Why Do We Test Software?

Paul Ammann & Jeff Offutt

http://www.cs.gmu.edu/~offutt/softwaretest/
Software is a Skin that Surrounds Our Civilization

Quote due to Dr. Mark Harman
Testing in the 21st Century

• Software defines **behavior**
  – network routers, finance, switching networks, other infrastructure

• Today’s software market:
  – is much bigger
  – is more competitive
  – has more users

• Embedded Control Applications
  – Mobile phones
  – airplanes, air traffic control
  – spaceships
  – watches
  – ovens
  – PDAs
  – memory seats
  – DVD players
  – garage door openers

• Agile processes put increased pressure on testers
  – Programmers must unit test – with no training or education!
  – Tests are key to functional requirements – but who builds those tests?

Industry is going through a revolution in what testing means to the success of software products
• *Bug* is used informally

• Sometimes speakers mean fault, sometimes error, sometimes failure ... often the speaker doesn’t know what it means!

• This class will try to use words that have precise, defined, and unambiguous meanings
• **Software Fault**: A static defect in the software

• **Software Failure**: External, incorrect behavior with respect to the requirements or other description of the expected behavior

• **Software Error**: An incorrect internal state that is the manifestation of some fault
Fault and Failure Example

• A patient gives a doctor a list of symptoms
  – Failures
• The doctor tries to diagnose the root cause, the ailment
  – Fault
• The doctor may look for anomalous internal conditions (high blood pressure, irregular heartbeat, bacteria in the blood stream)
  – Errors
public static int numZero (int[] arr) {
  // Effects: If arr is null throw NullPointerException
  // else return the number of occurrences of 0 in arr
  int count = 0;
  for (int i = 1; i < arr.length; i++) {
    if (arr[i] == 0) {
      count++;
    }
  }
  return count;
}

Fault: Should start searching at 0, not 1

Test 1
[2, 7, 0]
Expected: 1
Actual: 1

Error: i is 1, not 0, on the first iteration
Failure: none

Test 2
[0, 2, 7]
Expected: 1
Actual: 0

Error: i is 1, not 0
Error propagates to the variable count
Failure: count is 0 at the return statement
Spectacular Software Failures

- NASA’s Mars lander: September 1999, crashed due to a units integration fault
- THERAC-25 radiation machine: Poor testing of safety-critical software can cost lives: 3 patients were killed
- Ariane 5 explosion: Very expensive
- Intel’s Pentium FDIV fault: Public relations nightmare

We need our software to be dependable. Testing is one way to assess dependability.

THERAC-25 design

Ariane 5: exception-handling bug: forced self-destruct on maiden flight (64-bit to 16-bit conversion: about 370 million $ lost)
Northeast Blackout of 2003

- 508 generating units and 256 power plants shut down
- Affected 10 million people in Ontario, Canada
- Affected 40 million people in 8 US states
- Financial losses of $6 Billion USD

The alarm system in the energy management system failed due to a software error and operators were not informed of the power overload in the system.
Costly Software Failures

  - Inadequate software testing costs the US alone between $22 and $59 billion annually
  - Better approaches could cut this amount in half

- Huge losses due to web application failures
  - Financial services: $6.5 million per hour (just in USA!)
  - Credit card sales applications: $2.4 million per hour (in USA)

- In Dec 2006, amazon.com’s BOGO offer turned into a double discount

- 2007: Symantec says that most security vulnerabilities are due to faulty software

World-wide monetary loss due to poor software is staggering
Testing in the 21st Century

• More safety critical, real-time software
• Embedded software is ubiquitous ... check your pockets
• Enterprise applications means bigger programs, more users
• Paradoxically, free software increases our expectations!
  – Security is now all about software faults
    – Secure software is reliable software
• The web offers a new deployment platform
  – Very competitive and very available to more users
  – Web apps are distributed

Industry desperately needs our inventions!
Assume $1000 unit cost, per fault, 100 faults

Cost of Late Testing

Software Engineering Institute; Carnegie Mellon University; Handbook CMU/SEI-96-HB-002
A tester’s goal is to eliminate faults as early as possible

- Improve quality
- Reduce cost
- Preserve customer satisfaction