Errors, Failures, and Risks

Understanding the risks and reasons for computer failures.

Trusting Computers

- What can go wrong?
- Case Studies
- Increasing Reliability and Safety
- Computer Models
- Perspectives on Failures, Dependence, Risk and Progress
- Learning from other technologies

What can go wrong?

- Describe a system failure that affected…
  - You or a friend
  - Many people

RISKs digest

- “Forum on Risks to the Public in Computers and Related systems”

BSoD forces students to retake standardized test (May 15, 2007)

2900 Virginia students will have to re-take standardized tests because the computer systems failed during the testing process. There are two descriptions of what went wrong: the testing vendor "reported that there was a problem with a connection between two servers" and students' "computer screens suddenly turned blue and displayed an error message" (i.e., a BSoD). Whether this is one problem or two is unclear... "State officials said there was an unrelated computer problem with online testing last week [where] 1,300 tests were interrupted and that the students will have to be retested."

RISKs digest

Scots Jail hi-tech door locking system broke (September 20, 2005)

Prison officers have been forced to abandon a new security system and return to the use of keys after the cutting-edge technology repeatedly failed. The system, which is thought to have cost over £3 million, used fingerprint recognition to activate the locking system at the high-security Glenochil Prison near Tullibody, Clackmannanshire. ... For more than a month, the 420 inmates - including some murderers and other high-risk inmates - had been able to wander around the high-security jail. Staff claim that the unlimited access to all parts of the prison had allowed some prisoners to settle old scores with rivals.

RISKs digest

Computer problem suspected in erratic Airbus flight (Jan 11, 2008)

An Air Canada flight from Victoria -> Toronto rolled suddenly from side to side and then plunged in the air. It may have suffered technical problems, according to passengers interviewed after the plane was diverted to Calgary. There had been a computer failure and pilots were flying the plane manually. Ten people were admitted to hospital.

What can go wrong?

- Facts about computer errors
  - Error-free software is very, very difficult to achieve.
  - Errors are often caused by several factors.
  - Errors can be reduced by following good software development procedures.
  - Confidence in code can be obtained by keeping it small enough to be understood.

What can go wrong?

- Roles
  - Computer user
    - Should understand limitations of computers
    - Should understand need for proper training and responsible use
  - Computer professional
    (buyer, developer, system manager…)
    - Needs to understand the sources and consequences of failures
  - Educated member of society
    - Understanding computer risks helps in discussing and deciding public policy.

What can go wrong?

- Categories of computer errors and failures
  - Problems for individuals
    - affects one or a few people
  - System Failures
    - affects large numbers of people or costs large amounts of money or both
    - classic example: telecommunications network
  - Safety-Critical Applications
    - where property may be damaged or destroyed
    - where people may be injured or killed

What can go wrong?

- Problems for Individuals
  - Billing errors
    - Lack of tests for inconsistencies and inappropriate amounts
    - Perhaps caused by data entry errors due to poor HCI
  - Database accuracy problems
    - Incorrect information resulting in wrongful treatment or acts
  - Causes
    - large, diverse population
    - human common-sense not part of automated processing
    - overconfidence in accuracy of data from a computer
    - information not updated or corrected
    - lack of accountability for errors
- Q: How accurate should computer systems be?

What can go wrong?

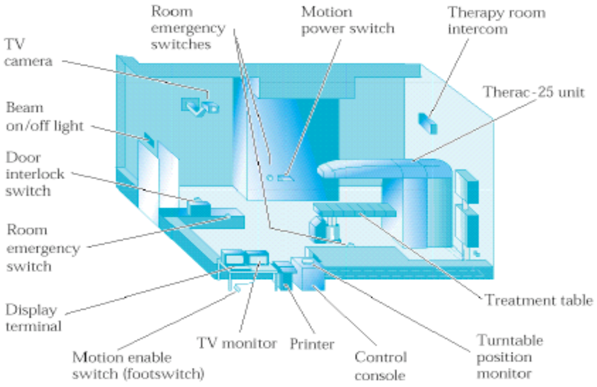
- System failures
  - Communications
    - Telephone, online, and broadcast services
    - Cellular, paging, long-distance network
    - Broadband outages
  - Business
    - Inventory and management software
  - Financial
    - Stock exchange, brokerages, banks, etc.
  - Transportation
    - Reservations, ticketing, and baggage handling
  - Causes
    - Insufficient testing and debugging time
    - Significant changes in specifications (during and after project start)
    - Overconfidence in system
    - Mismanagement of the project

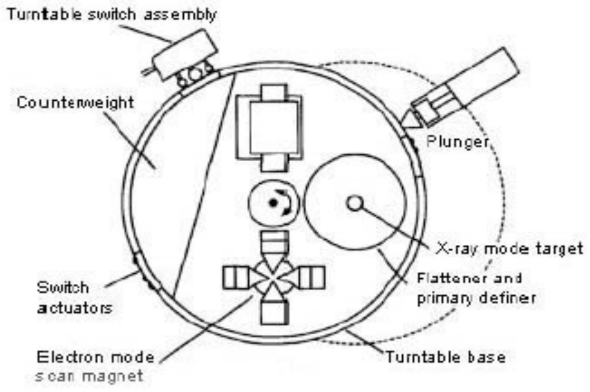
What can go wrong?

- Safety-Critical Applications
  - Uses
    - Medicine, health-services
    - Power plants
    - Aircraft
    - Trains
    - Automated Factories
    - Military applications
  - Causes of error:
    - Overconfidence
    - Lack of override features
    - Insufficient testing
    - Poor HCI
    - Sheer complexity of system (both computer and environment)

Case Study: Therac-25

- Produced by the Atomic Energy of Canada Limited (AECL)
- Introduced in 1983 as a successor model to the Therac-6 and Therac-20
- This new model provided full computer control, which itself was intended to provide several benefits
  - Faster setup by operators, hence more treatments per day.
  - Software checks replaced earlier hardware interlocks




Turntable switch assembly
Counterweight
Plunger
X-ray mode target
Flattener and primary definer
Switch actuators
Electron mode scan magnet
Turntable base

Detour: radiotherapy

- External Beam Therapy: idea
  - Destroy tumours by focusing energy at a specific region
  - Insight: to minimize damage to surrounding tissue, beam “rotates” around patient.
  - Simulation, treatment planning, treatment delivery
  - Radiation oncologist, radiation physicist, dosimetrist and radiation therapist are all involved
  - Radiation therapist controls the machine
  - Actual beams, radiation dosage, etc. are planned out
  - Linear accelerators or cobalt machines are used to generate radiation.

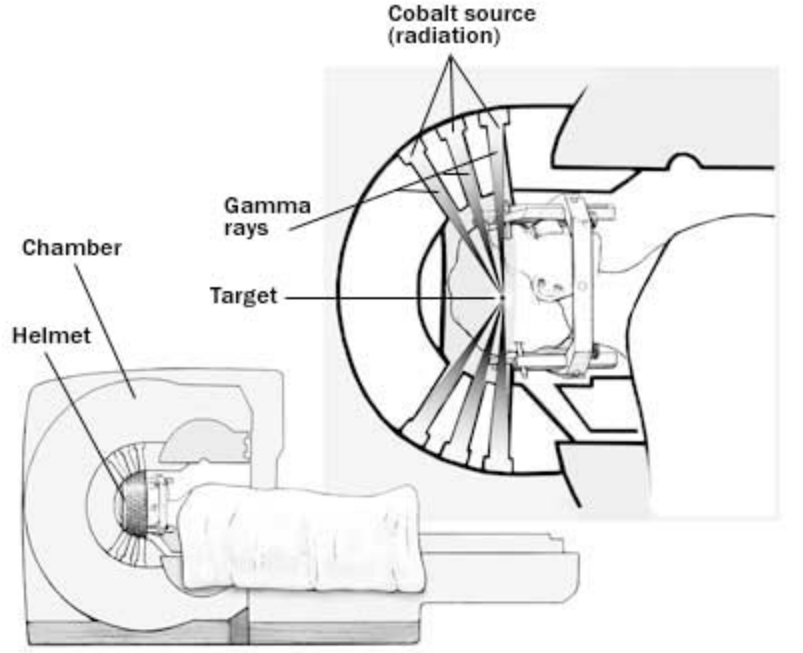


Image of “gamma-knife radiosurgery”

Case Study: Therac-25

- Overdosed 6 patients over a two year period
  - First case occurred in Hamilton, Ontario
  - Normal radioactive dosages is 100-200 rads
  - To compare:
    - Dental x-ray: 0.02 rads
    - Chest x-ray (two views): 0.02 – 0.07 rads
    - CT scan of head or chest: 1 rad
  - Overdoses were estimated at 13,000 and 25,000 rads
    - Given to six people
    - Three of the six people died

Case Study: Therac-25

- Multiple causes of failure (many interrelated):
  - Poor safety design
  - Insufficient testing and debugging
  - Software errors
  - Lack of safety interlocks (hardware)
  - Overconfidence
  - Inadequate reporting / investigation of accidents
- Therac-25 machine continued to be used afterwards but after retrofitted with hardware interlocks

Case Study: Patriot Missile Failure

- February 1991, During first Gulf War
  - Patriot Missile battery in Dharan, Saudi Arabia, failed to intercept incoming Iraqi Scud missle
  - Scud struck an American Army barracks
  - 28 soldiers killed
- Reported cause of failure
  - Inaccurate calculation of “time since boot”
  - Due to computer-arithmetic errors

Case Study: Patriot Missile Failure

- Technical details
  - System's internal clock measured time in units of 1/10th of a second
    - That is, a time of 1000 units corresponded to 100 seconds
  - To obtain time in seconds, units were multiplied by 1/10th.
  - Multiplication was performed in a 24-bit fixed-point register
  - Binary expansion of 1/10 is 0.00011001100110011 …
  - Therefore some truncation is necessary
- System had been up for 100 hours
- Error in calculation was 0.34 seconds
- Scud missile travels at 1,676 meters per second
  - 0.34 seconds therefore equals about half a kilometer

Interface Design Perspective

- To the user “the interface is the system”
  - Understanding of the system operation is based on its interface
  - If this conceptual model is wrong, errors result
- Human attention is incomplete – will not notice every detail, every notification
- Good design will help prevent human error and help users recover from errors

Interface Design Perspective

- Beware of “response chaining”
- Example: Nurse accidently turned off alarm on monitoring for critically ill patient
  - Normal operation – set alarm & then progress through a series of confirmation screens by pressing enter
  - Action became automatic (tap tap tap) – failed to notice warning that alarm was off

What goes wrong?

- Computer systems fail because:
  - The job they are doing is inherently difficult
  - The job is sometimes done poorly
- Compounding the reliability issue
  - Developers and users are overconfident in the system
  - Re-used system software may not work correctly in different environments
- Intellectual overload
  - Large bodies of code are more difficult to understand than smaller ones

Q: How can we increase reliability & safety?

- As developers?
- As system managers?
- As users?

Increasing reliability and safety

- Be professional and responsible:
  - Follow good software-engineering practices
    - What do you believe constitutes “good”?
  - Construct well-designed user interfaces + take human factors into account
  - Include built-in redundancy
  - Incorporate self-checking where appropriate
  - Follow good testing principles and techniques

Increasing reliability and safety

- Law and regulation
  - Civil and criminal penalties
    - to recover from loss due to faulty or unsafe systems
    - to provide incentives to produce reliable and safe systems
  - Warranties
    - to guarantee a certain level of quality
  - Provincial or Federal regulations
    - E.g. Database accuracy regulations - to protect the public from inaccurate information maintained by private companies and government
  - Mandatory licensing of software developers
    - to ensure proper training, competency, and continuing education
    - Q: Do you think this is a good idea?

Computer Models

- Points to consider
  - Models are simplifications of either physical or intangible systems
  - Those who design and develop models must be honest and accurate with results
  - Computer professionals and the general public must be able to evaluate the claims of the developers

Q: Give an example of a computer model. How accurate is it?

Computer models

- Evaluating models
  - Why models might not be accurate:
    - Developers have incomplete knowledge of the system being modeled
    - Data might be incomplete or inaccurate
    - Power of the computer might be inadequate
    - Variables are difficult to quantify (numerically)
    - Political and economic motivation to distort results
  - Two contrasting examples:
    - Car-crash models
    - Climate models

Computer models

- Car-crash models
  - How well do the modelers understand the system or material being studied (or both)?
  - How accurate and complete are the data?
  - What are the assumptions and simplifications of the model?
  - Do the results or predictions correspond with results in the real world?
- Climate models
  - (Same questions)

Perspectives

- Failures
  - What are “acceptable rates” of failures?
  - How accurate should software be?
- Dependence
  - How dependent on computer systems are our ordinary activities?
  - How useful are computer systems to our ordinary activities?
- Risks and progress
  - How do new technologies become safer?
  - Can progress in software safety keep up with the pace of change in computer technology?

Risk is not new

- Risks existed with older technologies
  - Cars, airplanes, trains…
  - Steam engines…
  - Elevators…
- Computers change the types and scope of risk and the allocation of blame
  - Larger, more complex systems
  - Computers make decisions