of private databases is recommended by some privacy advocates. Most of the discussion above about liability, criminal negligence, and regulation applies as well to accuracy of private (business) databases. Achieving and maintaining accuracy in government databases is made difficult by the lack of market incentives for accuracy and the fact that the government can refuse to be sued. The government argues that it should not have to pay for mistakes, such as drug raids on the wrong house or problems caused by errors in government databases. Outside of government, we pay for carelessness. If you are playing ball in your backyard and accidentally throw the ball through a neighbor's window, you pay for the window.

Professional licensing

Another very controversial approach to improving software quality is licensing of software development professionals. Licenses are required by law for hundreds of trades and professions. Licensing requirements typically include specific training, passing competency exams, ethical requirements, and continuing education. The desired effect is to protect the public from poor quality and unethical behavior. The history of licensing in many fields shows that the actual goals and the effects were and are not always very noble. In some trades, particularly plumbing, the licensing requirements were devised to keep black people out. Economic analysis shows that the effect of licensing is to reduce the number of practitioners in the field and keep prices and income for licensees higher than they would otherwise be, in some cases without any improvement in quality. Some people consider licensing to be a fundamental violation of the freedom to work, that is, to offer one's services without needing the government's permission. These objections do not apply to voluntary approaches to measuring qualifications of software personnel. A diploma from a respected school is one measure. Certification programs by professional organizations, particularly for advanced training in specialized areas, can be useful.[43]

4.3.4 Taking Responsibility

Businesses

In some of the cases we mentioned, businesses made large payments to customers in compensation for problems or damages caused by computer programs. For example, Intuit offered to pay interest and penalties that resulted from the errors in its flawed income tax programs. Pepsi paid $10 million to customers who thought they had won its contest. A quick and voluntary decision to pay for damages is nothing new with computers of course. In the spring of 1994, jet fuel was accidentally mixed with fuel for propeller airplanes at several California airports. The improper fuel can cause engines to fail. Within weeks, the fuel company agreed to pay for overhaul of the affected engines. The cost of rectifying their mistake was estimated at $40–50 million. Their motivation may have been partly to reduce losses that would result from the inevitable liability suits or government fines. Other factors include recognition of the importance of customer satisfaction and the reputation of the business. We noted that business pressures are often a reason for cutting corners and releasing defective products. Business pressure can also be a cause for insistence on quality and maintaining good customer relations. Also, some businesses have an ethical

policy of behaving responsibly and paying for mistakes, just like the person who pays for breaking a neighbor's window.

Customer awareness

How can customers protect themselves from faulty software? How can a business avoid buying a seriously flawed program? How can a hospital protect its patients from dangerous systems?

The first step is to recognize and accept that complex computer systems are difficult to design and develop, and many will have flaws. For high-volume, consumer software, one can consult the many magazines that review new programs. Specialized systems with a small market are more difficult to evaluate before purchase. We can use a hint from another field where there seem to be some reliable and some questionable practitioners: home remodeling. We can check the company's reputation with the Better Business Bureau. We can get references (i.e., names of previous customers) and ask them how well the job was done. Online user groups for specific software products are excellent sources of information for prospective customers. In the case of the Therac-25, the users eventually spread information among themselves. If there had been an online Therac-25 user group at the time of the accidents, it is likely that the problems would have been identified sooner and some of the accidents would have been avoided.

4.4 PERSPECTIVES ON DEPENDENCE, RISK, AND PROGRESS

4.4.1 Are We Too Dependent on Computers?

A fire in March 1994 at a telephone switching facility in Los Angeles disrupted telephone service for half a day. Here are some of the effects on computer users, as reported in a newspaper article.[44]

- "More than 150,000 customers, cut off from the outside world, could barely function without their phones and modem-equipped computers."
- Fax machines were idled.
- Drivers could not buy gasoline with their credit cards. "Customers were really angry," said a gas station manager.
- Stockbrokers could not connect to New York by phone or computer.
- More than 1000 automated teller machines did not function; they use phone lines to connect to central computers.
- A travel agent said, "I can't get in to make reservations for our clients." The agency no longer uses printed airline schedules; it is "computer- and phone-dependent."
- "A California lottery spokesman said that 1200 of the state's 22,000 terminals were down." "The fire made it difficult for many to buy tickets or get their winnings."

The underlying problem here was the phone network. Redundancy and separate dedicated networks for certain applications might have reduced the problems, but the incident serves as a good reminder about how many ordinary daily activities are dependent on computer networks. A physician who specializes in medical information systems commented that "modern hospitals and clinics cannot function efficiently without them."[45] Modern crime fighting depends on computers. Some military jets cannot fly without the assistance of computers. Because of their usefulness and flexibility, computers are now virtually everywhere. Is this good or bad? Or neutral?

Are criticisms of "dependence on computers" fair?

Comments about our dependence on computers appear in many discussions of the social impact of computers. What do they mean? Often the word "dependence" has a negative connotation. "Dependence on computers" suggests a criticism of our society or of our use of computers. Is it appropriate? Several aspects of these criticisms are wrong and some are valid. Some misconceptions about dependence on computers come from a poor understanding of the role of risk, confusion of "dependence" with "use," and blaming computers for failures where they were only innocent bystanders. On the other hand, abdication of responsibility that comes from overconfidence or ignorance is a serious problem. Also, there are valid technical criticisms of dependence when a system is designed so that a failure in one component can cause a major breakdown.

"Dependence" or "use"?

> *Electricity lets us heat our homes, cook our food, and enjoy security and entertainment. It also can kill you if you're not careful.*
>
> –"Energy Notes," May 1994. (Flyer sent with San Diego Gas & Electric utility bills)

Hospitals and clinics cannot operate without electricity. We use electricity for lighting, entertainment, manufacturing—just about everything. In the early 1990s there were several disruptions of telephone systems and air traffic because of computer problems. In those same years there were several disruptions of telephone systems and air traffic because of electric power problems. The four-hour disruption of AT&T phone service and air traffic at the three major New York area airports in September 1991 was the result of batteries running down because a backup power generator was not properly connected. In January 1995, one of the New York area airports had to be closed after workers accidentally cut electrical cables, causing a power blackout.[46]

Is our "dependence" on computers different from our dependence on electricity. Is it different from a farmer's dependence on a plow? The Sioux people's dependence on their bows and arrows? Modern surgery's dependence on anesthesia? Computers and plows are tools. We use tools because we are better off with them than without them. They reduce the need for hard physical labor and tedious routine mental labor; they help us be more productive, or safer, or more comfortable. When we have a good tool, we may forget or no longer even learn the older method of performing a task. If the tool breaks down, we are stuck; we cannot perform the task until the tool is fixed. That may mean that no telephone calls get through for several hours. It may mean that a large amount of money is lost, and it may mean that people are endangered or die. But the negative effects of a breakdown do not condemn the tool. To the contrary, for many computer applications (not all), the inconveniences or dangers of a breakdown are a reminder of the convenience and productivity

provided by the tool when it is working, for example, of the billions of telephone calls (carrying voice, e-mail, files, and data) that are completed—that are made possible or more convenient or cheaper because of computers—each year. We saw that a bad computerized inventory system devastated some small businesses. On the other hand, thousands of businesses now use computerized inventory systems successfully.

We could avoid the risk of a broken plow by plowing with our hands. We could avoid the risk of losing a document file on a disk by doing all our writing by hand on paper. Even ignoring the possibility of a fire destroying our paper records, it should be clear that we do not choose the "safe," or nondependent option because, most of the time, it is less convenient and less productive. If one enjoys wilderness camping, as I do, one can observe how "dependent" we normally are on electric lights, refrigeration, and plumbing. That does not mean we should cook on camp stoves and read by firelight at home.

Risk

Things we thought were absolutely OK collapsed.

–An earthquake analyst, after the devastating Kobe Japan quake in 1995[47]

We trust our lives to technology every day. We trust older, noncomputer technologies every time we step into an elevator, a car, or a building. As the tools and technologies we use become larger, more complex, and more interconnected, the amount of damage that results from an individual disruption or failure increases, and the costs may be paid in dramatic and tragic events. If a person out for a walk bumps into another person, neither is likely to be hurt. If both are driving cars at 60 miles per hour, they may be killed. If two jets collide, or one loses an engine, several hundred people may be killed. However, the death rate per mile traveled is about the same for air travel as for cars.[48]

Most new technologies were not very safe when they were first developed. If the death rate from commercial airline accidents in the U.S. were the same now as it was 40 years ago, 8,000 people would die in plane crashes each year (instead of fewer than 200). Some early polio vaccines, in which the virus was not totally inactivated, caused polio in some children. We learn how to make improvements; problems are discovered and solved; scientists and engineers study disasters and learn how to prevent them. What has happened to the safety record in other technologies? The number of deaths from automobile accidents declined from 54,633 in 1970 to 43,536 in 1991 (while population, of course, increased). Why? Some significant reasons are increased education about responsible use (i.e., the campaign against drunk driving), devices that protect people when the system fails (seat belts and airbags), and improvements in technology, many of which use computers.* In the same period the rate of death from commercial airplane accidents declined from 0.8 per 100,000 people to 0.4 per 100,000—while the use of computers in airplanes increased.[49]

Risk is not restricted to technology and machines. It is a part of life. Sharp tools are risky. Someone living in a jungle faces danger from animals. A desert hiker faces rattlesnakes. Just as with complex technological systems, a person will be safer if he or she

*The 55 mph speed limit was not a significant factor, as it was widely ignored.

knows the risks and takes reasonable precautions. Just as with complex technological systems, the precautions are sometimes not enough.

We mentioned a few cases where delays in implementing computer systems cost millions of dollars. Delays in large construction and engineering projects (before the use of computers) were not uncommon either. Ask anyone who has written a book if it was completed on schedule!

Software safety expert Nancy Leveson says that the mistakes made in software are the same as those that used to be common in engineering. Over many years engineers have developed techniques and procedures to increase safety. Software developers need to learn from engineers and adapt their methods to software.

There are some important differences between computers and other technologies. Computers make decisions; electricity does not. The power and flexibility of computers encourages us to build more complex systems—where failures have more serious consequences. The pace of change in computer technology is much higher than for other technologies. Software is not built from standard, trusted parts as is the case in many engineering fields. The software industry is still going through its "growing pains"; it has not yet developed into a mature, fully developed discipline.

False charges—or, blaming the stove for a poorly cooked meal

As we have said several times already, computers are virtually everywhere. That means that when anything goes wrong, there is probably a computer that can be blamed, sometimes unfairly. I will mention just a few such examples that have appeared in other books.

In Holland, the body of a reclusive, elderly man who died in his apartment was not discovered until six months after his death, when someone noticed that he had a large accumulation of mail. This incident was described as a "particularly disturbing example of computer dependency." Many of the man's bills, including rent and utilities, were paid automatically, and his pension check was automatically deposited in his bank account. Thus "all the relevant authorities assumed that he was still alive."[50] But who expects the local gas company or other "relevant authorities" to discover a death? The problem here clearly was the lack of concerned family, friends, and neighbors. I happened to be present in a similar situation. An elderly, reclusive woman died in her home. Within *two days,* not six months, the mailman noticed that she had not taken in her mail. He informed a neighbor, and together they checked the house. I do not know if her utility bills were paid by computer; it is irrelevant.

Published collections of computer-related risks involving trains include cases of faulty brakes, operators disabling safety controls, a loose wire, and a confused driver who drove his train in the wrong direction during rush hour. Many of the cases involved human errors. We have seen that some human errors are the result of confusing or poorly designed computer systems, but that was not apparently the case in several of the reported incidents. An incident where a computer reportedly fell on a man's foot was listed as a health risk of computers.[51]

I mention these cases because accurate identification of the source of a problem is an important step to solving it. Including such incidents as risks of computers obscures the distinction between them and cases where computer systems *are* at fault—and where we must focus our attention to make improvements.

4.4.2 Making Trade-offs When Perfection Is Not an Option

We have emphasized through a number of examples that software is complex. We cannot be sure that all possible situations have been considered or that all bugs have been found and corrected. In fact, we can be more certain that flaws do exist in the system.

How close to perfection should we expect our systems to be? A water utility company sent a customer an incorrect bill for $22,000. A spokesman for the company pointed out that one incorrect bill out of 275,000 monthly bills is a good error rate. Is that reasonable? How accurate should the software for ATMs be? The double withdrawal incident mentioned in Chapter 1 affected roughly 150,000 accounts. With approximately eight billion ATM transactions each year, that is one error in roughly 45,000 transactions. Is that an acceptable rate? (There were probably other ATM errors in that year, but the publicity given this case suggests that it affected far more transactions than others.) How accurate should software for check processing be? 99%? 99.9%? Bank of America processes 17 million checks per day. Even if it made errors on 1000 checks every day, that would be an accuracy rate of better than 99.99%.[52]

At some point, the expense of improving a system is not worth the gain, especially for applications where errors can be detected and corrected at lower cost than it would take to try to eliminate them. How should the decision be made about how much to invest to make a system more reliable? For many applications, the decision is probably best left to the people responsible for the costs of the improvements and the costs of a failure (in liability and customer dissatisfaction).

For many applications that involve health and safety (with or without computers), though we may be more reluctant to accept it, there is also a point at which improvements to reduce risk are not worth the cost.[53] At this point, however, there are probably few, if any, safety-critical computer systems developers who are wasting money on "too much" safety.

4.4.3 Conclusions

We have made several points:

1. Many of the issues related to reliability for computers have arisen before with other technologies.
2. Perfection is not an option. The complexity of computer systems makes errors, oversights, and so on, a near certainty.
3. There is a "learning curve" for new technologies. By studying failures, we can reduce their occurrence.
4. Risks of using computers should be compared with risks of other methods and with benefits obtained.

This does not mean that computer errors and failures should be excused or ignored because failures occur in other technologies. It does not mean that carelessness or negligence should be tolerated because perfection is not possible. It does not mean that accidents should be excused as part of the learning process, and it does not mean that accidents should be excused because, on balance, the contribution of computers is positive.

I emphasize the similarities with failures and delays in other technologies (and nontechnological activities) to provide some perspective. Some critics of computers speak as if risks and system failures are new phenomena. I am not arguing that computer failures are less serious because other systems have problems too. The potential for serious disruption of normal activities and danger to people's lives and health because of flaws in computer systems should always remind the computer professional of the importance of doing his or her job responsibly. Computer system developers and other professionals responsible for planning and choosing systems must assess risks carefully and honestly, and include safety protections, appropriate plans for shutdown of a system when it fails, for backup systems where appropriate, and for recovery.

Knowing that one will be liable for the damages one causes is strong incentive to find improvements and increase safety. When evaluating a specific instance of a failure, we can look for those responsible and try to ensure that they bear the costs of the damage they caused. It is when evaluating computer use in a particular application area or when evaluating the technology as a whole that we should look at the balance between risks and benefits and compare the risks and benefits with those of noncomputerized alternatives.

4.5 EVALUATING COMPUTER MODELS

4.5.1 Models and Social Policy

Computer-generated predictions and conclusions about subjects of important social interest frequently appear in the news. Figure 4.2 lists a few examples.

Predictions that come from complex computer programs and expensive computers impress people. But the programs vary enormously in quality. Some are worthless, whereas others are very reliable. Predictions about problems like those in Figure 4.2 are used to justify multibillion-dollar government programs, restrictions on people's freedom of action, and regulations with significant impact on the economy and the standard of living of hundreds of millions of people. It is important for both computer professionals and the general public to have some idea of what is in such computer programs, where their uncertainties and weaknesses may lie, and how to evaluate their claims.

- When we will run out of a critical natural resource.
- Population growth.
- The cost of a proposed government program.
- The cost of waste disposal for juice boxes.
- The effects of second-hand smoke.
- The effects of a tax cut on the economy.
- The threat of global warming.
- When a big earthquake is likely to occur.

FIGURE 4.2: Some problems studied with computer models.

What are computer models?

A mathematical model is a collection of data and equations describing, or simulating, characteristics and behavior of the thing being studied. The models and simulations of interest to us here require so much data and/or computation that computers are needed to do the required computations. Computers are used extensively to model and simulate both physical systems, such as the design for a new airplane or the flow of water in a river, and abstract systems, such as parts of the economy.

Models allow us to investigate the possible effects of different designs, scenarios, and policies. They have obvious social and economic benefits: They enable us to consider alternatives and make better decisions, reducing waste, cost, and risk.

Although these models are abstract (i.e., mathematical), the meaning of the word "model" here is similar to its meaning in "model airplane." Models are simplifications. Model airplanes generally do not have an engine, and the wing flaps may not move. Models are not necessarily toys. In a chemistry class, we may use sticks and balls to build models of molecules, to help us understand their properties. A model of a molecule may not show the components of the individual atoms. Similarly, mathematical models do not include equations for every factor that may influence the outcome, or they may include equations that are simplified because the correct ones are unknown or too complicated. For example, a constant known as the acceleration of gravity can be used in equations to determine when an object dropped from a high place will hit the ground. The effect of wind may not be included in the equations, but on some days, wind could make a difference.

Physical models are not the same size as the real thing. Model planes are smaller; the molecule model is larger. In mathematical models, it is time rather than physical size that often differs from reality. Computations done on a computer to model a complex physical process in detail may take more time than the actual process takes. For models of long-range social phenomena, such as population growth, the computation must take less time than the real phenomenon for the results to be useful.

4.5.2 Evaluating models

Among three models developed to predict the change in health care costs that would result if the U.S. adopted a Canadian style national health plan, the predictions varied by $279 billion. Two of the models predicted large increases and one predicted a drastic decrease.[54] Why was there such a difference? There are both political and technical reasons why models may not be accurate. Political reasons, especially for this example, are probably obvious. Among the technical reasons,

- We may not have complete knowledge of the system being modeled. In other words, the basic physical or social science involved may not be fully understood.
- The data describing current conditions or characteristics may be incomplete or inaccurate.
- Computing power may be inadequate for the number of computations that would be needed if the full complexity of the system were modeled.

- It is difficult, if not impossible, to numerically quantify variables that represent human values and choices.

The people who design models decide what simplifications and assumptions to make.

Are reusable (washable cloth) diapers better for the environment than disposable diapers? When bans and taxes on disposable diapers were proposed, this controversy consumed almost as much energy as diaper manufacturing. Several computer models were developed to study the question. The particular kind of model used is called a life-cycle analysis; it attempts to consider the resource use and environmental effects of all aspects of the product, including manufacture, use, and disposal. To illustrate how difficult such a study may be, Figure 4.3 lists a few of the questions where assumptions were made by the modelers. Depending on the assumptions, the conclusions differed.[55]

- How many times is a cloth diaper used before it is discarded? (Values ranged from 90 to 167.)
- Should credit be given for energy recovered when waste is incinerated, or does pollution from incineration counterbalance the benefit?
- What value should be assigned for the labor cost of washing diapers?
- How many cloth diapers are used each time a baby is changed? (Many parents use two at once for increased protection.) The models used values ranging from 1.72 to 1.9.
- How should the pesticides used in growing cotton be counted?

FIGURE 4.3: Factors in diaper life-cycle modeling.

There are examples of highly publicized models where the simplifications and assumptions were questionable. Their impact on the results can be decisive. The TAPPS computer model for nuclear winter, popularized by Carl Sagan in the 1980s, predicted that millions of tons of smoke from a nuclear war would stay in the sky for months blocking sunlight, causing temperatures on earth to drop 15°C–25°C, causing crops and people to freeze. A critic of the model pointed out that it represented the earth as a smooth, oceanless ball, did not distinguish day and night, and did not include wind.[56]

For many scientific models, experiments can be done to test the model, or significant parts of it. For many of the issues of social and political interest, experiments are not possible or are limited to small parts of the model. Especially in such cases, and especially when a model is being used as an argument in a debate about government policy, it is helpful to know the state of knowledge about the subject and the assumptions made by the modelers.

The following three questions help us determine the validity and accuracy of a model.

1. How well do the modelers understand the underlying science or theory (be it physics, chemistry, economics, or whatever) of the system being studied? How well understood are the relevant properties of the materials involved? How accurate and complete are the data used?

2. Models necessarily involve simplifications of reality. What are the simplifications in the model?
3. How closely do the results or predictions of the model correspond with results from physical experiments?

The case studies in the next two sections illustrate our discussion of evaluating computer models and simulations. The topics are crash-analysis models used in the design of cars and climate models used to study global warming.

4.6 CASE STUDY: CAR CRASH-ANALYSIS PROGRAMS*

4.6.1 Background

Car crash-analysis programs gained wide usage by the late 1980s. One of the major programs, DYNA3D, was developed at Lawrence Livermore National Laboratory for military applications, but is now used in product design. The program models the interactions of physical objects on impact. DYNA3D is especially designed for high-speed collisions. It uses a technique called the finite-element method. A grid is superimposed on the frame of a car, dividing the car into a finite number of small pieces, or elements. The grid is entered into the program, along with data describing the specifications of the materials making up each element (e.g., density, strength, elasticity, etc.). Suppose we are studying the effects on the structure of the car from a head-on collision. Data can be initialized to represent a crash into a wall at a specified speed. The program computes the force, acceleration, and displacement at each grid point and the stress and strain within each element. These calculations are repeated to show what happens as time passes in small increments. Using graphics programs, the simulation produces a picture of the car at intervals after impact, as illustrated in Figure 4.4. To simulate 40–100 milliseconds of real time from the impact takes up to 35 hours of computer time on a supercomputer. Clearly, these programs require intensive computation.[57]

The cost of a real crash test can range from $50,000 to $800,000. The high figure is for building and testing a unique prototype for a new car design. The crash-analysis programs allow engineers to vary the thickness of steel used for selected components, or change materials altogether, and find out what the effect would be without building another prototype for each alternative. But how good are the programs?

4.6.2 Evaluating the Models

How well is the physics of car crashes understood?

Force and acceleration are basic principles; the physics involved in these programs would be considered fairly easy. The relevant properties of steel, plastics, aluminum, glass, and other materials in a car are fairly well known. However, although the behavior of the materials when force is applied gradually is well known, the behavior of some materials

*This section appeared in my chapter, "Social and Legal Issues," in *An Invitation to Computer Science* by G. Michael Schneider and Judith L. Gersting, West Publishing Co., 1995. (Used with permission.)

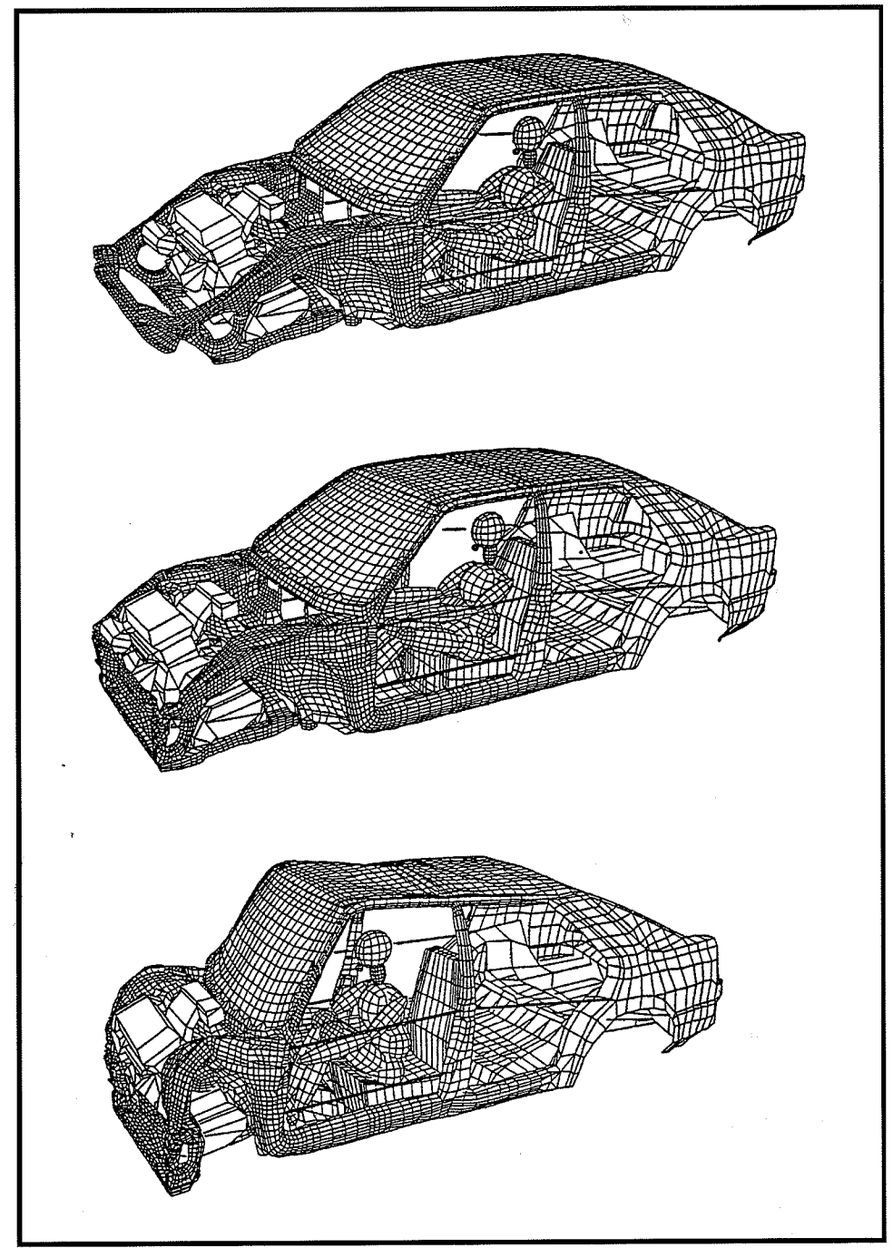


FIGURE 4.4: LS-DYNA3D simulation of a frontal Crash (35 mph before impact; 20 and 70 milliseconds after). (Reproduced by the permission of Livermore Software Technology Corporation.)

under abrupt acceleration, as in a high-speed impact, and their behavior near or at their breaking point are less understood. There are good data on the density, elasticity, and other characteristics of materials used in the model.

What simplifications are made in the programs?

The grid pattern is the most obvious; a car is smooth, not made up of little blocks. Also, time is continuous; it does not occur in steps. The accuracy of the simulation will depend in part on how fine the grid is and how small the time intervals are. As computer speeds increase, we can do more precise computation. In the early 1990s, crash-analysis programs used roughly 10,000–50,000 elements and updated the calculations for time intervals of one millionth of a second.

How do the computed results compare to actual crash tests on real cars?

How are such comparisons performed? The real tests are taped by high-speed cameras. Various kinds of sensors, such as strain gauges, are attached to the car, and reference points are marked on the frame. The tapes can be visually compared with the computer output. The values recorded by the sensors are compared with values computed by the program, and the distortion or displacement of the reference points can be physically measured and compared to the computed positions. From the results of the physical crash, elementary physics can be used to calculate backward and determine the deceleration and other forces acting on the car. These can be compared to the values computed in the simulation. The conclusion? The crash-analysis programs do an extremely good job. Results from the program correspond very closely to data collected in actual test crashes.[58]

Once we know that the models are reasonably accurate, we can conclude that the computer programs provide some benefits we cannot get from physical testing. The computer can provide more data than the sensors and can compute what is happening in areas of the car that the cameras cannot see. The simulation can provide more information than a real crash if there is unexpected damage in a position where few sensors were placed.

4.6.3 Uses of the Crash-analysis Models

Car crash-analysis programs are replacing physical crash testing as a design tool for new cars. The crash test is used as confirmation and is required by the federal government. Should the simulation results replace the physical crash? There can be many answers to this question, depending on its context. Suppose the government did not require a physical crash test. Would you buy a car that had been certified crashworthy only by a computer? To decide whether or not to do physical crash tests, a car manufacturer would probably consider the accuracy of the models, the costs of physical testing, liability laws, and public relations. A company that provides liability insurance for car manufacturers would consider whether the simulations are reliable enough for them to do accurate risk analysis. A legal scholar or an economist might consider whether the law should specify a specific test or focus on rules for liability—letting the manufacturer decide on the best way to ensure that its cars are safe. A legislator may consider the reliability of the simulations, public attitudes about computers, and the arguments of lobbyists. In fact, engineers who work with the crash-analysis programs do not believe that physical crashes will or should be eliminated. They remind us that the simulation is an implementation of theory. The program may give

poor results if it is used by someone who does not understand it well. Results may be poor if something happens that the program simply was not designed to consider. Overall, the crash-analysis programs are excellent design tools that enable increases in safety with far less development cost. But the real crash is the proof.

The DYNA3D program, and some variations of it, are used in a large variety of other impact applications. Some are listed in Figure 4.5. One reason for its wide use is the increase in computing power and the declining cost. In the late 1970s serious engineering applications of DYNA3D were run on $10 million computers. Now, many applications can be run on workstations in the $20,000–$60,000 range. Another reason is the confidence that has developed over time in the validity of the results.

- To predict damage to a hazardous waste container if dropped.
- To predict damage to an airplane windshield or nacelle (engine covering) if hit by a bird.
- To determine whether beer cans would get dented if an assembly line were speeded up.
- To simulate a medical procedure called balloon angioplasty, where a balloon is inserted in a blocked artery and inflated to open the artery. The computer program helps researchers determine how to perform the procedure with less damage to the arterial wall.
- To predict the action of airbags and the proper location for sensors that inflate them.
- To design interior parts of cars to reduce injuries during crashes (e.g., the impact of a steering wheel on a human chest).
- To design bicycle and motorcycle helmets to reduce head injuries.
- To design cameras to reduce damage if dropped.
- To forecast effects of earthquakes on bridges and buildings.

FIGURE 4.5: Other uses of DYNA3D and related programs.

4.7 CASE STUDY: CLIMATE MODELS AND GLOBAL WARMING*

4.7.1 Background

In the late 1980s, the news was full of reports of impending global warming caused by the human-induced increase of carbon dioxide (CO_2) and other greenhouse gases in the atmosphere. Some scientists predicted that we might see the warming within a decade. Global warming predictions are based on computer models of climate. In this section we will look at the computerized models. First we need a little background.

*This section appeared in my chapter, "Social and Legal Issues," in *An Invitation to Computer Science* by G. Michael Schneider and Judith L. Gersting, West Publishing Co., 1995. (It has been slightly revised. Used with permission.)

The earth is warmed by solar radiation. Some of the heat is reflected back; some is trapped by gases in the atmosphere. The latter phenomenon is known as the greenhouse effect. Without it, the temperature on the earth would be too cold to support life. The main "greenhouse gases" are water vapor, carbon dioxide (CO_2), methane, chlorofluorocarbons (CFCs), and ozone. Among those whose concentration has been increased by human activity, CO_2 is the most important. The problem of concern now is that this increase may enhance the greenhouse effect significantly, increasing temperatures and causing other major climate changes.

CO_2 currently makes up roughly one-thirtieth of one percent of the atmosphere, or 355 parts per million (ppm) by volume. This is substantially higher than for most of the past 160,000 years.* An upward trend of both CO_2 and methane began roughly 16,000 years ago. However, since the beginning of the Industrial Revolution, concentrations have been increasing at a faster rate. Since 1950 the climb has been very steep. The main source of increased CO_2 is the burning of fossil fuels (e.g., oil and coal).

The computer models used to study climate are called general circulation models (GCMs). GCMs were developed from atmospheric models that have been used for a long time for weather prediction. They are quite complex. They contain information about the sun's energy output; the orbit, inclination, and rotation of the earth; geography (a map of land masses); topography (mountains, etc.); clouds; sea and polar ice; soil and air moisture; and a large number of other factors. Like the crash-analysis models, the GCMs use a grid. The grid circles the earth and rises through the atmosphere. The computer programs solve equations for each grid point and element (grid box) for specified time intervals. The equations simulate factors such as atmospheric pressure, temperature, incoming solar energy, outgoing radiant energy, wind speed and direction, moisture, and precipitation. For global warming studies, the atmospheric models are combined with models of the oceans that include temperature, currents, and other factors. The modeling programs are generally run to compute the effects of doubling CO_2 concentration from its approximate level at the beginning of the 20th century. (Current trends suggest that CO_2 concentration will double sometime in the 21st century.) More recent studies include other greenhouse gases as well.

Because of the global importance of potential climate changes, the Intergovernmental Panel on Climate Change (IPCC), sponsored by the United Nations and the World Meteorological Organization, published several major reports on the scientific assessment of climate change. The reports were prepared and reviewed by several hundred scientists worldwide. They are considered an authoritative review of the state of scientific knowledge about climate change. They are the main references used for this discussion.[59]

4.7.2 Evaluating the Models

How well is the science of climate activity understood? How complete and accurate are the data?

The climate system is composed of five subsystems: atmosphere, oceans, cryosphere (ice packs, snow), geosphere (land, soil), and biosphere (vegetation). Effects in the air (such as changes in temperature, wind speed and direction, etc.) can be computed using

*The past data come from measurements of gases trapped in ice cores drilled in Antarctica and Greenland.

well-understood principles of physics. The laws of physics are used to compute temperature and other effects in the oceans as well. However, there is still a large amount of scientific uncertainty about each component.

The oceans have a large capacity to absorb heat. The circulation of water in the oceans, due to currents, affects heat absorption. Surface currents are fairly well known; deep currents are less well known. Not enough is known about the exchange of energy between the oceans and the atmosphere.

Clouds are extremely important to climate. Many processes involved with the formation, effects, and dissipation of clouds are not particularly well understood.

Some greenhouse gases have cooling effects as well as warming effects. Although the IPCC tried to quantify these effects in 1990, by 1992, IPCC scientists decided the figures published in 1990 were likely to be in substantial error, and they did not yet know enough to give new numerical values.[60]

If temperatures rise, natural wetlands give off more methane, contributing to more warming. This is called positive, or destabilizing, feedback: An output or side effect of the process amplifies the original effect. There are negative, or stabilizing, feedbacks too. When the earth warms, more water evaporates and forms clouds; the clouds reflect some of the sun's heat back up away from the earth. (Clouds have positive feedback effects also.) There are many such feedback mechanisms affecting climate. For some, it is not known if the feedback is positive or negative. Many uncertainties in the models are attributed to lack of knowledge or lack of full representation of feedback mechanisms.

Another area of uncertainty is natural climate variation. The earth has experienced ice ages and warm periods. There is also a lot of year-to-year fluctuation. We do not know all the causes of the changes, and we do not know the current natural temperature trend.

The IPCC reports make several references to the need for more data on many climate factors including clouds, ocean currents, and the ozone layer. Research programs costing millions of dollars are in progress to collect such data and data on other factors that are not considered in current models.[61]

What simplifications are made in the models?

There are about half a dozen major climate modeling centers in the world. The models vary in many ways, and they are being modified and improved as time passes. Many of our comments are about the models used in the 1980s—those on which fears of catastrophic global warming were based. Some of our comments are generalizations; they may not all be true of all the models at any one time.

The grid points in the models are typically spaced about 500 kilometers (roughly 300 miles) apart. The state of California, for example, may be represented by half a dozen grid points. Islands as small as Japan and England may not appear on the "map" at all, as they may lie between grid points in an ocean or sea. See Figure 4.6 for a sample map. The grid is coarse because the computing time required to run the programs depends on the number of points. Recall that it takes up to 35 hours of computer time to simulate 40–100 milliseconds of a car crash. To be useful, climate studies must be done faster than real time. A 500-kilometer grid, rising ten layers in the atmosphere, has roughly 20,000 points. Because of the grid size, small storms and other small phenomena fall between the grid points and either are not fully represented or are expanded to fill a cell of the grid.

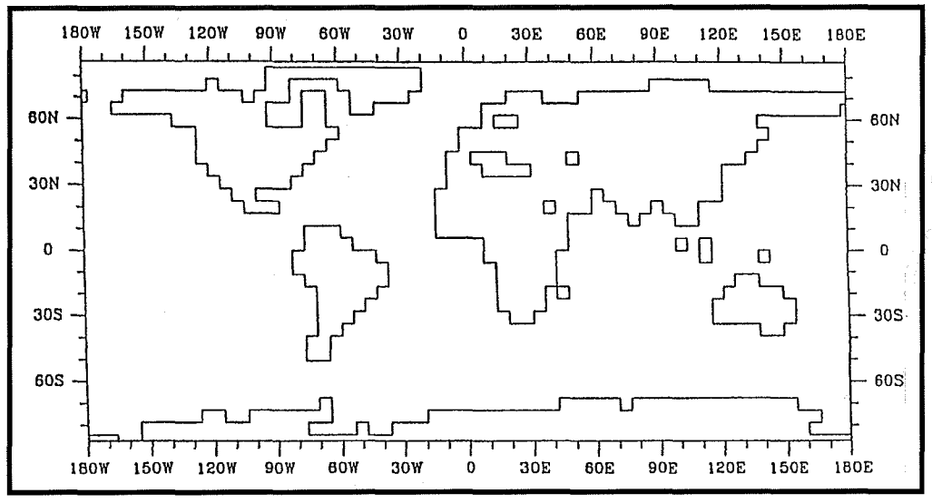


FIGURE 4.6: The land map for a typical climate model. (Reprinted with the permission of Dr. Ulrich Cubasch [Deutsches Klimarechenzentrum].)

Some models do not distinguish between day and night. This simplification is significant because temperature records for some areas of the northern hemisphere show increases in the nighttime winter lows and decreases in the daytime summer highs, which could be a benign or even beneficial form for an average warming to take.[62] The models include many other simplifications, for example, representing only the top layer of the oceans.

Ideally, all of the processes that affect climate would be represented in the GCMs by equations derived from the underlying science (generally, physics and some chemistry). This is not possible because it would require too much computation time and because all the underlying science is not known. Values of some variables are selected to make the models predict present conditions with reasonable accuracy. The process of modifying the values until the desired result is achieved is called "tuning" the model. Although models can be tuned by balancing the values for several variables so that the model accurately describes present conditions, it is not known if that choice of values will predict future conditions accurately.

How closely do the results of the computer programs correspond to results from physical experiments?

We cannot do physical experiments for global warming. Thus we need to consider other ways to assess the results of the computer simulations.

Predictions from the models for average global temperature increase range from a little more than 1°C to a little more than 5°C. The wide variation in the predictions of the models, by itself, suggests that we are far from having a clear understanding of climate behavior and the impact of the greenhouse gases. For a well-understood phenomenon, the predictions of good models would show more agreement.

Scientists gain confidence in the models because they predict seasonal variations and other broad-scale phenomena. The general patterns of predictions by different models are similar. For example, they all predict warming, and they all predict that more of the warming would take place near the poles and in winter.

The most obvious test that could be done to validate the models is to run them for conditions of the past century, including the increase in greenhouse gases that has occurred so far, to see if they predict the current temperature. Unfortunately, because of insufficient data about past conditions and for other technical reasons, the validity of the experiments that have been done is not clear. We will describe and analyze the results briefly.

As we mentioned earlier, concentrations of greenhouse gases in the atmosphere have increased very steeply since about 1950. Between the late 19th century and 1990, the average temperature rose by 0.3°C–0.6°C. Most of the warming occurred before 1950. Some of the climate models predicted temperature increases three to five times as high as actually occurred. Evaluating these results is not a simple task. There are many factors that affect temperature. An increase in aerosols (airborne particles) in the atmosphere, from volcanic eruptions and industrial activities, contribute to cooling. Urbanization, and other human activities besides emission of greenhouse gases, caused some warming. Still other factors, such as variation in solar output and other natural phenomena, could have affected the temperature in either direction. Thus, it is not known how much of the warming over the past century is attributable to greenhouse gases. The magnitude and rate of the temperature change in the past century may not be unusual; it is possible that all of it is within the range of natural fluctuation. Considering all the uncertainties and the overly high predictions of the models, IPCC scientists estimate that if all the warming that occurred was due to greenhouse gases, then the likely future warming would be 1°C–2°C, that is, at the lower end of the range predicted by the GCMs.

The IPCC report predicts a global mean temperature increase of 0.2°C–0.5°C per decade. Satellite data show an average global temperature rise of only 0.06°C for the decade of the 1980s. Although the models predict very high warming near the poles, arctic temperatures in some regions have gone down over the past 50 years.

Another way to try to understand the accuracy of GCM predictions is to consider how sensitive they are to modifications and improvements. There is very much scientific uncertainty about the behavior of clouds and their feedback properties. One modeling center did an experiment where they ran the same model except that clouds were treated differently each time. Their predictions for temperature increase ranged from 1.9°C to 5.2°C, more than a 170% difference between the lowest and highest predictions.

Newer models, using more complete representations of the oceans, and/or including some of the cooling effects of industrial activity, show substantially reduced or slower warming. In 1995, as a result of improvements in the models, the IPCC reduced its prediction of the global average temperature increase.[63]

Conclusions

GCMs have improved dramatically in the few decades that scientists have been developing and working with them. Increased computer power allows more experiments, better calibration, and increased resolution (smaller grid size). Increased data collection and basic science research improve our understanding of the behavior and interactions of climate system components. But given results so far and the complexity of climate phe-

nomena, it seems fair to conclude, as an IPCC report comments, that "climate models are still in an early stage of development."[64] They seem to overestimate the amount of warming that will occur. The 1992 IPCC update says, "Since the 1990 report there has been a greater appreciation of many of the uncertainties which affect our predictions of the timing, magnitude and regional patterns of climate change."[65] The models are a tool for understanding climate change, but are not yet at a stage where we can have a lot of confidence in the precision of their results. Certainly they have not achieved the level of reliability of the car crash-analysis models.

EXERCISES

Review exercises

1. List several cases described in Sections 4.1–4.4 where insufficient testing was a factor in a program error or failure.
2. List several cases described in Sections 4.1–4.4 where the provider did an inadequate job of informing customers about flaws in the system.
3. The Therac-25 radiation machine involved errors in software, overall design, and management' or operations. Describe one error of each type.
4. List the three questions we used to evaluate computer models.
5. What is one simplification in the car crash models? In the climate models?

General exercises

6. Describe two or three computer errors or system failures you had heard about before reading this chapter (preferably ones that are not in the chapter).
7. Many records stored on computers include a date, and some older software systems use two digits to represent the year (e.g., 78, 95). Pick any two applications (e.g., billing for telephones, driver's license renewal), and describe problems that might occur in the year 2000.
8. **(a)** Suppose you write a program to add two integers. Assume that both integers and their sum will fit in the standard memory unit used by the computer for integers. How likely do you think it is that the sum will be correct? (If you used the program a million times on different pairs of integers, how many times do you think it would give the correct answer?)
   (b) Suppose a utility company has a million customers and it runs a program to determine if any customers have overdue bills. How likely do you think it is that the results of the program will be correct?
   (c) Probably your answers to parts (a) and (b) were different. (They should be!) Give some reasons why the likely number of errors would be different in these two examples.
9. If you have access to Usenet, read a few recent postings in comp.risks, the Risks Forum news group. Write a summary of two items.
10. Consider the case described in Section 4.1.2, in which a boy was assumed to be a drug abuser because two schools used different disciplinary codes in their computerized records.
    (a) Is this kind of problem more likely to occur with computerized records than with paper records? Why, or why not?
    (b) Describe some policies or practices that can help prevent such problems.

11. In order to keep illegal immigrants and foreign visitors from working in the U.S., the federal government is considering requiring that that every job applicant be checked against a national database before being hired. One U.S. senator commented, "Over the years we've heard many complaints about the accuracy of the INS [Immigration and Naturalization Service] database. Isn't it problematic to rely on a faulty database for verification of employment authorization?" Another replied, "I do not think it is problematical to come up with a registry that is going to have the verifiable information that we're looking for" if the government is willing to spend the money to improve it.[66]

    Discuss the strength of both arguments concerning the probable reliability of the database. Describe some potential consequences of inaccuracies.
12. For a long time, it has been possible to make airplane reservations by phone; tickets were mailed to the customer. Now, several airlines have a new system that does not use paper tickets. A customer makes a reservation by phone, gets a confirmation number, and just shows a picture ID, such as a driver's license, at the airport gate to board the plane.
    (a) What is the role of computers in making ticketless service possible?
    (b) What is one advantage of this service to the customer? To the airline? To society in general?
    (c) Describe two potential problems that could occur with this service.
13. Suppose you are responsible for the design and development of a computer system to control an amusement park ride. The operator will enter the number of people on the ride and what seats they occupy, so the software can consider weight and balance. The system will control the speed and time of the ride. The amusement park wants a system where, once the ride starts, a person is not needed to operate it.

    List some important things that can or should be done to ensure the safety of the system. Consider all aspects of development, technical issues, operating instructions, and so on.[67]
14. The Strategic Defense Initiative of the 1980s was a proposal for a computer system to detect a nuclear attack and automatically launch weapons to destroy the incoming missiles. Discuss some potential risks of such a system.
15. Who are the "good guys"? Pick two people or organizations mentioned in this chapter whose work helped make computer systems safer or reduced the negative consequences of errors. Tell why you picked them.
16. Most consumer and business software packages are sold with a statement that the seller is not liable for damages caused by the use of the software. Some people advocate a law requiring software vendors to provide stronger warranties. Discuss the pros and cons of such a requirement. Consider the likely effects on and/or relevance of quality, price, competition, and freedom of contract.
17. There is a story that a major retail company "lost" a warehouse from its inventory computer system for three years. No goods were shipped to or from the warehouse. Payroll was handled by a separate system, so the employees continued to be paid. To what extent is this a computer failure? What other important factors are involved? Why would such an incident be less likely without computers? Is this incident a strong argument against using computerized inventory systems?
18. Theft of expensive home appliances (e.g., TVs, stereo systems) is not new. In what ways is the impact on the victim of the theft of a home computer more serious?
19. During the Gulf War, a commander in the Royal Air Force left disks containing military plans for Desert Storm in his car. They were stolen—apparently by ordinary thieves who took his computer, not by spies. One book describes this as a "particularly disturbing example of computer dependency."[68] The point of this exercise is to analyze how much difference it would

have made if the plans were in a paper file folder. Give reasons for your answers to each of the questions.

Do you think the commander would have been more likely or less likely to leave paper files in his car? Would the thieves have been more or less likely to recognize what the files contained if they had been on paper? Would the thieves have been likely to panic and discard the files, return them, or sell them to Iraq? Was the commander more or less likely to face a court martial if the files were on paper? Overall, how much do you think computers have to do with the essential issues in this case?

20. The robot cow-milking machines mentioned in Chapter 1 have some problems. The machines are twice as expensive as conventional milking machines that must be attached by hand. They have software glitches. The system alerts the farmer by beeper when there is a problem. He is beeped several times a day, often for false alarms. The machines have trouble milking cows with unusual-sized udders. Some cows kick the machines. About 10% of the cows can't use the machine. In the summer, when cows are at pasture, they must be herded into the barn to use the machine.[69]

Evaluate these problems. Which are serious? Which are likely to be solved? Which are not really problems caused by the machine?

21. Write five questions whose answers would be needed in a life-cycle analysis model comparing the environmental impact of juice boxes with the environmental impact of juice in bottles. (Use the questions for diapers in Figure 4.3 as a guide.) Consider manufacture, transportation, use, and disposal.

22. Suppose the government no longer required physical crash tests for new cars. What factors would you consider in deciding whether or not to buy a car that was crash tested only by computer simulation? What magazines or technical literature would you consult, if any, in making your decision?

23. We suggested that different people and institutions would consider different factors when deciding whether to require physical crash testing of cars. Which group do you think would rely most heavily on technical information about the quality of the computer simulation programs: customers, car manufacturers, companies that insure the manufacturers, legal scholars, or legislators? Why?

24. An article in the magazine *Audubon*[70] states that "Since the 1960s more than 100 separate studies have confirmed that a doubling of the CO_2 concentration would raise average surface temperatures by one to four degrees centigrade." Is this an accurate statement? Explain your answer.

25. Suppose there are three companies that make a similar product. One does physical tests to check the safety of the product. One uses only computer simulations. The third will not disclose the testing methods it uses. Suppose you are on a jury for a case where someone is suing one of the companies because of an injury received from the product.

(a) Would your decision about awarding money to the plaintiff be affected by the company's policy about testing? If so, how? If not, what factors are more important to you?

(b) Suppose reliable data show that the injury rate for the product is almost identical for all three companies. With that additional information, would your decision be affected by the company's testing policy?

26. Which of the following models do you think produce very accurate results? Which do you think are less reliable? Give your reasons.

- Models that predict the effect of an income tax change on government revenue.
- Models that predict the position of the moon.
- Models that predict the speed of a new racing boat hull design under specified wind conditions.*

Assignments

These exercises require some research or activity that may need to be done during business hours or a few days before the assignment is due.

27. Find newspaper or magazine articles about the more than 500-point drop in the stock market that occurred in October 1987. What was the role of computer programs in this incident?

28. Suppose you have the following data:

- The number of tons in the known reserves of an important natural resource like, say, copper.
- The average amount of the resource used per person (worldwide) per year.
- The total population of the world.

(a) Write a program, using a programming language or a spreadsheet, to determine in how many years the resource will run out. The program's input should be the three data described above.

(b) One obvious flaw in the program is that it assumes the population is constant. Include the rate of population increase per year as another input.

(c) Suppose your program is correct and the input data are reliable. List all the reasons you can think of why this program is really not a good predictor of when we will run out of the resource.

(d) In 1972, a group called the Club of Rome received a lot of attention when it published a study using computer models that implied that the world would run out of several important natural resources in the 1980s. Today, many of those resources are cheaper than they were then, indicating that they are now less scarce. Why do you think so many people accepted the predictions in the study?

28. Find two articles about the TAPPS nuclear winter model from newspapers or magazines in the mid-1980s (try 1984–1986). (Do not use either of the two mentioned in the endnotes for Section 4.5.) What information, if any, do they provide about the assumptions, simplifications, and limitations of the model?

Class exercises

1. Assume that the family of one of the victims of the Therac-25 is suing the hospital where the machine was used, AECL (the maker of the machine), and the programmer who wrote the Therac-25 software. Divide students into six groups: attorneys for the family against each of the three respondents and attorneys for each of the three respondents. Each group is to present a five-minute summation of arguments for its case. Then, let the class discuss all aspects of the case and vote concerning which if any of the respondents should pay the family and whether any should be charged with a criminal offense.

2. Consider the following scenario. A state's highway patrol keeps records of stolen cars in its computer system. A car can be checked by typing in the license plate number. The records are not routinely updated when stolen cars are recovered. A car was still listed as stolen a few years after it had been recovered and later sold. The new owner of the car was shot and killed by a police officer during a traffic stop; the officer thought the car was stolen and that the driver was acting suspiciously. An investigation concluded that the officer "acted in good faith." The family

*Extensive computer modeling is used to design boats for the America's Cup races.

has filed a wrongful death suit against the highway patrol and the police officer. Divide the class into teams of attorneys for the family, the highway patrol, and the officer. Each team is to present a five-minute summation of arguments for its case. Then, let the class vote concerning which, if any, of the respondents should pay the family and whether anyone should be charged with a criminal offense.*

3. Poll the class and find out how many students have tried hang gliding or bungee jumping. How many say "No way!"? How many would like to be one of the first people to buy a new pocket-sized computer that has novel new features (say, the user communicates with it by speaking to it in normal conversational English)? How many would wait six months or so to see how well it really works? How many would ride on a computer-controlled subway train that had no human driver?

Generate a discussion of personal differences in risk-taking. Is there one correct level of risk?

NOTES

1 Sources for some of the billing error cases include Philip E. Ross, "The Day the Software Crashed," *Forbes,* April 25, 1994, 153:9, pp. 142–56; and Peter G. Neumann, "Inside Risks," *Communications of the ACM,* July 1992, p. 122.

2 I learned of the mortgage problem because I was one of the victims. Sources for the other credit database accuracy cases include Jeffrey Rothfeder, *Privacy For Sale,* Simon & Schuster, 1992, p. 34 and pp. 130–31; "A Case of Mistaken Identity," *Privacy Journal,* December 1992, p. 7; "In the States," *Privacy Journal,* January 1993, p. 3; Neumann, "Inside Risks," *Communications of the ACM,* January 1992, p. 186, and July 1992, p. 122.

3 Associated Press, "Teen 'Convicted' by Computer," *San Jose Mercury,* March 7, 1996, p. 3B.

4 "Who's Reading Your Medical Records?" *Consumer Reports,* October 1994, pp. 628–32. "How Private Is My Medical Information?" Fact Sheet No. 8, Privacy Rights Clearinghouse, Center for Public Interest Law, University of San Diego.

5 Dan Joyce, e-mail correspondence, May 17, 1996 (the adoption case). Study by the Office of Technology Assessment, reported in Rothfeder, *Privacy For Sale.* "Jailing the Wrong Man," *Time,* February 25, 1985, p. 25. David Burnham, "Tales of a Computer State," *The Nation,* April 1983, p. 527. Evelyn Richards, "Proposed FBI Crime Computer System Raises Questions on Accuracy, Privacy," *Washington Post,* February 13, 1989, p. A6. "Wrong Suspect Settles His Case for $55,000," *New York Times,* March 6, 1998, p. 30. Peter G. Neumann, "Risks to the Public in Computer and Related Systems," *Software Engineering Notes,* April 1988, 13:2, p.11. Several similar cases are reported by Peter G. Neumann in "Inside Risks," *Communications of the ACM,* January 1992, p. 186.

6 Joan E. Rigdon, "Buggy PC Software Is Botching Tax Returns," *Wall Street Journal,* March 3, 1995, pp. B1, B4.

7 Sources about the Pentium chip and other bugs include various Internet postings and the following articles, all from the *Wall Street Journal:* Don Clark, "Some Scientists Are Angry over Flaw in Pentium Chip, and Intel's Response," November 25, 1994, p. B6; Don Clark, "Intel's Grove Airs Apology for Pentium over the Internet," November 29, 1994, p. B6; Jim Carlton and Stephen Kreider Yoder, "Humble Pie: Intel to Replace Its Pentium Chips," December 21, 1994, pp. B1, B8; Scott McCartney, "Compaq Recalling Notebook Computer From Dealers in Europe to Repair Bug," December 22, 1994, p. B2.

*I have changed some details, but this scenario was suggested by a real case.

8 Thomas Nicely, quoted in Carlton and Kreider Yoder, "Humble Pie."

9 Frederick Rose and Richard Turner, "The Movie Was a Hit, The CD-ROM a Dud; Software Bites Disney," *Wall Street Journal,* January 23, 1995 pp. A1, A6. The comments are from Mike Maples, Microsoft, in Joan E. Rigdon, "Frequent Glitches in New Software Bug Users," *Wall Street Journal,* January 18, 1995, pp. B1, B5.

10 "AT&T Phone Failure Downs Three New York Airports for Four Hours," *Software Engineering Notes,* October 1991, 16:4, p. 6–7.

11 "AT&T Crash, 15 Jan 90: The Official Report," in "Subsection on Telephone Systems," *Software Engineering Notes,* April 1990, 15:2, p. 11–14.

12 "Subsection on Telephone Systems," *Software Engineering Notes,* April 1990, 15:2, pp. 11–14. Philip E. Ross, "The Day the Software Crashed," *Forbes,* April 25, 1994, 153:9, pp. 142–56. Ann Lindstrom, "Outage Hits AT&T in New England," *Telephony,* November 11, 1991, 221:20, p. 10. "Data Entry Typo Mutes Millions of U.S. Pagers," *Wall Street Journal,* September 27, 1995, p. A11.

13 David Craig, "NASDAQ Blackout Rattles Investors," *USA Today,* July 18, 1994, p. 2B. Associated Press, "NASDAQ Defends Its System after Stock-Pricing Errors," *New York Times,* September 13, 1994, p. D19. "Note to Readers," *Boston Globe,* October 25, 1994, p. 52.

14 *Science News,* September 13, 1986, p. 172. Philip E. Ross, "The Day the Software Crashed," *Forbes,* April 25, 1994, 153:9, pp. 142–56.

15 Tamala M. Edwards, "Numbers Nightmare," *Time,* August 9, 1993, p. 53.

16 Sources for the inventory program problems include Thomas Hoffman, "NCR Users Cry Foul over I series Glitch," *Computerworld,* February 15, 1993, p. 72; Milo Geyelin, "Faulty Software Means Business for Litigators," *Wall Street Journal,* January 21, 1994, p. B1; Milo Geyelin, "How an NCR System for Inventory Control Turned into a Virtual Saboteur," *Wall Street Journal,* August 8, 1994, p. A1; Mary Brandel and Thomas Hoffman, "User Lawsuits Drag On for NCR," *Computerworld,* August 15, 1994, p. 1.

17 William M. Bulkeley and Joann S. Lublin, "Ben & Jerry's New CEO Will Face Shrinking Sales and Growing Fears of Fat," *Wall Street Journal,* January 10, 1995, p. B1. The Walgreen case is in Ross, "The Day the Software Crashed," p. 146. Bruce Campion-Smith, "Glitches Stalling Updated Airport Radar," *Toronto Star,* August 3, 1992, p. A1. David Hughes, "ATC, Airport Upgrades Poised for New Growth," *Aviation Week and Space Technology,* April 5, 1993, 138:14, p. 40.

18 Carl Ingram, "DMV Spent $44 Million on Failed Project," *Los Angeles Times,* April 27, 1994, p. A3. Effy Oz, "When Professional Standards Are Lax: The CONFIRM Failure and its Lessons," *Communications of the ACM,* October 1994, 37:10, pp. 29–36.

19 The DIA delay was widely reported in the news media. A few of the articles used as sources for the discussion here are W. Wayt Gibbs, "Software's Chronic Crisis," *Scientific American,* September 1994, 271:3, pp. 86–95. Robert L. Scheier, "Software Snafu Grounds Denver's High-Tech Airport," *PC Week,* 11:19, May 16, 1994, p. 1. Price Colman, "Software Glitch Could Be the Hitch. Misplaced Comma Might Dull Baggage System's Cutting Edge," *Rocky Mountain News,* April 30, 1994, p. 9A. Steve Higgins, "Denver Airport: Another Tale of Government High-Tech Run Amok" *Investor's Business Daily,* May 23, 1994, p. A4. Julie Schmit, "Tiny Company Is Blamed for Denver Delays," USA Today, May 5, 1994, pp. 1B, 2B.

20 Scheier, "Software Snafu Grounds Denver's High-tech Airport."

21 Data compiled by Peter Mellor, Centre for Software Reliability, England, reprinted in Peter G. Neumann, *Computer-Related Risks,* Addison Wesley, 1995, p. 309.

22 "Airbus Safety Claim 'Cannot Be Proved'," *New Scientist,* September 7, 1991, 131:1785, p. 30. Robert Morrell Jr., Risks Forum Digest, July 6, 1994, 16:20. "Training 'Inadequate' Says A320 Crash Report," *Flight International,* December 22, 1993, p. 11 (on the January 1992 Strasbourg A320 crash, excerpted by Peter B. Ladkin in Risks Forum Digest, January 2, 1994). Ross, "The Day the Software Crashed," p. 156, on the 1993 Warsaw crash. David Learmont, "Lessons from the Cockpit," *Flight International,* January 11, 1994.

23 William M. Carley, "Could a Minor Change in Design Have Saved American Flight 965?" *Wall Street Journal,* January 8, 1995, pp. A1, A8.

24 Barry H. Kantowitz, "Pilot Workload and Flightdeck Automation," in M. Mouloua and R. Parasuraman, eds., *Human Performance in Automated Systems: Current Research and Trends,* pp. 212–23. The TCAS problems are mentioned on p. 214. Tom Forester and Perry Morrison, *Computer Ethics: Cautionary Tales and Ethical Dilemmas in Computing,* second edition, MIT Press, 1994, mentions avoidance of a collision, p. 113.

25 Peter G. Neumann, *Computer-Related Risks,* p. 158.

26 Nancy G. Leveson and Clark S. Turner, "An Investigation of the Therac-25 Accidents," *IEEE Computer,* July 1993, 26:7, pp. 18–41. Jonathan Jacky, "Safety-Critical Computing: Hazards, Practices, Standards, and Regulation," in Charles Dunlop and Rob Kling, eds., *Computerization and Controversy,* Academic Press, 1991. Most of the factual information about the Therac-25 incidents in this chapter is from Leveson and Turner.

27 Conversation with Nancy Leveson, January 19, 1995.

28 Jonathan Jacky, "Safety-Critical Computing: Hazards, Practices, Standards, and Regulation," in Charles Dunlop and Rob Kling, eds., *Computerization and Controversy,* Academic Press, 1991, p. 615. Peter G. Neumann, "Risks to the Public in Computers and Related Systems," *Software Engineering Notes,* April 1991, 16:2, p. 4. Ted Wendling, "Lethal Doses: Radiation That Kills," *Plain Dealer,* December 16, 1992, p. 12A. (I thank my student Irene Radomyshelsky for bringing the last reference to my attention.)

29 "Airbus Safety Claim 'Cannot Be Proved'," *New Scientist,* September 7, 1991, 131:1785, p. 30.

30 Richard P. Feynman, *What Do You Care What Other People Think?,* W. W. Norton & Co., 1988, pp. 182–183.

31 Govinda Rajan and Mathew Lodge, "X Windows Makes Patient Breathless ," Risks Forum, March 10, 1994, 15:64.

32 Ross, "The Day the Software Crashed," p. 146. A colleague warned me not to describe this or any other example in my book as "well done." The day the book is published there could be a headline saying "U.S. Banking System Crashes Due to CIPS Software Error." That would be a reminder that certainty and perfection in complex software systems are not possible. Professional expertise, responsibility, good planning, sensible testing, and so on, are.

33 From an e-mail advertisement for Nancy G. Leveson, *Safeware: System Safety and Computers* (Addison Wesley, 1995).

34 See, for example, the Shneiderman book in the list of references below.

35 Kantowitz, "Pilot Workload and Flightdeck Automation."

36 Feynman, *What Do You Care What Other People Think?* Feynman also mentions that the Challenger used obsolete hardware. Part of the reason was the recognition of the difficulty and expense of developing a completely new system from scratch for modern hardware that would meet the safety requirements of the shuttle. But using the old equipment had safety risks, for example, from the difficulty of getting high quality parts. Feynman thought it was time to make the change. The shuttle, like other systems, is not immune from problems. The Risks Forum includes reports of computer failures caused by a loose piece of solder, subtle timing errors, and other factors.

37 "AT&T Crash, 15 January 90: The Official Report."

38 Feynman, *What Do You Care What Other People Think?,* pp. 190–94 and 232–36. Aeronautics and Space Engineering Board, National Research Council, *An Assessment of Space Shuttle Flight Software Development Processes,* National Academy Press, 1993.

39 Mary Brandel and Thomas Hoffman, "User Lawsuits Drag on for NCR," *Computerworld,* August 15, 1994, p. 1. Andy Pasztor, "Lucas's $18.5 Million Fine for Work on Navy Jet Underscores Safety Fears," *Wall Street Journal,* January 1995, p. A4.

40 These problems and trade-offs occur often with regulation of new drugs and medical devices, regulation of pollution, and various kinds of safety regulation. They are discussed primarily in journals on the economics of regulation, but some received general publicity when AIDS activists encountered and fought the FDA bureaucracy. Several medical devices that doctors and surgeons believe could save thousands of lives are available in other countries but cannot be used in the U.S.

41 Jacky, "Safety-Critical Computing: Hazards, Practices, Standards, and Regulation," p. 624.

42 Leveson and Turner, "An Investigation of the Therac-25 Accidents," p. 40.

43 See, for example, Walter Williams, *The State Against Blacks,* McGraw-Hill, 1982, Chapters 5–7. One year during a construction lull, a state failed everyone who took the contractor's license exam. It is illegal in 48 states for most software engineers to call themselves software engineers because of licensing laws for engineers. One company was forced to spend thousands of dollars changing job titles, business cards, and marketing literature to remove the word "engineer." (Julia King, "Engineers to IS: Drop That Title!" *Computerworld,* May 30, 1994, 28:22, pp. 1, 119.)

44 Miles Corwin and John L. Mitchell, "Fire Disrupts L.A. Phones, Services," *Los Angeles Times,* March 16, 1994, p. A1.

45 William R. Hersh, "Informatics: Development and Evaluation of Information Technology in Medicine," *Journal of the American Medical Association,* January 1, 1992, p. 167.

46 "AT&T Phone Failure Downs Three New York Airports for Four Hours," *Software Engineering Notes,* October 1991, 16:4, pp. 6–7. "What's News," *Wall Street Journal,* January 10, 1995, p. A1.

47 Yoichiro Fujiyoshi, quoted in Craig Forman et al, "Quake's Aftershocks Will Rattle Segments of Japan's Economy," *Wall Street Journal,* January 18, 1995, p. A1.

48 National Transportation Statistics, Annual Report, September 1993, Historical Compendium, 1960–1992, U.S. Dept. of Transportation, p. 74 and p. 95.

49 Data from the National Air Transportation Safety Board and the U. S. Bureau of the Census, *Statistical Abstract of the United States:* 1994, Tables 134 and 996. Deaths in railroad accidents have also declined.

50 Forester and Morrison, *Computer Ethics,* p. 4.

51 Peter G. Neumann, "Inside Risks: Risks on the Rails," *Communications of the ACM,* July 1993, 36:7, p. 130, and *Computer-Related Risks,* p. 71. Although I take issue with some particulars, Neumann's Risks Forum, articles, and book are invaluable sources of information and analysis about risks of computer systems.

52 "More Risks of Computer Billing—$22,000 Water Bill," *Software Engineering Notes,* October 1991, 16:4, p. 6. Richard M. Rosenberg, "Success Components for the 21st Century," *The Magazine of Bank Management,* January/February 1994.

53 Here is a non-computer-related example. A study by the Harvard Center for Risk Analysis of the cost of various medical procedures and techniques for accident prevention and pollution control

found that control of benzene emissions at tire manufacturing plants costs almost $20 billion per life-year saved. Most other techniques studied cost far less per life-year saved. Overall safety could be improved if resources were directed at the more cost-effective techniques. (Tammy O. Tengs et al, "Five Hundred Life-Saving Interventions and Their Cost-Effectiveness," Center for Risk Analysis, Harvard School of Public Health. Some safety experts and consumer advocates object to such analyses because they can be manipulated to justify reductions in safety or to shirk responsibility for unsafe systems. Risk experts considered the Harvard study to be thorough and rigorously done.)

54 Amanda Bennett, "Strange 'Science': Predicting Health-Care Costs," *Wall Street Journal,* February 7, 1994, p. B1.

55 Cynthia Crossen, "How 'Tactical Research' Muddied Diaper Debate," *Wall Street Journal,* May 17, 1994, pp. B1, B9.

56 Richard P. Turco, Owen Toon, Thomas Ackerman, James Pollack, and Carl Sagan, "Nuclear Winter: Global Consequences of Multiple Nuclear Explosions," *Science,* December 23, 1984, p. 1284. Russell Seitz, "In From the Cold: 'Nuclear Winter' Melts Down," *The National Interest,* Fall 1986.

57 The main sources for this section include J. O. Hallquist and D. J. Benson, "DYNA3D—An Explicit Finite Element Program for Impact Calculations," in *Crashworthiness and Occupant Protection in Transportation Systems,* T. B. Khalil and A. I. King, eds., American Society of Mechanical Engineers, v. 106, p. 1. William H. Allen, "How To Build a Better Beer Can," pp. 32–36, and "DYNA Gets to the Heart of the Matter," p. 36, both in *Supercomputing Review,* March 1991. Steve Wampler, "DYNA3D: This Computer Code Seems to Offer Something for Everyone," *The Quarterly,* Lawrence Livermore National Laboratory, September 1989, 20:2, pp. 7–11. "Motorists, Pedestrians May Find LLNL Computer Code a Life-Saver" and "LLNL Computer Code Makes the Jump from Modeling Machines to Man," news releases from Lawrence Livermore National Laboratories, March 7, 1991 (NR-91-03-01 and NR-91-03-02).

58 Thomas Frank and Karl Gruber, "Numerical Simulation of Frontal Impact and Frontal Offset Collisions," *Cray Channels,* Winter 1992, pp. 2–6.

59 J. T. Houghton, G. J. Jenkins, and J. J. Ephraums, eds., *Climate Change: The IPCC Scientific Assessment,* Cambridge University Press, 1990. J. T. Houghton, B. A. Callander, and S. K. Varney, eds., *Climate Change 1992: The Supplementary Report to the IPCC Scientific Assessment,* Cambridge University Press, 1992. Most of the information in this section, including strengths and weaknesses of the models, comes from these reports. I also used a large variety of other books and articles for background. One book by a climatologist critical of the models is included in the references at the end of the chapter.

60 Houghton et al, *Climate Change* 1992, p. 14.

61 Andrew Lacis, *R & D,* June 1992, 34:7, p. 22.

62 Houghton et al, *Climate Change 1992,* pp. 17, 139. Patrick J. Michaels, *Sound and Fury: The Science and Politics of Global Warming,* Cato Institute, 1992, pp. 114–118.

63 Houghton et al, *Climate Change 1992,* p. 103. Carl Zimmer, "Verdict (Almost) In," *Discover,* January 1996, pp. 78–79. "Report on Global Warming Makes Dire Predictions," *Wall Street Journal,* October 25, 1995, p. B5. (In spite of the headline, this article reports that the new IPCC prediction for global temperature increase in the next century is 1°C-3.5°C, which is lower than its prediction of a few years earlier.)

64 Houghton et al, *Climate Change: The IPCC Scientific Assessment,* p. 243.

65 Houghton et al, *Climate Change 1992,* p. 19.

66 The quotes are from Edward Kennedy and Barbara Jordan, in David Lawsky, "New Job Checks on Immigration Status Proposed," Reuters World Service, August 3, 1994.

67 A crash of a computer-controlled roller coaster in Missouri in 1990 injured 28 people. Peter G. Neumann, "Inside Risks: Risks on the Rails," *Communications of the ACM,* July 1993, 36:7, p. 130.

68 Forester and Morrison, *Computer Ethics,* pp. 4–5.

69 Dana Milbank, "Barnyard Breakthrough: Cow Milks Herself," *Wall Street Journal,* May 8, 1995, pp. B1, B6.

70 Bruce Stutz, "The Landscape of Hunger," *Audubon,* March/April 1993, pp. 54–63 (quotation on p. 62).

FOR FURTHER READING

W. Robert Collins, Keith W. Miller, Bethany J. Spielman, and Phillip Wherry, "How Good Is Good Enough?" *Communications of the ACM,* January 1994, 37:1, pp. 81–91. A discussion of ethical issues about quality for software developers.

Richard Epstein, *The Case of the Killer Robot,* John Wiley and Sons, 1996.

Richard P. Feynman, *What Do You Care What Other People Think?,* W. W. Norton & Co., 1988. Includes Feynman's report on the investigation of the destruction of the Challenger space shuttle, with many insights about how to, and how not to, investigate a system failure.

J. T. Houghton, G. J. Jenkins, and J. J. Ephraums, eds., *Climate Change: The IPCC Scientific Assessment,* Cambridge University Press, 1990.

J. T. Houghton, B. A. Callander, and S. K. Varney, eds., *Climate Change 1992: The Supplementary Report to the IPCC Scientific Assessment,* Cambridge University Press, 1992.

Jonathan Jacky, "Safety-Critical Computing: Hazards, Practices, Standards, and Regulation," in Charles Dunlop and Rob Kling, eds., *Computerization and Controversy,* Academic Press, 1991.

Thomas K. Landauer, *The Trouble With Computers: Usefulness, Usability, and Productivity,* MIT Press, 1995.

Nancy G. Leveson, *Safeware: System Safety and the Computer Age,* Addison Wesley, 1995.

Nancy G. Leveson and Clark S. Turner, "An Investigation of the Therac-25 Accidents," *IEEE Computer,* July 1993, 26:7, pp. 18–41.

Patrick J. Michaels, *Sound and Fury: The Science and Politics of Global Warming,* Cato Institute, 1992. Michaels is a climatologist critical of the climate models.

Peter G. Neumann, moderator, Risks Forum, Usenet (comp.risks).

Peter G. Neumann, *Computer-Related Risks,* Addison-Wesley, 1995.

Peter G. Neumann et al., "Inside Risks," *Communications of the ACM,* regular column, on the last page of each issue.

Donald Norman, *The Psychology of Everyday Things,* Basic Books, 1988. A study of good and bad user interfaces on many everyday devices and appliances.

Effy Oz, "When Professional Standards Are Lax: The CONFIRM Failure and its Lessons," *Communications of the ACM,* October 1994, 37:10, pp. 29–36. A study of a $125-million-dollar project that was canceled.

David Parnas, "SDI: Violation of Professional Responsibility," *Abacus,* Winter 1987, pp. 46–52.

Ivars Peterson, *Fatal Defect: Chasing Killer Computer Bugs,* Times Books (Random House), 1995.

Henry Petroski, *To Engineer Is Human: The Role of Failure in Successful Design,* St. Martin's Press, 1985.

Jeffrey Rothfeder, *Privacy for Sale,* Simon & Schuster, 1992. Although the main focus of this book is privacy, it contains many examples of problems that resulted from errors in databases.

Ben Shneiderman, *Designing the User Interface,* Addison-Wesley, 1987.

Aaron Wildavsky, *Searching for Safety,* Transaction Books, 1988. On the role of risk in making us safer.

5 PROTECTING SOFTWARE AND OTHER INTELLECTUAL PROPERTY