Errors, Failures, and Risks

CSE 312 – Legal, Social, and Ethical Issues in Information Systems

Stony Brook University

http://www.cs.stonybrook.edu/~cse312

Ch 8: Errors, Failures, and Risks

Failures and Errors in Computer Systems

- Most computer applications are so complex it is virtually impossible to produce programs with no errors
- Real incidents:

"Navigation System Directs Car Into River"

"Data Entry Typo Mutes Millions of U.S. Pagers"

"Flaws Found in Software That Tracks Nuclear Materials"

"Software Glitch Makes Scooter Wheels Suddenly Reverse Direction"

"IRS Computer Sends Bill for $68 Billion in Penalties"

"Robot Kills Worker"

"California Junks $100 Million Child Support System"

"Man Arrested Five Times Due to Faulty FBI Computer Data"

Failures and Errors in Computer Systems

Are computer systems too unreliable and too unsafe to use?

- The cause of failure is often more than one factor
- Computer professionals must study failures to learn how to avoid them
  - faulty design
  - confusing documentation
  - careless or insufficiently trained users
  - poor user interfaces
  - sloppy manufacturing or servicing
- Computer professionals must study failures to understand the impacts of poor work
  - distinguish between errors we should accept as trade-offs for the benefits of the system and errors that are due to inexcusable carelessness, incompetence, or dishonesty

Failures and Errors in Computer Systems

Problems for Individuals in their roles as consumers

- Billing errors
  - A woman received a $6.3 million bill for electricity (the correct amount was $63)
  - IRS:
    - sent an Illinois couple a bill with $68 billion in penalties
    - sent 3000 people bills for slightly more than $300 million
    - sent a woman a $40,000,001,541.13 bill
  - The auto insurance rate of a 101-year-old man suddenly tripled: the rates depend on age, but the program handled ages only up to 100, so it mistakenly classified the man as a teenager
  - Hundreds of Chicago cat owners received bills from the city for failure to register dachshunds, which they did not own (the code in one DB for cat was a dachshund in the other DB)

Failures and Errors in Computer Systems

Problems for Individuals in their roles as consumers

- Inaccurate and misinterpreted data in databases
  - Large population where people may share names
    - Stores used a database to screen job applicants. It listed a man as a shoplifter- A real shoplifter had given the police the innocent man's identification from a lost wallet
    - A family was harassed, threatened, and physically attacked after their state posted an online list of addresses where sex offenders live – the state did not know the offender had moved away before the family moved in
    - A high school excluded a 14-year-old boy from football and some classes without explanation because two schools used different disciplinary codes in their computerized records: the boy had been guilty of chewing gum and being late, but the system listed him as a drug user
  - Overconfidence in the accuracy of data
    - FBI's National Crime Information Center (NCIC) database showed the innocent man as wanted—someone using his name was committing crimes.
  - Errors in data entry
  - Lack of accountability for errors

Failures and Errors in Computer Systems

System Failures

- Modern communications, power, medical, financial, retail, and transportation systems depend heavily on computer systems
  - Millions of BlackBerry users did not get their email for nine hours after the company installed a faulty software update.
  - Customers of AT&T lost telephone service for voice and data for hours because of a software error in a four-million-line program.
  - A three-line change in a two-million-line telecommunications switching program caused a failure of telephone networks in several major cities
    - the program underwent 13 weeks of testing, but it was not retested after the change—which contained a typo.
  - American Express Company's credit card verification system failed during the Christmas shopping season.
    - Merchants had to call in for verification, overwhelming the call center

Failures and Errors in Computer Systems

System Failures

- Galaxy IV satellite computer failed
  - Pager service stopped for an estimated 85% of users in the United States, including hospitals and police departments.
  - Airlines that got their weather information from the satellite had to delay flights.
  - The gas stations of a major chain could not verify credit cards.
- An error in a software upgrade shut down trading on the Tokyo Stock Exchange
- A glitch in an upgrade in the computer system at Charles Schwab Corporation crashed the system for more than two hours and caused intermittent problems for several days. Customers could not access their accounts or trade online.
- A computer malfunction froze the London Stock Exchange for almost eight hours—on the last day of the tax year, affecting many people's tax bills
- Amtrak's reservation and ticketing system failed during Thanksgiving weekend caused delays
- Virgin America airline switched to a new reservation system a month before Thanksgiving and its website and checkin kiosks did not work properly for weeks

Failures and Errors in Computer Systems

System Failures

- The $125 million Mars Climate Orbiter disappeared when it should have gone into orbit around Mars.
  - One team working on the navigation software used English-measure units while another team used metric units.
- NASA's Spirit rover became unresponsive a few weeks after landing on Mars.
  - Engineers found that too many files had accumulated in the rover's flash memory
- Mars Climate Orbiter was also destroyed due to software on the ground generating commands in pound-force (lbf), while the orbiter expected newtons (N).
- Mars Polar Lander was destroyed because its flight software mistook vibrations due to atmospheric turbulence for evidence that the vehicle had landed and shut off the engines 40 meters from the Martian surface
- A booster went off course during launch, resulting in the destruction of NASA Mariner 1 because of an incorrect formula in its FORTRAN software
- The European Space Agency's Ariane 5 Flight 501 was destroyed 40 seconds after takeoff because of a bug in the on-board guidance software
- Mars Pathfinder mission was jeopardised by a bug in concurrent software shortly after the rover landed
- A Zenit 3SL launch failed due to faulty ground software not closing a valve in the rocket's second stage pneumatic system

Failures and Errors in Computer Systems

System Failures

- Voting systems
  - The U.S. presidential election of 2000 demonstrated some of the problems of old-fashioned election machines and paper or punch-card ballots.
  - Human vote counters found these ballots sometimes difficult to read or ambiguous
  - In 2002, Congress passed the Help America Vote Act and authorized $3.8 billion to improve voting systems.
  - By the 2006 elections, only a very small percentage of Americans voted with paper ballots
- Technical failures
  - Machines in North Carolina lost more than 4000 votes because the machine's memory was full
  - A programming error generated 100,000 extra votes in one Texas county
  - A programming error caused some candidates to receive votes actually cast for other candidates.
- Programmers or hackers rigging software to produce inaccurate results.
- Vulnerability to viruses

Failures and Errors in Computer Systems

System Failures

- Stalled airports: Denver
  - The computer-controlled baggage-handling system, which cost $193 million, caused most of the delay
    - An automated system of carts traveling at up to 19 miles per hour on 22 miles of underground tracks
    - Carts crashed into each other at track intersections.
    - The system misrouted, dumped, and flung luggage
    - A software error caused the routing of carts to waiting pens when they were actually needed
  - Main causes:
    - Time allowed for development was insufficient
      - The only other baggage system of comparable size was at Frankfurt Airport in Germany and it took 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
    - Denver made significant changes in specifications after the project began: originally it was designed for United, but Denver officials decided to expand it to include the entire airport (14 times larger)

Failures and Errors in Computer Systems

System Failures

- Abandoned systems
  - Some flaws in systems are so extreme that the systems are discarded after wasting millions, or even billions, of dollars.
    - A large British food retailer spent more than $500 million on an automated supply management system; it did not work
    - Ford Motor Company abandoned a $400 million purchasing system
    - California and Washington state motor vehicle departments each spent more than $40 million on computer systems that never worked

Failures and Errors in Computer Systems

System Failures

- 15% of information technology projects are abandoned before or soon after delivery as "hopelessly inadequate"
  - Lack of clear, well-thought-out goals and specifications
  - Poor management and poor communication among customers, designers, programmers, etc.
  - Institutional and political pressures that encourage unrealistically low bids, low budget requests, and underestimates of time requirements
  - Use of very new technology, with unknown reliability and problems
  - Refusal to recognize or admit a project is in trouble

Failures and Errors in Computer Systems

System Failures

- Legacy systems = out-of-date systems (hardware, software, or peripheral equipment) still in use, often with special interfaces, conversion software, and other adaptations to make them interact with more modern systems
  - These old systems are "Reliable but inflexible"
  - Expensive to replace: the early adopters of computers were: bank, power companies, airlines, goverments
  - Little or no documentation
- After US Airways and America West merged, they combined their reservations systems
  - most airline systems date from the 1960s and 1970s
  - The self-service check-in kiosks failed
- Merging different computer systems is extremely tricky, and problems are common

Failures and Errors in Computer Systems

System Failures

- Legacy systems
  - Old hardware fails and replacement parts are hard to find.
  - Old software often runs on newer hardware, but it is still old software.
  - Programmers no longer learn the old programming languages.
  - Old programs often had little or no documentation, and the programmers who wrote the software or operated the systems have left the company, retired, or died.
  - Limited computer memory led to obscure and terse programming practices.
  - The systems grew gradually.
  - A complete redesign and development of a fully new, modern system would be expensive.
  - It would require a major retraining project.
  - The conversion to the new system, requiring some downtime, could also be very disruptive.

Failures and Errors in Computer Systems

System Failures

- Legacy systems
- Lesson: someone might be using your software 30 or 40 years from now.
  - It is important for flexibility, expansion, and upgrades to document, document, document your work.

Failures and Errors in Computer Systems

What Goes Wrong?

- The job they are doing is inherently difficult.
  - Computer systems interact with the real world
  - Automobiles, passenger airplanes, and jet fighters contain millions of lines of computer code
- Sometimes the job is done poorly
  - Inadequate attention to potential safety risks
  - Interaction with physical devices that do not work as expected
  - Incompatibility of software and hardware, or of application software and the operating system
  - Not planning and designing for unexpected inputs or circumstances
  - Confusing user interfaces
  - Insufficient testing
  - Reuse of software from another system without adequate checking
  - Data-entry errors
  - Inadequate training of users
  - Errors in interpreting results or output
  - Failure to keep information in databases up to date
  - Insufficient planning for failures; no backup systems or procedures

Failures and Errors in Computer Systems

What Goes Wrong?

- Reuse of software: the Ariane 5 rocket and "No Fly" lists
- Ariane 5 rocket ($500 million) veered off course and was destroyed as a safety precaution
  - It reused software designed for the earlier, successful Ariane 4
  - This module did calculations related to velocity, but the Ariane 5 traveled faster than the Ariane 4 after takeoff.
  - The calculations produced numbers bigger than the program could handle ("overflow" errors), causing the system to halt.
- "No Fly" lists
  - TSA software checked only the first initial for first name
- It is essential to reexamine the specifications and design of the software, consider implications and risks for the new environment, and retest the software for the new use.

8.2 Case Study: The Therac-25

Therac-25 Radiation Overdoses

- Massive overdoses of radiation were given; the machine said no dose had been administered at all
- Caused severe and painful injuries and the death of three patients
- Insufficient testing, bugs in the software that controlled the machines, and an inadequate system of reporting and investigating the accidents
- Important to study to avoid repeating errors
- Manufacturer, computer programmer, and hospitals/clinics all have some responsibility

Case Study: The Therac-25

Software and Design problems

- Re-used software from older systems (Therac-6 and Therac-20), unaware of bugs in previous software
- Weaknesses in design of operator interface
- Inadequate test plan
- Bugs in software
  - Allowed beam to deploy when table not in proper position
    - A flag variable indicated whether a specific device on the machine was in the correct position. A zero value meant the device was ready; a nonzero value meant it must be checked.
    - The flag variable was stored in one byte. After the 256th call to the routine, the flag overflowed and showed a value of zero.
    - The solution is to set the flag variable to a fixed value, say 1, rather than incrementing it, to indicate that the device needs checking.
  - Ignored changes and corrections operators made at console

Case Study: The Therac-25

Why So Many Incidents?

- Overconfidence in software (that it cannot have errors)
  - The decision to eliminate the hardware safety mechanisms
  - In the first overdose incident, when the patient told the machine operator that the machine had "burned" her, the operator told her that was impossible
  - A camera in the treatment room and an intercom system enabled the operator to monitor the treatment and communicate with the patient
    - 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
    - The patient received a second overdose before getting to the door and pound on it

Case Study: The Therac-25

Why So Many Incidents?

- Hospitals had never seen such massive overdoses before, were unsure of the cause
- Manufacturer said the machine could not have caused the overdoses and no other incidents had been reported (which was untrue)
- The manufacturer made changes to the turntable and claimed they had improved safety after the second accident.
  - The changes did not correct any of the causes identified later.
- Recommendations were made for further changes to enhance safety; the manufacturer did not implement them.
- The FDA declared the machine defective after the fifth accident.
- The sixth accident occurred while the FDA was negotiating with the manufacturer on what changes were needed.

Case Study: The Therac-25

Observations and Perspective

- Design and implementation errors usually occur in complex systems
  - 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.
- The problems in the Therac-25 case were not minor and suggest irresponsibility
- Accidents occurred on other radiation treatment equipment without computer controls:
  - Did not properly measure the radioactive drugs
  - Confused micro-curies and milli-curies

Case Study: The Therac-25

Discussion Question

- *If you were a judge who had to assign responsibility in this case, how much responsibility would you assign to the programmer, the manufacturer, and the hospital or clinic using the machine?*

8.3 Increasing Reliability and Safety

Professional techniques

- Many large, complex computer systems work extremely well
  - The New York Stock Exchange installed a $2 billion system with hundreds of computers, 200 miles of fiber-optic cable, 8000 telephone circuits, and 300 data routers. The system handled the sales without errors or delays
- We rely on them daily.
- Importance of good software engineering and professional responsibility
  - How can we design, build, and operate systems that are likely to function well?
- To produce good systems, we must use good software engineering techniques at all stages of development, including specifications, design, implementation, documentation, and testing
- Testing
  - Include real world testing with real users

Increasing Reliability and Safety

Professional techniques

- *High reliability organizations* (HRO) are organizations (business or government) that operate in difficult environments, often with complex technology, where failures can have extreme consequences (for example, air traffic control, nuclear power plants)
- High reliability organization principles
  - preoccupation with failure: always assume something unexpected can go wrong—not just plan, design, and program for all problems the team can foresee, but always being aware that they might miss something
  - loose structure: it should be easy for a designer or programmer to speak to people in other departments or higher up in the company without going through rigid channels that discourage communication

Increasing Reliability and Safety

Safety-critical applications

- Identify risks and protect against them
  - developers must "design in" safety from the start
- Convincing case for safety
- Avoid complacency
- Example: the night before the scheduled launch of space shuttle Challenger, the engineers argued for a delay
  - they knew the cold weather posed a severe threat to the shuttle
  - They could not prove absolutely that a system is safe, nor prove absolutely that it will fail
- A large piece of insulating foam dislodged and struck the wing of the Columbia space shuttle as it launched
  - NASA knew this happened, but pieces of foam had dislodged and struck the shuttle on other flights without causing a major problem

Increasing Reliability and Safety

Specifications

- Learn the needs of the client: understand how the client will use the system
  - long planning stage allows for discovering and modifying unrealistic goals
    - One company developed a successful financial system that processes one trillion dollars in transactions per day spent several years developing specifications for the system, then only six months programming, followed by carefully designed, extensive testing

Increasing Reliability and Safety

User interfaces and human factors

- User interfaces should:
  - provide clear instructions and error messages
  - be consistent
  - include appropriate checking of input to reduce major system failures caused by typos or other errors a person will likely make
- The crash of American Airlines Flight 965 near Cali, Colombia
  - While approaching the airport, the pilot intended to lock the autopilot onto the beacon, called Rozo, that would lead the plane to the airport
  - The pilot typed "R," and the computer system displayed six beacons beginning with "R." Normally, the closest beacon is at the top of the list
  - The pilot selected it without checking carefully (it was Romeo, 100miles away).

Increasing Reliability and Safety

User interfaces and human factors

- The user needs feedback to understand what the system is doing at any time.
- Ground Proximity Warning system (GPWS) contains a digital map of the world's topography and gives warning 1 minute before a crash
  - The GWPS is likely responsible for preventing crashes in several incidents in which pilots incorrectly set an altimeter, attempted to land with poor visibility, mistook building lights for airport lights
  - No commercial U.S. airliner has crashed into a mountain since the GPWS was implemented
- The system should behave as an experienced user expects.
- A workload that is too low can be dangerous.
  - An overworked operator is more likely to make mistakes.
  - However, a workload that is too low can lead to boredom, inattention, or lack of awareness of the current status

Increasing Reliability and Safety

Redundancy and self-checking

- Redundancy
  - Multiple computers capable of same task; if one fails, another can do the job.
- Voting redundancy
  - Software modules can check their own results—either against a standard or by computing the same thing in two different ways and then comparing to see if the two results match
  - Three independent teams write modules for the same purpose, in three different programming languages.
  - The modules run on three separate computers.
  - A fourth unit examines the outputs of the three modules and chooses the result obtained by at least two out of three

Increasing Reliability and Safety

Testing

- Even small changes need thorough testing
- Independent verification and validation (IV&V)
  - not the programmers nor the customer tests and validation of the software
  - They act as "adversaries" and try to find flaws
  - It works for 2 reasons:
    - The people who designed and/or developed a system think the system works.
      - They think they thought about potential problems and solved them.
      - With the best of intentions, they tend to test for the problems they have already considered.
    - The people who created the system may be reluctant to find flaws in it.
  - External IV is not practical for all projects
    - Many companies have their own testing teams

Increasing Reliability and Safety

Testing

- Beta testing
  - it is near-final stage of testing
  - A selected set of customers use a complete, presumably well-tested system in their "real-world" environment
  - It can detect device limitations and bugs that designers, programmers, and testers missed.
  - It can also uncover confusing aspects of user interfaces and problems that occur when interfacing with other systems.

Trust the Human or the Computer System?

- Traffic Collision Avoidance System (TCAS)
  - Computers in some airplanes prevent certain pilot actions
  - According to the head of the Airline Pilots Association's safety committee: it is a great advance in safety,
  - Pilots of the Airbus 380 are trained to allow its autopilot system to control the plane when a midair collision threatens
  - Prevents pilots from doing something "stupid"
  - Some people object, arguing that the pilot should have ultimate control in case unusual action is needed in an emergency

Law, Regulation, and Markets

- Criminal and civil penalties
  - Provide incentives to produce good systems, but shouldn't inhibit innovation
  - Are important legal tools for increasing reliability and safety of computer systems and accuracy of data in databases
- Examples:
  - Therac-25 victims sued; they settled out of court.
  - Several people have won large judgments against credit bureaus for incorrect data in credit reports that caused havoc in their lives

Law, Regulation, and Markets

- Regulation for safety-critical applications
  - 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.
    - Companies must do extensive testing, provide huge quantities of documentation, and get government approval before they sell new drugs and some medical devices. A
  - Patients do not have the expertise to judge the safety or reliability of a system
  - Overly strict standards can inhibit progress, require techniques behind the state of the art, and transfer responsibility from the manufacturer to the government.

Law, Regulation, and Markets

- Professional licensing
  - Mandatory licensing of software development professionals
  - Laws require licenses for hundreds of trades and professions
  - Licensing requirements typically include specific training, the passing of competency exams, ethical requirements, and continuing education
  - Requirements for specific degrees and training programs, as opposed to learning on one's own or on the job, tend to keep poorer people from qualifying for licenses
- Taking responsibility
  - In some cases of computer errors, businesses have an ethical policy of behaving responsibly and paying for mistakes (without a lawsuit)
    - Intuit offered to pay interest and penalties that resulted from errors in flawed income-tax programs

8.4 Dependence, Risk, and Progress

- Are We Too Dependent on Computers?
  - Because of their usefulness and flexibility, computers, cellphones, and similar devices are now virtually everywhere
    - Many drivers would be lost if their navigation system failed
    - Some military jets cannot fly without the assistance of computers.
  - In Holland, no one discovered the body of a reclusive, elderly man who died in his apartment until six months after his death
    - Many of the man's bills, including rent and utilities, were paid automatically.
    - His pension check went automatically to his bank account
    - "All the relevant authorities assumed that he was still alive."
  - In a similar case, 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
  - They are not the only dependence
    - Electricity: Computers make decisions; electricity does not

Dependence, Risk, and Progress

- Risk and Progress
  - We trust older technologies
  - Many new technologies were not very safe when they were first developed
  - We develop and improve new technologies in response to accidents and disasters
  - We should compare the risks of using computers with the risks of other methods and the benefits to be gained
    - The death rate from motor vehicle accidents in the United States declined almost 80% from 1965 to 2010 (from 5.30 per 100 million vehicle miles traveled to 1.13 per 100 million vehicle miles traveled).
    - The risk of dying in an on-the-job accident dropped from 39 among 100,000 workers (in 1934) to 5 in 100,000 in 2008
    - Why?

Dependence, Risk, and Progress

Discussion Questions

- *Do you believe we are too dependent on computers? Why or why not?*
- *In what ways are we safer due to new technologies?*

Conclusion for this chapter

- There is a "learning curve" for new technologies
  - Much is known about how to design, develop, and use complex systems well and safely. Ethical professionals learn and follow these methods.
- Perfection is not an option. The complexity of computer systems makes errors, oversights, and failures likely
  - This does not mean that we should excuse or ignore computer errors
  - Computer system developers and other professionals responsible for planning and choosing systems must assess risks carefully and honestly, include safety protections, and make 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