Software Testing and Verification
Overview and V&V Framework

Sanjai Rayadurgam

SENG 5811 – Spring 2025 – Session 1

University of Minnesota
Driven to Discover®

Welcome to SENG 5811

Introductions

Name

Work

Fun facts

Course Overview

Syllabus

Expectations

Assessments

Preliminaries

Why testing

Principles & Processes

Discrete Math Basics

- Software is everywhere
- It ~~can~~ will fail
- Failures have consequences

JAMIE LYNCH
SEPTEMBER 19, 2017

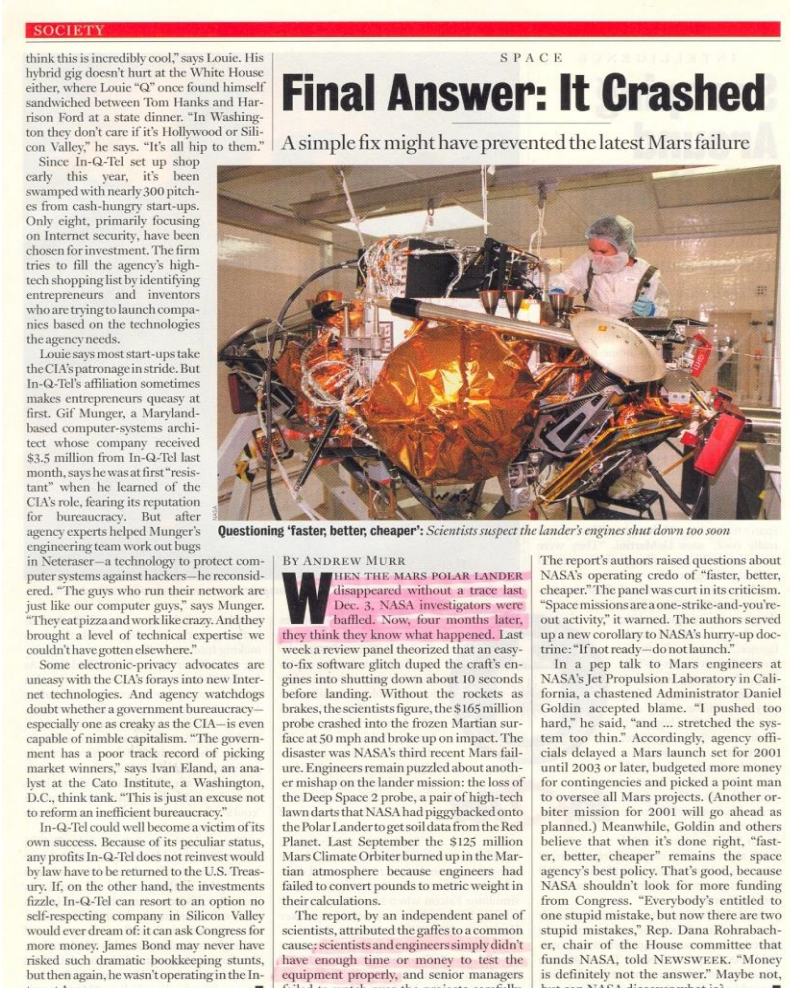
The Worst Computer Bugs in History: Race conditions in Therac-25

Losing $460m in 45 minutes

A computer program mistaking production for a test environment meant that many, many trades, each losing a few cents, ended up costing Knight Capital $460 million. Read more here.

A clue to MNsure's problems found in a similar software precursor

Elizabeth Stawicki St. Paul, Minn. January 21, 2014 10:00 a.m.



SOCIETY

think this is incredibly cool," says Louie. His hybrid gig doesn't hurt at the White House either, where Louie "Q" once found himself sandwiched between Tom Hanks and Harrison Ford at a state dinner. "In Washington they don't care if it's Hollywood or Silicon Valley," he says. "It's all hip to them."

Since In-Q-Tel set up shop early this year, it's been swamped with nearly 300 pitches from cash-hungry start-ups. Only eight, primarily focusing on Internet security, have been chosen for investment. The firm tries to fill the agency's high-tech shopping list by identifying entrepreneurs and inventors who are trying to launch companies based on the technologies the agency needs.

Louie says most start-ups take the CIA's patronage in stride. But In-Q-Tel's affiliation sometimes makes entrepreneurs queasy at first. Gif Munger, a Maryland-based computer-systems architect whose company received $3.5 million from In-Q-Tel last month, says he was at first "resistant" when he learned of the CIA's role, fearing its reputation for bureaucracy. But after agency experts helped Munger's engineering team work out bugs in Neteraser—a technology to protect computer systems against hackers—he reconsidered. "The guys who run their network are just like our computer guys," says Munger. "They eat pizza and work like crazy. And they brought a level of technical expertise we couldn't have gotten elsewhere."

Some electronic-privacy advocates are uneasy with the CIA's forays into new Internet technologies. And agency watchdogs doubt whether a government bureaucracy—especially one as creaky as the CIA—is even capable of nimble capitalism. "The government has a poor track record of picking market winners," says Ivan Eland, an analyst at the Cato Institute, a Washington, D.C., think tank. "This is just an excuse not to reform an inefficient bureaucracy."

In-Q-Tel could well become a victim of its own success. Because of its peculiar status, any profits In-Q-Tel does not reinvest would by law have to be returned to the U.S. Treasury. If, on the other hand, the investments fizzle, In-Q-Tel can resort to an option no self-respecting company in Silicon Valley would ever dream of: it can ask Congress for more money. James Bond may never have risked such dramatic bookkeeping stunts, but then again, he wasn't operating in the In-

SPACE

Final Answer: It Crashed

A simple fix might have prevented the latest Mars failure

Questioning 'faster, better, cheaper': *Scientists suspect the lander's engines shut down too soon*

BY ANDREW MURR

WHEN THE MARS POLAR LANDER disappeared without a trace last Dec. 3, NASA investigators were baffled. Now, four months later, they think they know what happened. Last week a review panel theorized that an easy-to-fix software glitch duped the craft's engines into shutting down about 10 seconds before landing. Without the rockets as brakes, the scientists figure, the $165 million probe crashed into the frozen Martian surface at 50 mph and broke up on impact. The disaster was NASA's third recent Mars failure. Engineers remain puzzled about another mishap on the lander mission: the loss of the Deep Space 2 probe, a pair of high-tech lawn darts that NASA had piggybacked onto the Polar Lander to get soil data from the Red Planet. Last September the $125 million Mars Climate Orbiter burned up in the Martian atmosphere because engineers had failed to convert pounds to metric weight in their calculations.

The report, by an independent panel of scientists, attributed the gaffes to a common cause: scientists and engineers simply didn't have enough time or money to test the equipment properly, and senior managers

The report's authors raised questions about NASA's operating credo of "faster, better, cheaper." The panel was curt in its criticism. "Space missions are a one-strike-and-you're-out activity," it warned. The authors served up a new corollary to NASA's hurry-up doctrine: "If not ready—do not launch."

In a pep talk to Mars engineers at NASA's Jet Propulsion Laboratory in California, a chastened Administrator Daniel Goldin accepted blame. "I pushed too hard," he said, "and ... stretched the system too thin." Accordingly, agency officials delayed a Mars launch set for 2001 until 2003 or later, budgeted more money for contingencies and picked a point man to oversee all Mars projects. (Another orbiter mission for 2001 will go ahead as planned.) Meanwhile, Goldin and others believe that when it's done right, "faster, better, cheaper" remains the space agency's best policy. That's good, because NASA shouldn't look for more funding from Congress. "Everybody's entitled to one stupid mistake, but now there are two stupid mistakes," Rep. Dana Rohrabacher, chair of the House committee that funds NASA, told NEWSWEEK. "Money is definitely not the answer." Maybe not,

Poor testing is a typical (though not the only) feature in spectacular software failures…

Before we go further …

define “Software Testing”

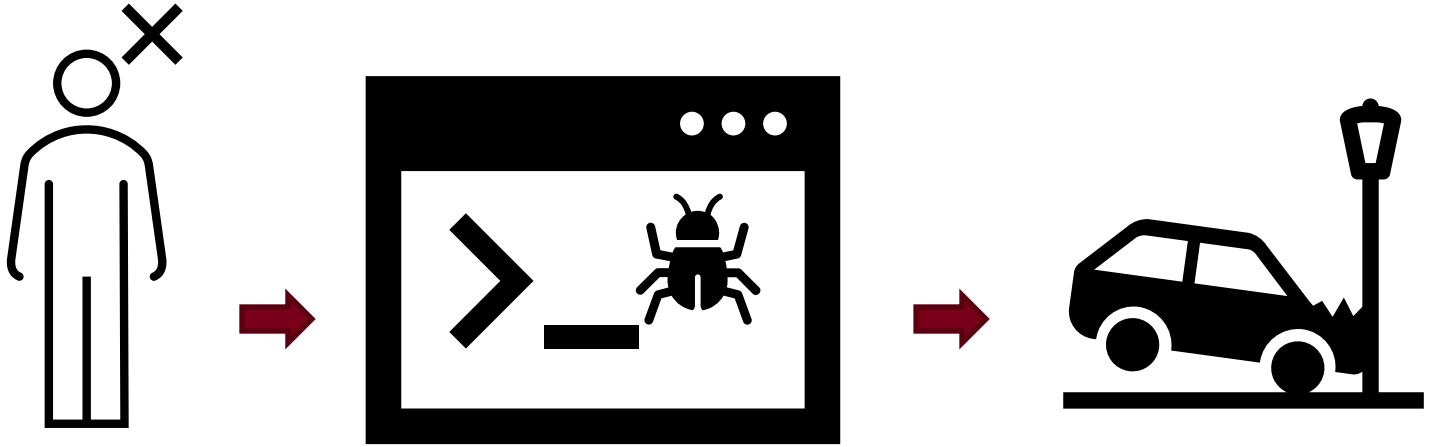
Several “definitions”…

- (**Myers**, **Humphrey**) The process of executing a program (or part of a program) with the intention of finding errors
- (**Jorgensen**) Testing – the process of finding errors and of validating the program/system
- (**Kaner**) The process of searching for errors
- (**Spillner**) The whole process of systematically executing programs to demonstrate the correct implementation of the requirements, to increase confidence, and to detect failures
- [**GBV**] The process consisting of **all life cycle activities**, both static and dynamic, concerned with planning, preparation, and evaluation of **software** products and **related work products** to determine that they **satisfy specified requirements**, to demonstrate that they are **fit for purpose** and to **detect defects**.

Software testing is ….

- not mere execution of tests
- more than verification (meeting spec)
- encompassing validation (meeting user need)
- not debugging

Software anomalies and associated terms



Mistake/Error

Defect/Bug/Fault

Failure/Problem/Glitch

IEEE Std1044™ - 2009
Classification for Software Anomalies

Problem – Failure

A problem may be caused by one or more failures.

A failure may cause one or more problems.

Failure – Fault

A failure may be caused by (and thus indicate the presence of) a fault.

A fault may cause one or more failures.

Fault – Defect

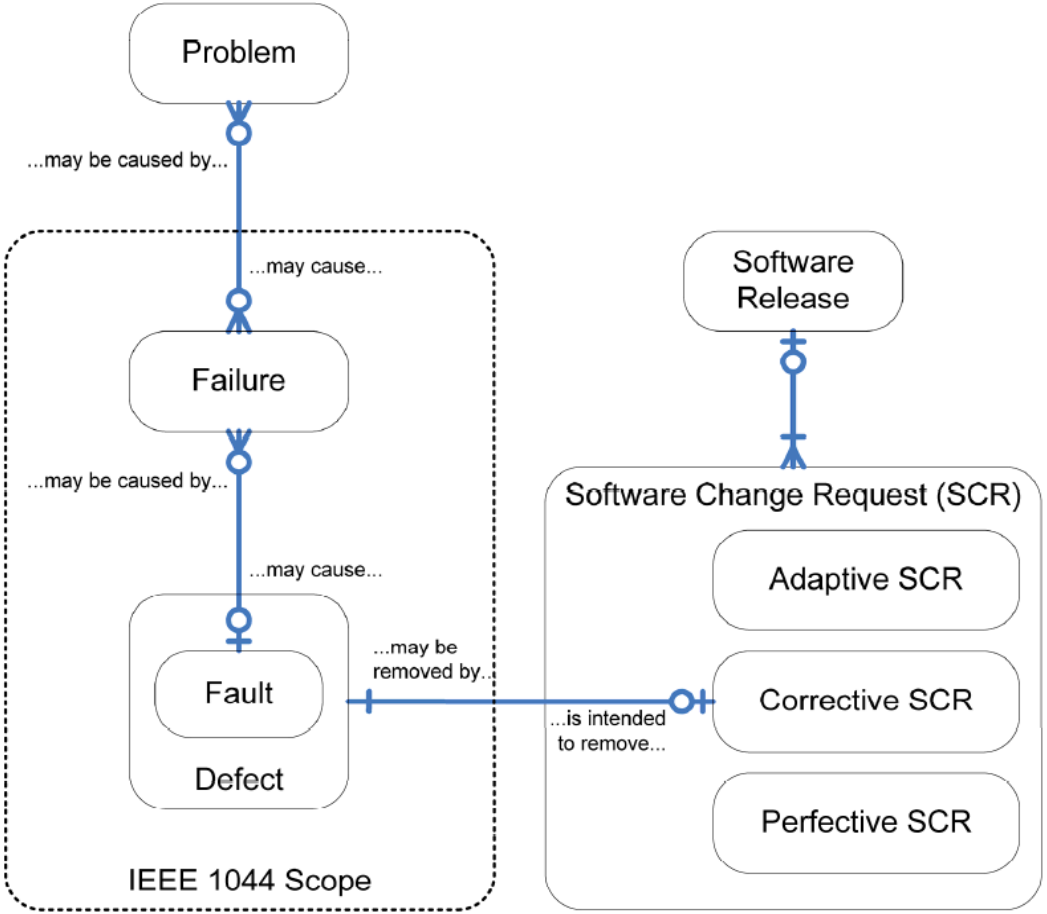
A fault is a defect that is encountered during software execution.

A defect is not a fault if it is detected by inspection or static analysis and removed prior to execution.

Defect – Change Request

A defect may be removed via completion of a corrective change request.

A corrective change request is intended to remove a defect.



For what purposes do we test software?

- Evaluate work product
- Verify requirements have been met
- Validate that user needs are satisfied
- Build confidence
- Find failures
- Prevent defects
- Assess quality of a *test object*
- Reduce risk
- Meet regulatory or contractual obligations

Relation to Quality

- What is quality?
  - The degree to which a desired characteristic exists or is met

- Quality Management
  - Assurance – over process
  - Control – over product

Quality assurance
ensures good testing process which
in turn ensures quality control of the product

(Discuss assigned reading) Why is it hard to test software?

None of the steps are simple and none of the techniques are complete

“The first and most important thing to be done is to recognize the complex nature of testing and take it seriously. “

“Hire the smartest people you can find, help them get the tools and training they need to learn their craft, and listen to them when they tell you about the quality of your software. Ignoring them might be the most expensive mistake you ever make.”

Is the situation better/worse/same 20+ years since?

(How) is “AI” helping?

Model the environment

- Interfaces, data formats, interaction sequences

Select test scenarios

- Input coverage, code coverage

Execute and evaluate

- Automation, oracles

Track and assess progress

- Adequacy metrics, reliability models

Seven Principles

1. Testing shows the presence of defects, not their absence
2. Exhaustive testing is impossible
3. Early testing saves time and money
4. Defects cluster together
5. Beware of the pesticide paradox
6. Testing is context dependent
7. Absence-of-errors is a fallacy

Some questions that we will seek to answer

- How do we know what to test?
- What methods do we use to do this testing?
- How do we know that our tests are any good?
- How do we know when to stop testing?
- What do we do if a test fails?

A little *testing* exercise

```
Usage: ./mylen <non-decreasing-sequence-of-numbers>
        Returns the position of an element in the
        sequence that is equal to the sequence length
        or zero if no such element exists.
```

Your task: Come-up with tests for this toy program

A refresher

Discrete Mathematics

UNIVERSITY OF MINNESOTA

Driven to Discover®

Crookston Duluth Morris Rochester Twin Cities