Topic 8: Failures and risks

MPU33213

Computer Ethics and Professional Responsibility

Topics covered

- Individual errors and system failures
- Reasons for errors
- Increasing reliability and safety
- Law and regulation
- Professional licensing

Problems caused by failures

- 1983 (Soviet nuclear false alarm incident): bugs in early-warning monitoring system nearly caused nuclear war.

https://en.wikipedia.org/wiki/1983_Soviet_nuclear_false_alarm_incident#:~:text=On%2026%20September%201983%2C%20during,bases%20in%20the%20United%20States.

- 1996 (software bug): $924,844,208.32 credited to each of 823 customers of a major U.S. bank.

https://www.linkedin.com/pulse/damaging-impacts-ignoring-importance-software-testing-youssef-touati

- 1997 (millennium bug): a transaction processing system failed to handle credit cards that expire in year 2000

https://en.wikipedia.org/wiki/Year_2000_problem

- 2014 (Microsoft bug): found a bug that had been in an OS for at least 19 years

https://www.washingtonpost.com/news/the-switch/wp/2014/11/12/microsoft-just-squashed-a-19-year-old-software-bug-how-did-it-go-undetected-so-long/

- 2016 (computer misjudged altitude): spacecraft crash landed on Mars
  - https://www.theguardian.com/science/2016/nov/24/mars-lander-smashed-into-ground-at-540kmh-after-misjudging-its-altitude

Problems caused by failures

- 2019 (Google Map Bug): Navigation system directs car into river
- https://www.vice.com/en/article/panan7/truck-falls-into-river-accident-google-maps-directions

- *Possible to produce complex program with no error?*

Failures

- Affect individuals / consumers.
- Affect a large numbers of people and/or cost large amount of money.
- Problems in safety-critical applications may cause injury / death.

Problems for individuals

Many people are inconvenienced and/or suffer losses from errors in billing systems and database containing personal data

- Billing errors:
  - $63 electricity bill became $6.3 million, caused by input error using a new computer system.
  - Auto insurance of a 101 years old man tripled because the program handled age only up to 100. The man is classified as a teenager.
  - Cat owners received bills for failure to register dogs which they did not own. Cause: 2 databases were used. Same code was used for cat in one database and for dog in another database.

Problems for individuals

- Some errors can be avoided. E.g. test to determine whether a bill amount is outside some reasonable range or changed significantly from previous bills.
- Significant errors, bigger affected scope are easily identified and fixed quickly.
  - What about errors in individual’s record?
  - i.e., submitted record, without pre-examined etc.

Problems for individuals

- Wrong middle name was used in a credit report, another person had the exact name could not get credit to buy a car or a house.
- Job applications of a man were turned down because the database used to screen applicants listed him as a shoplifter. A shoplifter had given police the innocent man’s identification from a lost wallet.-*individual entries wrongly keyed*

Problems for individuals

- A family was harassed, threatened and attacked after a list of addresses where sex offenders live was posted online. The offender had moved, the authority didn't know and the list was not updated. – *database manager not doing a good job*
- A 14 years old boy was excluded from some classes because the school officials thought he had been using drugs. Same code was used for different disciplinary issues in 2 different databases. – *database manager not doing a good job*

Problems for individuals

- Errors in databases caused the police arrested innocent people.
  - Rented car mistakenly listed as stolen car.
  - Name was used by a criminal.

Problems for individuals

- Factors that contribute to the problems:
  - A large population with same / similar names.
  - Automated processing without human common sense or the power to recognise special cases.
  - Overconfidence in the accuracy of data stored.
  - Errors in data entry.
  - Failure to update or correct errors.
  - Lack of accountability for errors.
- Arguments:
  - The first item is unlikely to change. It is the context in which we live.
  - The second is partly a side effect of the speed and processing ability of computer technology, but we can reduce its negative impact with better system specifications and training of users.
  - Even if someone corrects an error in a database, problems may not be over for the affected person. Computer records are copied easily and often. Copies of the incorrect data may remain in other systems. The remaining factors in the list above are all within our control as individuals, and policy makers.

System Failures

Modern communication, power, medical, financial, retail, and transportation systems depend heavily on computer systems. They do not always function as planned.

- BlackBerry users didn’t get email for 9 hours after the company installed a faulty software update.
- AT&T customers lost service for hours because of a software error.
- American Express credit card verification system failed, merchants had to call in for verification, overwhelming the call center.

System Failures

- Galaxy IV satellite computer failed, airlines had to delay flights, credit card verifications failed.
- Error in software upgrade shut down trading on Tokyo Stock Exchange.
- Failure of Amtrak’s reservation & ticketing system caused delays.
- Mars Climate Orbiter disappeared, caused by different measuring units used by 2 teams.

System Failures

Electronic systems have the potential for reducing some kinds of fraud and accidental loss of ballots but they have not yet reached the level of security to ensure a reasonable degree of trust.

- Voting system
  - Failed to count more than 400 votes because of technical problem.
  - One county lost more than 4000 vote because the machine's memory was full.
  - Programming error generated 100,000 extra votes in one county.
  - Programming error caused some candidates received votes actually cast for the opponents.
  - Insecure encryption techniques or none at all.

System Failures

- Voting system
  - Insufficient security for installation of upgrades to software, poor physical protection of the memory card on which the system stores votes.
  - Vulnerability to virus to took over the machine and manipulate the results.
  - Developers lacked sufficient security training, omitted basic procedures such as input validation and boundary checks.

System Failures

- Voting system
  - Requires a high degree of professionalism, a high degree of security.
  - Many of the failures that occurred result from causes we will see over and over: lack of sufficient planning and thought about security issues, insufficient testing, and insufficient training

One policy that can help reduce errors and fraud it to make the software in electronic voting machines public. Then various experts and tech-savy members of the public can examine it. Companies that produce electronic voting systems have resisted this, arguing that the software is a propriatery. If they release it, they may lose an advantage over competitors.

System Failures

- Stalled airports (Denver International Airport)
  - The opening of Denver International Airport was rescheduled at least 4 times, mostly caused by the computer-controlled baggage-handling system.
  - Ambitious system:
    - Luggage was to travel to any part of the airport via automated carts system.
    - Computers used a database of flights, gates and routing information to control to route the carts to their destinations.

System Failures

- Stalled airports (Denver International Airport)
  - Problems occurred:
    - **Real world problems**: scanners got dirty or knocked out of alignment and could not detect carts. Faulty latches on the carts caused luggage to fall onto the tracks.
    - **Problems in other system**: Electrical system could not handle the power surges associated with the baggage system.
    - **Software errors**: error caused the routing of carts to waiting pens when they were actually needed.

System Failures

- Stalled airports (Denver International Airport)
  - 2 main causes:
    - **Insufficient development and testing time**. Comparable system at Frankfurt Airport in Germany was developed in 6 years and tested and debugged in additional 2 years. Denver system was ask to complete everything in 2 years.
    - **Significant changes were made after the project began**. The system was to United Airlines. Denver officials decided to expand it to include the entire airport, the system scope expanded.

System Failures

- Stalled airports (Hong Kong & Kuala Lumpur)
  - The systems were to manage **everything**: moving luggage, coordinating and scheduling crews, gate assignments, etc.
  - Hong Kong: cleaning crews, fuel trucks, baggage, passengers and cargo went to the wrong gates. Airplanes scheduled to take off were empty.
  - KL: airport employees had to write boarding passes by hand and carry luggage, flights delayed, food cargo rotted.

System Failures

- Stalled airports (Hong Kong & Kuala Lumpur)
  - Both airports blamed on the incorrect information entered by users.

- System that has a large number of users must be designed and tested to handle input mistakes.

Abandoned System

- The flaws in many systems are so fundamentals that the systems end up in the trash after wasting millions, or even billions of dollars.
  - A British food retailer spent more than 500 million on an automated supply management system that end up didn't work.
  - Ford Motor abandoned a 400 million purchasing system.

    https://www.cnet.com/tech/services-and-software/ford-scraps-oracle-based-procurement-system/
  - A consortium of hotels & rental car business cancelled a reservation system that had spent 125 million because it didn't work.

Many projects require much more time and money than originally planned. Some are never completed because they are beyond the capabilities of the current technology. Software expert Robert Charette estimates that from 5% to 15% of information technology projects are abandoned before or soon after delivery – out of about $ 1 trillion spent each year worldwide.

Reusing Software

- After US Airways and America West merged, they combined their reservation system. The system failed. Long lines of tickets counters delayed thousands of passengers. Merging different computer systems is extremely tricky. Some of the reported problems were out of date systems (hardware, software or peripheral equipment).
- The problems of legacy systems are numerous. Old hardware fails, and replacement parts are hard to find. Old software often runs on new hardware, but it is still old software. Programmers no longer learn the old programming languages. The programmers who wrote the software operated the systems have left the company, retired or dead. Old programs often had little or no documentation. If there were good design documents and manuals, they probably no longer exist or cannot be found.

Causes: System Failures

- Some High Level causes of Computer System Failures:
  - Lack of clear goals and specifications.
  - Poor management and communication among customers, designers, programmers, etc.
  - Institutional / political pressures encourage unrealistically low bids, underestimates time requirements.
  - Use of very new technology with unknown reliability and problems.
  - Refusal to recognise / admit project is in trouble.

Two Perspectives: Design & development and Management & use

- Factors in computer system errors & failures
  - Design & development
    - Inadequate attention to potential safety risks
    - Interaction with physical devices that don't work as expected.
    - Software / hardware / software / operating system incompatibility.
    - Not planning / designing for unexpected inputs / circumstances.
    - Confusing user interfaces.

Two Perspectives: Design & development and Management & use

- Factors in computer system errors & failures
  - Design & development
    - Insufficient testing.
    - Reuse of software from another system without adequate checking / testing.
    - Overconfidence in software.
    - Carelessness.

Two Perspectives: Design & development and Management & use

- Factors in computer system errors & failures
  - Management and use:
    - Data entry errors.
    - Inadequate training of users.
    - Errors in interpreting results / output.
    - Failure to update information in databases.
    - Overconfidence in software by users.
    - Insufficient planning for failures, no backup systems / procedures.

System Failures: Ariane 5

- Ariane 5 rocket was destroyed as a safety precaution in less than 40 seconds after the launch.
  - Software designed for Ariane 4 was reuse in Ariane 5. A module did calculation related to velocity.
  - Ariane 5 travels faster than Ariane 4, the calculations produced numbers bigger than the program could handle, caused the system halt.
- Should we reuse software?

System Failures: Aircraft control system failed

- A $2 billion air traffic control system failed due to insufficient computer memory
- The system failure was sparked due to a U-2 spy plane that was flying through the region. The $2.4 billion system, made by Lockheed Martin Corp, cycled off and on trying to fix the error, which was due to a lack of altitude information in the airplane’s flight plan.

System Failures: Aircraft control system failed

- After an air traffic controller entered an estimated altitude of the plane, the system calculated all possible flight paths to ensure that it was not on a crash course with other planes.
- However, that process caused the system to fall short in memory and shut down every other flight processing function. Fortunately, no injuries or accidents were reported.

System Failures: USS Yorktown Incident

- On Sept. 21, 1997, The Navy's systems chief has begun an investigation into the computer failure that left the Aegis cruiser USS Yorktown dead in the water for several hours last fall.

System Failures: USS Yorktown Incident

- A systems administrator fed bad data into the ship's Remote Database Manager, which caused a buffer overflow when the software tried to divide by zero.
- The overflow crashed computers on the LAN and caused the Yorktown to lose control of its propulsion system, Navy officials said.

System Failures: Northeast Blackout

- A programming error has been identified as the cause of alarm failures that contributed to the widespread power outage throughout parts of the Northeastern and Midwestern United States
- The failures occurred when multiple systems trying to access the same information at once got the equivalent of busy signals. The software should have given one system precedent.

System Failures: Northeast Blackout

- With the software not functioning properly at that point, data that should have been deleted were instead retained, slowing performance. Similar troubles affected the backup systems.

The Therac-25

https://ethicsunwrapped.utexas.edu/case-study/therac-25

- A software-controlled radiation therapy machine.
- 1985 – 1987, 6 patients were given overdoses of radiation.
  - In some cases, operator repeated overdose as the machine's display indicated that it had given no dose.
- Factors: lapses of good safety design, insufficient testing, software bugs, inadequate reporting & investigating system.

The Therac-25

- Software & design problems:
  - Older machines had independent hardware safety interlock that prevented the beam from firing in unsafe condition. Therac-25 eliminated it.
  - Reused software and the developers assumed it functioned correctly. (Wrong assumption!)
    - There were bugs but the hardware safety mechanisms did their job. The bugs were carried to Therac-25.
  - Therac-25 had dose-rate malfunction frequently, generally underdoses, operator became used to it.

The Therac-25

- Software & design problems:
  - Weaknesses in interface design
    - Display error numbers or obscure messages (“Malfunction 54” or “H-tilt”) rather than meaningful messages.
    - The errors were not explained in manuals.
  - A single byte flag was used, after the 256th call, the flag overflowed and showed a value of zero (indicates the device was ready)

The Therac-25

- Software & design problems:
  - There were some bugs caused the machine to ignore changes made at the console.

The Therac-25

- Why were there many incidents?
  - Didn’t know that it caused the injuries (never had such cases before).
  - Manufacturer said that it couldn’t have caused the injuries.
  - Manufacturer made come changes and claimed that they have improved the safety, in fact they were not sure the cause of the accident.
  - Hospitals & clinics considered the lost of income and while the cause of accident was unknown.

The Therac-25

- This case shown the irresponsibility of the manufacturer (in designing, responding and solving the problems).

Systems Failures & Errors

- Most computer applications are so complex it is virtually impossible to produce programs with no errors
- The cause of failure is often more than one factor
- Computer professionals must study failures to learn how to avoid them
- Computer professionals must study failures to understand the impacts of poor work, i.e. Follows the procedures and guidelines established in the various relevant code of ethics and professional practices (The ACM/IEEE-CS Software Engineering Code of Ethics and Professional Practice and the ACM Code of Ethics and Professional Conduct).
- An atmosphere of open, honest communication within the software development company and between the company and the client is essential for learning of problems at an early stage.

Increasing Reliability & Safety

- Use good software engineering techniques at all stages of development
  - Include end-users in the design and testing stages.
  - Understand the entire scope of areas affected by the system. (Help clients to better understand their goals and requirements)
  - Allocate sufficient time and budget.

Increasing Reliability & Safety

- For safety critical software
  - “design in” safety from the start by using techniques of hazard analysis to identify risks.
  - developers should have special training.
  - ensure that the system is as safe as possible before its actual use.

Increasing Reliability & Safety

- Provide clear instructions & error messages.
- Validate input to reduce failures caused by typos or carelessness.
- Design of user interface should be consistent.
- Perform Independent verification and validation (IV&V).

Increasing Reliability & Safety

- Perform Beta testing
  - Selected set of customers use in “real-world” environment.
  - Detect software limitations and bugs that the designers, programmers, and testers missed.
  - It can uncover confusing aspects of user interfaces.
  - Uncover other problems: interfacing with other systems, running on old systems.

Law, Regulations & Markets

- Legal remedies for faulty systems include
  - lawsuits against the company that developed or sold the system.
  - criminal charges when fraud or criminal negligence occurs.
  - laws must be balanced to encourage innovation to produce good systems but penalise offenders.

Law, Regulations & Markets

- Regulations for safety-critical applications
  - Perform extensive specific testing.
  - Provide documentations.
  - Get approval from government.
- Arguments in favour:
  - Most customers at risk do not have expertise to judge the safety / reliability of a system.
  - Better to prevent use of a bad product than to rely on lawsuits.

Law, Regulations & Markets

- Mandatory licensing for software professionals (to improve software quality)
  - Include specific training, passing competency exams, ethical requirements and continuing education.
- Some businesses compensate customers for problems / damages because customer satisfaction and good business reputation are important.

Law, Regulations & Markets

- Some businesses pay more for uninterrupted satellite communications service.
- To protect from faulty software
  - Consult websites that review new programs.
  - Checking with social network or online group forums.
  - Check the seller's reputation with various business bureaus.

THE END