Today's Seattle Times (11/23/2011)

"The bottom line is the BCS is flawed," [Stanford football coach] Shaw said. "They themselves know it, which is why they've proposed a lot of changes going forward. All I've heard all year is the computers don't like Stanford. The computers haven't programed [sic] themselves."

CSE 331
SOFTWARE DESIGN & IMPLEMENTATION
SOFTWARE DISASTERS

Autumn 2011

Today's educational objective

- Hear about some classic "risks" related to software engineering – and learn about some of the underlying problems that caused them
- The primary intent is to help you understand your responsibilities as a software engineering

comp.risks

- Since 1985, Peter Neumann has moderated a Usenet newsgroup named `comp.risks`
- "Forum On Risks To The Public In Computers And Related Systems"
  - Safety, privacy, stupidity, etc.
- A web version is the Risks Digest
  - `http://catless.ncl.ac.uk/Risks`

Software safety

- *Safeware: System Safety and Computers*. Nancy Leveson. Addison Wesley (1995).
- Software safety is not the same as software reliability
  - Reliability concerns the probability that a system will perform in a certain way
  - Safety concerns what happens when a system fails

comp.risks first year: some 1985 topics

Legend: ! = Loss of Life; * = Potentially Life-Critical; $ = Loss of Money/Equipment; S = Security/Privacy/Integrity Flaw

- *$ Mariner 1: Atlas booster launch failure DO 100 I=1.10 (not 1,10) (SEN 8 5)
- *$ Mariner 18: aborted due to missing NOT in program (SEN 5 2)
- *$ F18: plane crashed due to missing exception condition, pilot OK (SEN 6 2)
- *$ El Dorado brake computer bug caused recall of all El Dorados (SEN 4 4)
- * Second Space Shuttle operational simulation: tight loop upon cancellation of an attempted abort; required manual override (SEN 7 1)
- * Gemini V 100mi landing err, prog ignored orbital motion around sun (SEN 9 1)
- * F16 simulation: plane flipped over whenever it crossed equator (SEN 5 2)
- * F16 simulation: upside-down F16 deadlock over left vs. right roll (SEN 9 5)
- * SF BART train doors sometimes open on long legs between stations (SEN 8 5)
- * IRS reprogramming cost USA interest on at least 1,150,000 refunds (SEN 10 3)
- Santa Clara prison data system (inmate altered release date) (SEN 10 1).
- Computerized time-bomb inserted by programmer (for extortion?) (10 3)
- *$ Colorado River flooding in 1983, due to faulty weather data and/or faulty model; too much water was kept dammed prior to spring thaws.
- $ 1979 AT&T program bug downed phone service to Greece for months (SEN 10 3)
- Quebec election prediction gave loser big win [1981] (SEN 10 2, p. 25-26)
- SW vendor rigs elections? (David Burnham, NY Times front page, 29 July 1985)
- Vancouver Stock Index lost 574 points over 22 months -- roundoff (SEN 9 1)

1985: more

- ! Michigan man killed by robotic die-casting machinery (SEN 10 2)
- ! Chinese computer builder electrocuted by his smart computer after he built a newer one. "Jealous Computer Zaps its Creator"! (SEN 10 1)
- * FAA Air Traffic Control: many computer system outages (e.g., SEN 5 3)
- * ARPANET ground to a complete halt [27 Oct 1980] (SEN 6 1 [Rosen])
- *$ Ford Mark VII wiring fires: flaw in computerized air suspension (SEN 10 3)
- $S Harrah's $1.7 Million payoff scam -- Trojan horse chip (SEN 8 5)
- $ Great Northeast power blackout due to threshold set-too-low being exceeded
- - SF Muni Metro: Ghost Train reappeared, forcing manual operation (SEN 8 3)
- *$ 8080 control system dropped bits and boulders from 80 ft conveyor (SEN 10 2)
- S 1984 Rose Bowl hoax, scoreboard takeover ("Cal Tech vs. MIT") (SEN 9 2)

1985: SDI ("Star Wars")

7

- [comp.risks] "The Strategic Defense Initiative (popularly known as Star Wars) is considering the feasibility of developing what is probably the most complex and most critical system ever contemplated"
- [NY Times] Washington, July 11 - A computer scientist has resigned from an advisory panel on antimissile defense, asserting that it will never be possible to program a vast complex of battle management computers reliably or to assume they will work when confronted with a salvo of nuclear missiles.

  The scientist, David L. Parnas, a professor at the University of Victoria in Victoria, British Columbia, who is consultant to the Office of Naval Research in Washington, was one of nine scientists asked by the Strategic Defense Initiative Office to serve … on the "panel on computing in support of battle management".

  Professor Parnas, an American citizen with secret military clearances, said in a letter of resignation and 17 accompanying memorandums that it would never be possible to test realistically the large array of computers that would link and control a system of sensors, antimissile weapons, guidance and aiming devices, and battle management stations.

  Nor, he protested, would it be possible to follow orthodox computer program-writing practices in which errors and "bugs" are detected and eliminated in prolonged everyday use. ...

  "I believe," Professor Parnas said, "that it is our duty, as scientists and engineers, to reply that we have no technological magic that will accomplish that. The President and the public should know that." ...

A quick on-line view of risks topics

8

- How do they change – and stay the same – over 25 years?
- http://catless.ncl.ac.uk/Risks/

A fun one

9

A short news article in a weekend newspaper told how a woman was woken by a mysterious phone call at 4:30am every day. She reported it to British Telecom who monitored the line for several months to track down the phone pest ... and eventually discovered that the calls were due to a programming error in one of their own test computers!

Another fun one

10

A woman who was phoned repeatedly by a public lavatory asking her to fill it with cleaning fluid had to ask British Telephone to put a stop to the calls. ... About 15% of all nuisance calls are caused by errors, most of which are traceable to faulty programming, according to a BT spokesperson. The most common type of computer-controlled nuisance call is from soft drink vending machines which need refilling.

A not fun one: Therac-25

11

- In the mid-1980's, the Therac-25 computer-controlled radiation therapy machine massively overdosed six people
- Electromechanical interlocks are common on such systems, intended to ensure that overdosing is avoided
- On the Therac-25, the mechanical interlocks were removed and the equivalent interlocks built in software

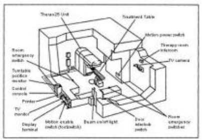

Figure 1. Typical Therac-25 facility

The problem I

12

- It's complex, but here's the basic idea
  - Leveson & Turner, *IEEE Computer* July 1993.
- Machine operators enter treatment plans on a console
- Operators complained that entering the plans was too time consuming
- It was modified (before initial release) to use carriage returns to copy treatment data

The problem II

- The data entry module was controlled by the keyboard handler, a mode indicated by a shared variable, and some calibration tables
  - The mode also influences the setting of the collimator (a part of the dosing mechanism)
- In short (and there were other problems), if the operator hit a bunch of carriage returns quickly, the interlocks could fail, overdosing the patient
- Key observation: the people who tested the system were not expert operators – the expert operators type much more quickly, which was needed to surface the problem

High level observations

- Why remove the hardware interlocks?
- The operators had been convinced by the manufacturer that the system had been built to be safe
- The changes to simplify the operators' job were perceived as entirely functional—the timing issues weren't studied carefully
- As with many accidents, it's a collection of small things more than one big thing
- And it could be fixed by hardware (retaining the interlocks) as well as by software

FDA recalls I

- A programmable implanted pacemaker that was inadvertently reprogrammed by emitted magnetic fields from an anti-theft device in a retail store; the patient's weakened heart was unable to stand the increased pace.
- An infusion-pump (used for insulin) had a software problem which caused the infusion of insulin or dextrose to be delivered at the maximum rather than the lower intended rate. This occurred when certain valid data was entered according to user instructions.

FDA recalls II

- A programmable pacemaker "locked-up" when being reset by an external programming device. Luckily this occurred in a doctor's office, and the doctor was able to revive the patient.
- A multiple-patient monitoring system was recalled because the software got patients' names mixed up with the wrong data.
- An algorithm was incorrectly programmed in a diagnostic lab instrument that caused certain patient data to be reported erroneously as all zeros.

Ariane 5 failure

Ariane 5 failure

- Inaugural flight in June 1996 blew up very early in flight, losing a $400M scientific payload plus the rocket
- Central issue: reuse of inertial reference system from Ariane 4; in Ariane 5, this software didn't handle an exception correctly
  - Conversion from 64-bit floating point to 16-bit integer was assumed not to overflow; but with a new flight model, it did
  - The exception caused a dual hardware shutdown since it had been assumed the software wouldn't fail

Different views

- It was a software bug and it should have been caught
- It was a management problem, making a bad risk tradeoff
  - For schedule and cost reasons, a decision had been explicitly made not to simulate this and a collection of other cases
- There is much truth to both of these

Voting?

The Yamhill County clerk discovered a computer programming error that reverses the election results of the county's district attorney's race. [The incumbent] didn't lose in the November election -- he won by a landslide. Clerk Charles Stern said the computer error occurred because the program failed to list the candidates in alphabetical order, as they were on the ballot. Mercer had supposedly lost to Bernt "Owl" Hansen, 16,539 votes to 8,519 votes

Voting… very serious

- Diebold and Princeton video

George or Benedict?

Humor

[The] sash cord broke on one of the voting machines in the precinct where I was working as a poll worker. ... The Judge ... took the machine out of service and sent for a technician ... As [the technician] came out of the polling station another poll worker asked: "Well, is the machine fixed?" The technician replied as he hurried on to his next assignment: "Now, now, we don't like to use the 'F-word' on Election Day. The word is 'repaired'."

Denver Airport
Gibbs, *Scientific American* 1994

Denver's new international airport was to be the pride of the Rockies, a wonder of modern engineering. … Even more impressive … is the airport's subterranean baggage-handling system. Tearing like intelligent coal-mine cars along 21 miles of steel track, 4,000 independent "telecars" route and deliver luggage between the counters, gates and claim areas of 20 different airlines. A central nervous system of some 100 computers networked to one another and to 5,000 electric eyes, 400 radio receivers and 56 bar-code scanners orchestrates the safe and timely arrival of every valise and ski bag.

At least that is the plan. For nine months, this Gulliver has been held captive by Lilliputians-errors in the software that controls its automated baggage system. Scheduled for takeoff by last Halloween, the airport's grand opening was postponed until December to allow BAE Automated Systems time to flush the gremlins out of its $193-million system. December yielded to March. March slipped to May. In June the airport's planners, their bond rating demoted to junk and their budget hemorrhaging red ink at the rate of $1.1 million a day in interest and operating costs, conceded that they could not predict when the baggage system would stabilize enough for the airport to open.

To veteran software developers, the Denver debacle is notable only for its visibility. Studies have shown that for every six new large-scale software systems that are put into operation, two others are canceled. The average software development project overshoots its schedule by half; larger projects generally do worse. And some three quarters of all large systems are "operating failures" that either do not function as intended or are not used at all.

Denver Airport: complexity^{nth}

Myerson, NY Times, 1994

- 14x bigger than previously largest system (SFO)
- First intended to serve an entire major airport
- First to have carts only slow down to pick up and drop off bags
- First to be run by a network of desktop computers rather than a mainframe
- First to use radio links
- First to handle oversized bags

System complexity

- Adding complexity in multiple dimensions increases risk – and these added complexities for Denver were not only in the "software" dimension.
- But software is the fall guy: this is fair only in the sense that different software could have avoided most or all of the problems
- Other imaginable approaches might have included
  - A more accurate risk-analysis perhaps along with planned alternatives for likely failures or delays
  - A different co-design perhaps balanced by a reduction in expected features

Failures

- It's a failure when the system doesn't work as expected
  - Even if it works as specified
  - Expectations are not always clear … and they change as users use a system
- Systems can break even when each part works as specified … failures at the interfaces, unexpected stresses, etc.

Most important

- Parnas resigned from the SDI and publicly described why – this generated attention and, even more important, discussion
- When you are "out there" and you see a likely (or even unlikely) flaw to safety, security, etc., it's your professional responsibility to say something … clearly and consistently

Next steps

- Assignment 0
  - Due today 11:59PM
- Assignment 1
  - out later today
  - due Wednesday (10/5) 11:59PM
- Assignment 2
  - out Wednesday (10/5)
    - due in two parts
    - part A on Friday (10/7) 11:59PM
    - part B the following Wednesday (10/12) at 11:59PM
- Lectures
  - Testing and Junit (M)
  - Equality (W)
  - Abstract data types I (F)
  - Abstract data types II (M)