

LECTURE 8: Software Reliability

(Some slides are from M. Quinn, *Ethics for the Information Age*, Pearson © 2013.)

Lecture Overview

- Introduction
- Data-entry or data-retrieval errors
- Software errors
- Notable software system failures
- Therac-25
- Software engineering
- Software warranties

1-2

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

1-3

Data-Entry or Data-Retrieval Errors

14

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

15

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

16

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

1-7

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?

1-8

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

1-9

Position of Privacy Advocates

- Number of records is increasing
- More erroneous records → more false arrests
- Accuracy of NCIC records more important than ever

1-10

Utilitarian Analysis: Database of Stolen Vehicles

- > 1 million cars stolen every year
 - Owners suffer emotional, financial harm
 - Raises insurance rates for all
- Transporting stolen car across a state line
 - Before NCIC, greatly reduced chance of recovery
 - After NCIC, nationwide stolen car retrieval
- At least 50,000 recoveries annually due to NCIC
- Few stories of faulty information causing false arrests
- Benefit > harm → Creating database the right action

1-11

Software Errors

1-12

Errors When Data Are Correct

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

1-13

Software Errors Leading to System Malfunctions

- 2001 Qwest sent incorrect bills to cell phone customers
 - 1.4% of customers affected, charged \$600/minute
- Faulty USDA beef price reports
 - Errors cost between \$15-\$20 million
- 1996 U.S. Postal Service error caused mail addressed to Patent and Trademark Office to be returned for 2 weeks
- 2009 New York City Housing authority overcharged renters
- 2010 About 450 California prison inmates with "high risk of violence" mistakenly released

1-14

Software Errors Leading to System Failures

- 1992 Ambulance dispatch system in London
 - 20 people died
- 1998 Chicago Board of Trade suspended trading
- 2003 Thailand's minister trapped in BMW limousine
- 2003 Japan's air traffic control system failed as well as backup system
- 2003 Los Angeles County + USC Medical Center new laboratory computer system, stopped all ambulances
- 2004 Comair's Christmas Day shutdown cancelled 1,100 flights (Delta Air Lines)
- 2005 Boeing 777 lost control of auto-pilot

1-15

Ethical Analysis: E-Retailer Posts Wrong Price, Refuses to Deliver

- 2003 Amazon.com in Britain offered iPaq handheld computer 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?

1-16

Rule Utilitarian Analysis

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

1-17

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 did something wrong

1-18

Notable Software System Failures

1-19

Patriot Missile 1991

- 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

1-20

Ariane 5 1996

- 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

1-21

**AT&T Long-Distance Network
1990**

- 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

1-22

**Robot Missions to Mars
1999**

- 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

1-23

**Denver International Airport
1993**

- 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

1-24

Tokyo Stock Exchange 2005

- First day of trading for J-Com
- Mizuho Securities employee mistakenly entered order to sell 610,00 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 tried to cancel order several times; software bug caused attempts to fail
- Mizuho lost \$225 million buying back shares

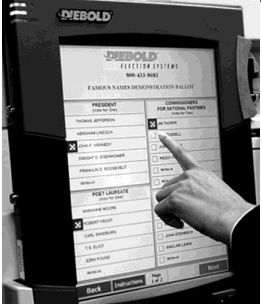
1-25

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

1-26

Diebold Electronic Voting Machine



© AP Photo/Rogelio Solis

1-27

Issues with DRE Voting Machines

- Voting irregularities
 - Failure to record votes 2002
 - Overcounting votes 2003
 - Misrecording votes 2004 & 2006
- Lack of a paper audit trail
- Vulnerability to tampering
 - CS prof Herbert Thompson examined centralized Diebold machine: lacked authentication mechanism
 - Inserted 5 lines of code and switched 5k votes from one candidate to another
- Source code a trade secret, can't be examined
- Possibility of widespread fraud through malicious programming

1-28

CASE STUDY Therac-25

1-29

Why this case study?

- Example of system where safety relies solely upon the quality of its embedded software
- Harms are very serious
- Although old, it is very well documented

Describe the Therac-25

Therac-25

- Medical machine generating radiation from linear accelerator
- Radiation treatment commonly used for cancer patients
 - between 50-60% today
- 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 11 systems in 1983
 - Patient in one room
 - Technician in adjoining room
- Very unreliable
 - 40 malfunctions/day
 - Resulted in 6 massive overdoses with three deaths
 - 1987 operations suspended

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)

1-33

What caused the safety failure?

The Socio-Technical System	
The Machine <ul style="list-style-type: none"> Supporting Systems (video, audio, etc.) Hardware Software Systems 	Hospitals and Clinics <ul style="list-style-type: none"> Doctors, Medical Physicists Management, User Groups Operators, Reporting Procedures
Atomic Energy Canada, Limited <ul style="list-style-type: none"> Management, Reporting Procedures, Design Teams, Sales Staff, Support and Field Engineers 	Government Medical Device Regulation <ul style="list-style-type: none"> Food and Drug Administration Canadian Radiation Protection Bureau Reporting Procedures

User Interface

PATIENT NAME: TEST	BEAM TYPE: X ENERGY (KeV):	A	1
TREATMENT MODE: FIX		25	
	ACTUAL	PERSCRIBED	
UNIT RATE/MINUTE	0	200	
MONITOR UNITS	50.00	200	
TIME (MIN)	0.27	1.00	
GANTRY ROTATION (DEG)	0.0	0	VERIFIED
COLLIMATOR ROTATION (DEG)	359.2	359	VERIFIED
COLLIMATOR X (CM)	14.2	14.3	VERIFIED
COLLIMATOR Y (CM)	27.2	27.3	VERIFIED
WEDGE NUMBER	1	1	VERIFIED
ACCESSORY NUMBER	0	0	VERIFIED
DATE: 84 OCT 26	SYSTEM: BEAM READY	OP.MODE: TREAT	AUTO
TIME: 12:55.8	TREAT: TREAT PAUSE	X.RAY	173/77
OPR ID: T25V02-R03	REASON: OPERATOR	COMMAND:	

Causes of Safety Failure

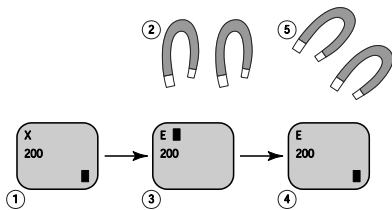
- Programming errors
- Poor human computer interaction design
- Inadequate safety engineering
- Lax culture of safety in the manufacturing organization
- Inadequate reporting structure at the company level and as required by the U.S. government

Software Errors

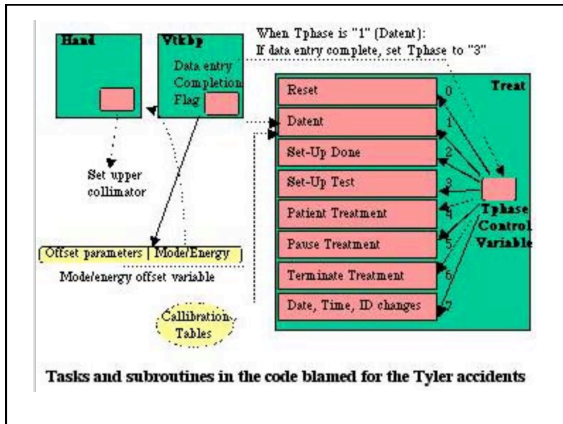
- Race condition: order in which two or more concurrent tasks access a shared variable can affect program's behavior
- Two race conditions in Therac-25 software
 - Command screen editing
 - Movement of electron beam gun
- Extremely difficult to diagnose and debug race condition

1-38

Race Condition Revealed by Fast-typing Operators



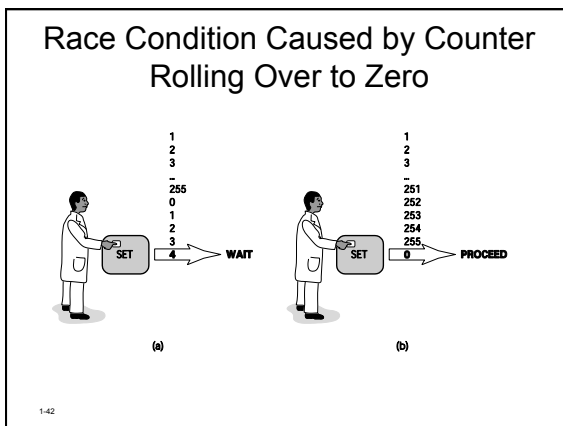
1-39



Code for Detent Task

```

Datent:
  if mode/energy specified then
    begin
      calculate table index
      repeat
        fetch parameter
        output parameter
        point to next parameter
      until all parameters set
      call Magnet
      if mode/energy changed then return
    end
  if data entry is complete then set Tphase to 3
  if data entry is not complete then
    if reset command entered then set Tphase to 0
  return
    
```



Post Mortem

- AECL focused on fixing individual bugs
 - System bugs are *interactions*
- System not designed to be fail-safe
 - No single point of failure should create catastrophe
 - Need fail-safe solutions that are not software
- No hardware or software 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
 - AECL told physicists in Washington & Texas overdose was impossible despite being sued by overdose patient in Georgia

1-43

Were the Therac-25 designers and producers morally responsible?

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

1-45

Responsibilities of the Programmer

- To make superiors aware of the dangers inherent in doing safety interlocks only in the software
 - Limits of software only
 - Redundancy
 - Over-ride testing
- Knowledge of professional software practices
 - Using unprotected memory
 - Improper initialization
 - Thoroughly test software for many possible conditions
 - Human-computer interaction as a system

Postscript

- Computer errors related to radiation machines continue to maim and kill patients
- Investigation by *The New York Times*
 - Scott Jerome-Parks, New York (2006)
 - 3 overdoses from linear accelerator, died
 - Alexandra Jn-Charles, New York (2006)
 - 27 days radiation overdoses, died

1-47

Improving Software Reliability by Software Engineering Practices

Software Engineering Practices

- Specification using system requirements
- Development
 - CASE tools
 - Object-oriented systems have advantages
 - Modular design and implementation
- Validation (Testing)

Validation (Testing)

- Ensure software satisfies specification
- Ensure software meets user's needs
- Challenges to testing software
 - Non-continuous responses to changes in input
 - Exhaustive testing impossible
 - Testing reveals bugs, but cannot prove none exist
- Test modules, then subsystems, then system
- Simulation
 - Validating software by prediction

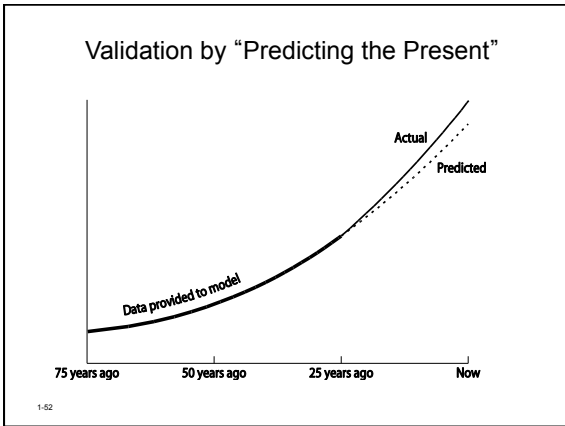
1-50

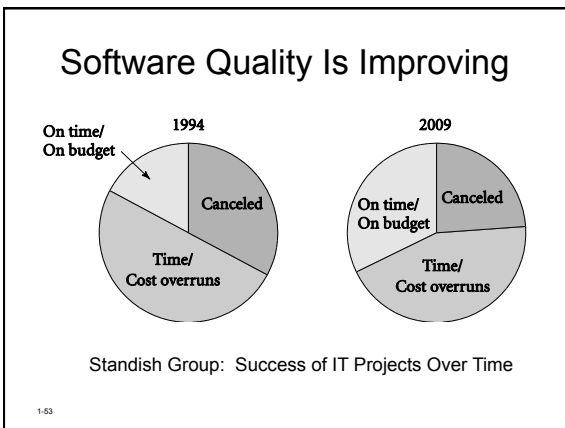
Validation by Comparing Predicted and Actual Outcomes



Courtesy of Daimler AG

1-51





Software Warranties

1-64

Shrinkwrap Non-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

1-55

Are Software “Warranties” Enforceable?

- No: company still responsible despite warranty
 - Article 2 of Uniform Commercial Code
 - Magnuson-Moss Warranty Act
 - 1987 *Step-Saver Data Systems v. Wyse Technology and The Software Link*
- Yes: company not responsible
 - 1994 *ProCD, Inc. v. Zeidenberg*
 - 1993 *Mortensen v. Timberline Software*

1-56

Moral Responsibility of Software Manufacturers

- If vendors were responsible for harmful consequences of defects
 - Companies would test software more
 - They would 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 wrong, but...
- Consumers should not have to pay for bug fixes

1-57

Overall Conclusions

- Data can be incorrectly entered
- Software inherently has limitations
 - Design (model failure)
 - Software Implementation (bugs)
 - Software Testing (impossible to catch all bugs)
 - Operational and maintenance failures
- Design, implement, test and train for FAILURE
