Chapter 8: Computer Reliability
Chapter Overview

• Introduction
• Data entry or data retrieval errors
• Software and billing errors
• Notable software system failures
• Therac-25
• Computer simulations
• Software engineering
• Software warranties and vendor liability
8.1 Introduction

- Computer systems are sometimes unreliable
  - Erroneous information in databases
  - Misinterpretation of database information
  - Malfunction of embedded systems

- Effects of computer errors
  - Inconvenience
  - Bad business decisions
  - Fatalities
8.2 Data-Entry or Data-Retrieval Errors
Two Kinds of Data-related Failure

- A computerized system may fail because wrong data entered into it
- A computerized system may fail because people incorrectly interpret data they retrieve
Disfranchised Voters

- November 2000 general election
- Florida disqualified thousands of voters
- Reason: People identified as felons
- Cause: Incorrect records in voter database
- Consequence: May have affected election’s outcome
False Arrests

- Sheila Jackson Stossier mistaken for Shirley Jackson
  - Arrested and spent five days in detention
- Roberto Hernandez mistaken for another Roberto Hernandez
  - Arrested twice and spent 12 days in jail
- Terry Dean Rogan arrested after someone stole his identity
  - Arrested five times, three times at gun point
Accuracy of NCIC Records

- March 2003: Justice Dept. announces FBI not responsible for accuracy of NCIC information
- Exempts NCIC from some provisions of Privacy Act of 1974
- Should government take responsibility for data correctness?
Dept. of Justice Position

• Impractical for FBI to be responsible for data’s accuracy
• Much information provided by other law enforcement and intelligence agencies
• Agents should be able to use discretion
• If provisions of Privacy Act strictly followed, much less information would be in NCIC
• Result: fewer arrests
Position of Privacy Advocates

- Number of records is increasing
- More erroneous records → more false arrests
- Accuracy of NCIC records more important than ever
Act Utilitarian Analysis: Database of Stolen Vehicles

- Over 1 million cars stolen every year
- Just over half are recovered, say 500,000
- Assume NCIC is responsible for at least 20%
- 100,000 cars recovered because of NCIC
- Benefit of $5,000 per car (owner gets car back; effects on national insurance rates; criminal doesn’t profit)
- Total value of NCIC stolen vehicle database: $500,000/year
- Only a few stories of false arrests
- Assume 1 false arrest per year (probably high)
- Assume harm caused by false arrest $55,000 (size of award to Rogan)
- Benefit surpasses harm by $445,000/year
- Conclusion: Good to have NCIC stolen vehicles database
8.3 Software and Billing Errors
Errors When Data Are Correct

- Assume data correctly fed into computerized system
- System may still fail if there is an error in its programming
Errors Leading to System Malfunctions

- Qwest sent incorrect bills to cell phone customers
- Faulty USDA beef price reports
- U.S. Postal Service returned mail addressed to Patent and Trademark Office
- Spelling and grammar error checkers increased errors
- New York City Housing authority overcharged renters
- About 450 California prison inmates mistakenly released
Errors Leading to System Failures

- Ambulance dispatch system in London
- Chicago Board of Trade
- BMW limousine
- Japan’s air traffic control system
- Los Angeles County + USC Medical Center laboratory computer system
- Comair’s Christmas Day shutdown
- Boeing 777
Comair Cancelled All Flights on Christmas Day, 2004
Analysis: E-Retailer Posts Wrong Price, Refuses to Deliver

- Amazon.com in Britain offered iPaq for £7 instead of £275
- Orders flooded in
- Amazon.com shut down site, refused to deliver unless customers paid true price
- Was Amazon.com wrong to refuse to fill the orders?
Rule Utilitarian Analysis

• Imagine rule: A company must always honor the advertised price
• Consequences
  – More time spent proofreading advertisements
  – Companies would take out insurance policies
  – Higher costs $\rightarrow$ higher prices
  – All consumers would pay higher prices
  – Few customers would benefit from errors
• Conclusion
  – Rule has more harms than benefits
  – Amazon.com did the right thing
Kantian Analysis

- Buyers knew 97.5% markdown was an error
- They attempted to take advantage of Amazon.com’s stockholders
- They were not acting in “good faith”
- Buyers were in the wrong, not Amazon.com
8.4 Notable Software System Failures
Patriot Missile

- Designed as anti-aircraft missile
- Used in 1991 Gulf War to intercept Scud missiles
- One battery failed to shoot at Scud that killed 28 soldiers
- Designed to operate only a few hours at a time
- Kept in operation > 100 hours
- Tiny truncation errors added up
- Clock error of 0.3433 seconds → tracking error of 687 meters
Patriot Missile Failure

Figure from SCIENCE 255:1347. Copyright ©1992 by The American Association for the Advancement of Science. Reprinted with permission.
Ariane 5

- Satellite launch vehicle
- 40 seconds into maiden flight, rocket self-destructed
  - $500 million of uninsured satellites lost
- Statement assigning floating-point value to integer raised exception
- Exception not caught and computer crashed
- Code reused from Ariane 4
  - Slower rocket
  - Smaller values being manipulated
  - Exception was impossible
AT&T Long-Distance Network

- Significant service disruption
  - About half of telephone-routing switches crashed
  - 70 million calls not put through
  - 60,000 people lost all service
  - AT&T lost revenue and credibility

- Cause
  - Single line of code in error-recovery procedure
  - Most switches running same software
  - Crashes propagated through switching network
AT&T Long Distance Network Failure
Robot Missions to Mars

• Mars Climate Orbiter
  – Disintegrated in Martian atmosphere
  – Lockheed Martin design used English units
  – Jet Propulsion Lab design used metric units

• Mars Polar Lander
  – Crashed into Martian surface
  – Engines shut off too soon
  – False signal from landing gear
Denver International Airport

- BAE built automated baggage handling system
- Problems
  - Airport designed before automated system chosen
  - Timeline too short
  - System complexity exceeded development team’s ability
- Results
  - Added conventional baggage system
  - 16-month delay in opening airport
  - Cost Denver $1 million a day
Tokyo Stock Exchange

• First day of trading for J-Com
• Mizuho Securities employee mistakenly enters order to sell 610,000 shares at 1 yen, instead of 1 share at 610,000 yen
• Employee overrides computer warning
• After sell order posted on exchange’s display board, Mizuho tries to cancel order several times; software bug causes attempts to fail
• Mizuho loses $225 million buying back shares
Direct Recording Electronic Voting Machines

- After problems with 2000 election, Congress passed Help America Vote Act of 2002
- HAVA provided money to states to replace punch card voting systems
- Many states used HAVA funds to purchase direct recording electronic (DRE) voting machines
- Brazil and India have run national elections using DRE voting machines exclusively
- In November 2006 1/3 of U.S. voters used DRE voting machines
Diebold Electronic Voting Machine
Issues with DRE Voting Machines

- Voting irregularities
  - Failure to record votes
  - Overcounting votes
  - Misrecording votes
- Lack of a paper audit trail
- Vulnerability to tampering
- Source code a trade secret, can’t be examined
- Possibility of widespread fraud through malicious programming
8.5 Therac-25
Genesis of the Therac-25

- AECL and CGR built Therac-6 and Therac-20
- Therac-25 built by AECL
  - PDP-11 an integral part of system
  - Hardware safety features replaced with software
  - Reused code from Therac-6 and Therac-20
- First Therac-25 shipped in 1983
  - Patient in one room
  - Technician in adjoining room
Chronology of Accidents and AECL Responses

- Marietta, Georgia (June 1985)
- Hamilton, Ontario (July 1985)
- First AECL investigation (July-Sept. 1985)
- Yakima, Washington (December 1985)
- Tyler, Texas (March 1986)
- Second AECL investigation (March 1986)
- Tyler, Texas (April 1986)
- Yakima, Washington (January 1987)
- FDA declares Therac-25 defective (February 1987)
Software Errors

- Race condition: order in which two or more concurrent tasks access a shared variable can affect program’s behavior
- Two race conditions in Therac-25 software
  - Command screen editing
  - Movement of electron beam gun
Race Condition Revealed by Fast-typing Operators
Race Condition Caused by Counter Rolling Over to Zero
Post Mortem

- AECL focused on fixing individual bugs
- System not designed to be fail-safe
- No devices to report overdoses
- Software lessons
  - Difficult to debug programs with concurrent tasks
  - Design must be as simple as possible
  - Documentation crucial
  - Code reuse does not always lead to higher quality
- AECL did not communicate fully with customers
Moral Responsibility of the Therac-25 Team

- Conditions for moral responsibility
  - Causal condition: actions (or inactions) caused the harm
  - Mental condition
    - Actions (or inactions) intended or willed -OR-
    - Moral agent is careless, reckless, or negligent

- Therac-25 team morally responsible
  - They constructed the device that caused the harm
  - They were negligent
Postscript

• Computer errors related to radiation machines continue to maim and kill patients

• Investigation by *The New York Times*
8.6 Computer Simulations
Uses of Simulations

• Simulations replace physical experiments
  – Experiment too expensive or time-consuming
  – Experiment unethical
  – Experiment impossible
• Model past events
• Understand world around us
• Predict the future
Simulations Predict Path and Speed of Hurricanes

Courtesy of NASA
Validating Simulations

- Verification: Does program correctly implement model?
- Validation: Does the model accurately represent the real system?
- Validation methods
  - Make prediction, wait to see if it comes true
  - Predict the present from old data
  - Test credibility with experts and decision makers
Validating a Model

<insert Figure 8.8>
Validation by “Predicting the Present”
8.7 Software Engineering
Specification

- Determine system requirements
- Understand constraints
- Determine feasibility
- End products
  - High-level statement of requirements
  - Mock-up of user interface
  - Low-level requirements statement
Development

- Create high-level design
- Discover and resolve mistakes, omissions in specification
- CASE tools to support design process
- Object-oriented systems have advantages
- After detailed design, actual programs written
- Result: working software system
Validation (Testing)

• Ensure software satisfies specification
• Ensure software meets user’s needs
• Challenges to testing software
  – Noncontinuous responses to changes in input
  – Exhaustive testing impossible
  – Testing reveals bugs, but cannot prove none exist
• Test modules, then subsystems, then system
Software Quality Is Improving

- Standish Group tracks IT projects
- Situation in 1994
  - 1/3 projects cancelled before completion
  - 1/2 projects had time and/or cost overruns
  - 1/6 projects completed on time and on budget
- Situation in 2009
  - 1/4 projects cancelled
  - 5/12 projects had time and/or cost overruns
  - 1/3 projects completed on time and on budget
Success of IT Projects Over Time

- In 1994:
  - On time/On budget: 1/4
  - Time/Cost overruns: 1/2
  - Canceled: 1/4

- In 2009:
  - On time/On budget: 1/4
  - Canceled: 1/2
  - Time/Cost overruns: 1/4
8.8 Software Warranties and Vendor Liability
Shrinkwrap Warranties

- Some say you accept software “as is”
- Some offer 90-day replacement or money-back guarantee
- None accept liability for harm caused by use of software
Are Software Warranties Enforceable?

• Mass-marketed software and software included in sale of hardware likely to be considered a good by a court of law
• Uniform Commercial Code applies to goods, despite what warranties may say
Key Court Cases

- **Step-Saver Data Systems v. Wyse Technology and the Software Link**
  - Court ruled that provisions of UCC held

- **ProCD v. Zeidenberg**
  - Court ruled shrinkwrap licenses are enforceable

- **Mortenson v. Timberline Software**
  - Court ruled in favor of Timberline and licensing agreement that limited consequential damages
Moral Responsibility of Software Manufacturers

• If vendors were responsible for harmful consequences of defects
  – Companies would test software more
  – They would have to purchase liability insurance
  – Software would cost more
  – Start-ups would be affected more than big companies
  – Less innovation in software industry?
  – Software would be more reliable?

• Making vendors responsible for harmful consequences of defects may be a bad idea, but…

• Consumers should not have to pay for bug fixes