Why Do We Test Software?

CS 3250
Software Testing

[Ammann and Offutt, “Introduction to Software Testing,” Ch. 1]

Think about your experience.

Where do you see or use software?

What could go wrong if software is not tested (appropriately, adequately, or not tested at all)?

(some) **Software Failures**

- 2023: Flights from coast to coast were grounded and unable to depart due to a software malfunction (FAA outage)
- 2023: Military helicopter crash caused by failure to apply a software patch
- 2022: Tesla recalled nearly 12K vehicles due to battery controller failures
- 2022: Millions of web server vulnerability due to a defect in the Log4j software
- 2021: Log4j did not sanitize its input, allowing malicious attackers to execute code remotely on any targeted computer
- 2020: More than 100 flights to and from London’s Heathrow airport disruption due to issues with departure boards and check-in systems
- 2020: A number of Hyundai and Kia recalls due to software park system malfunction
- 2020: Microsoft Azure experienced a six-hour outage due to issues with the building automation control system that caused a cooling system failure
- 2020: Google Cloud service disruptions affecting Gmail, Google Classroom, Nest, YouTube due to storage issues with Google’s authentication system and email configuration update
- 2020: AWS suffered over 6 hours due to an operating system configuration issue

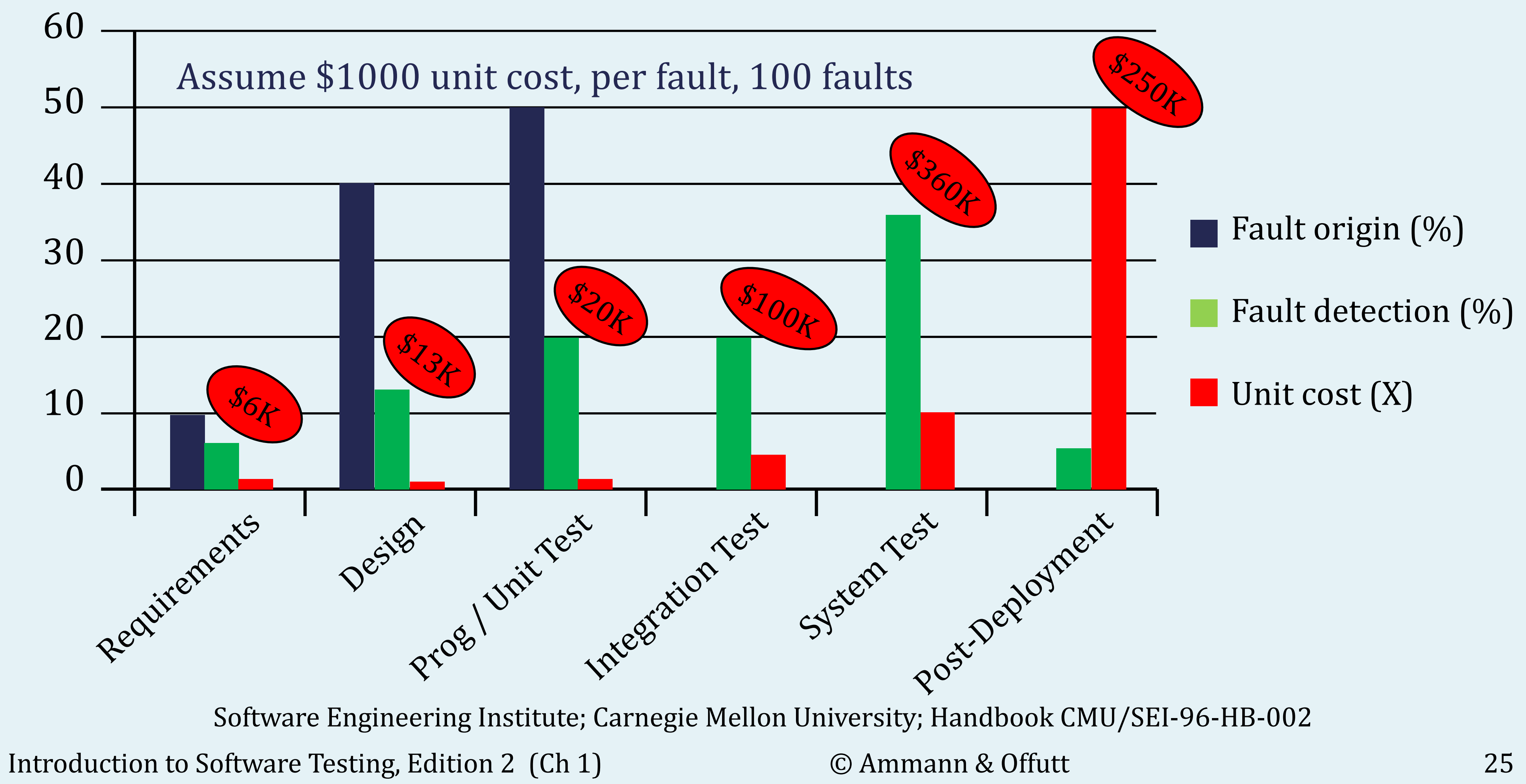
(some) **Software Failures** (cont.)

- 2019: Facebook, Instagram, WhatsApp 14 hours downtime due to Facebook News Feed issue in routine maintenance
- 2019: Boeing 737 Max crashed due to aggressive software flight overrides
- 2018: Hawaii Emergency Management Agency sent out a false missile alert due to no visible alterations between live alert and testing environments
- 2018: Pedestrian in Arizona was killed by an Uber car due to its self-driving software failure
- 2018: Google shut down Google+ due to the undetected fault that was present for more than two years, causing nearly 500,000 users' data to be compromised
- 2018: TSB system upgrade causes months of online banking disruption
- 2017: Cloudflare's major software fault led to customer sensitive data leakage
- 2017: 606 recorded software failures, impacting 3.7 billion people, 314 companies, $1.7 trillion in financial losses
- 2016: Nissan recalled 4 millions cars from the market due to software failure in the airbag sensory detectors
- 2016: Info lost due to the browser back button while using TurboTax software

(some) **Software Failures** (cont.)

- 2015: Bloomberg’s trading terminal failures forced the British government to postpone $4.4 billion debt sale
- 2014: Dropbox’s outage was due to a fault in a maintenance script
- 2012: Faults in a new Knight Capital’s trading software causes $440 millions
- 2007: Symantec concluded that most security vulnerabilities are due to faulty software
- 2003: Northeast blackout due to the alarm system in the energy management system failure, affecting 40 million people in 8 US states, 10 million people in Ontario, Canada
- 1999: NASA’s Mars lander crashed due to a unit integration fault
- 1997: Ariane 5 explosion: Exception-handling bug forced self-destruct on maiden flight (64-bit to 16-bit conversion), causing $370 millions
- 1986: 3 patients were killed by Therac-25 radiation machine due to poor testing of its safety-critical software

Cost of Late Testing

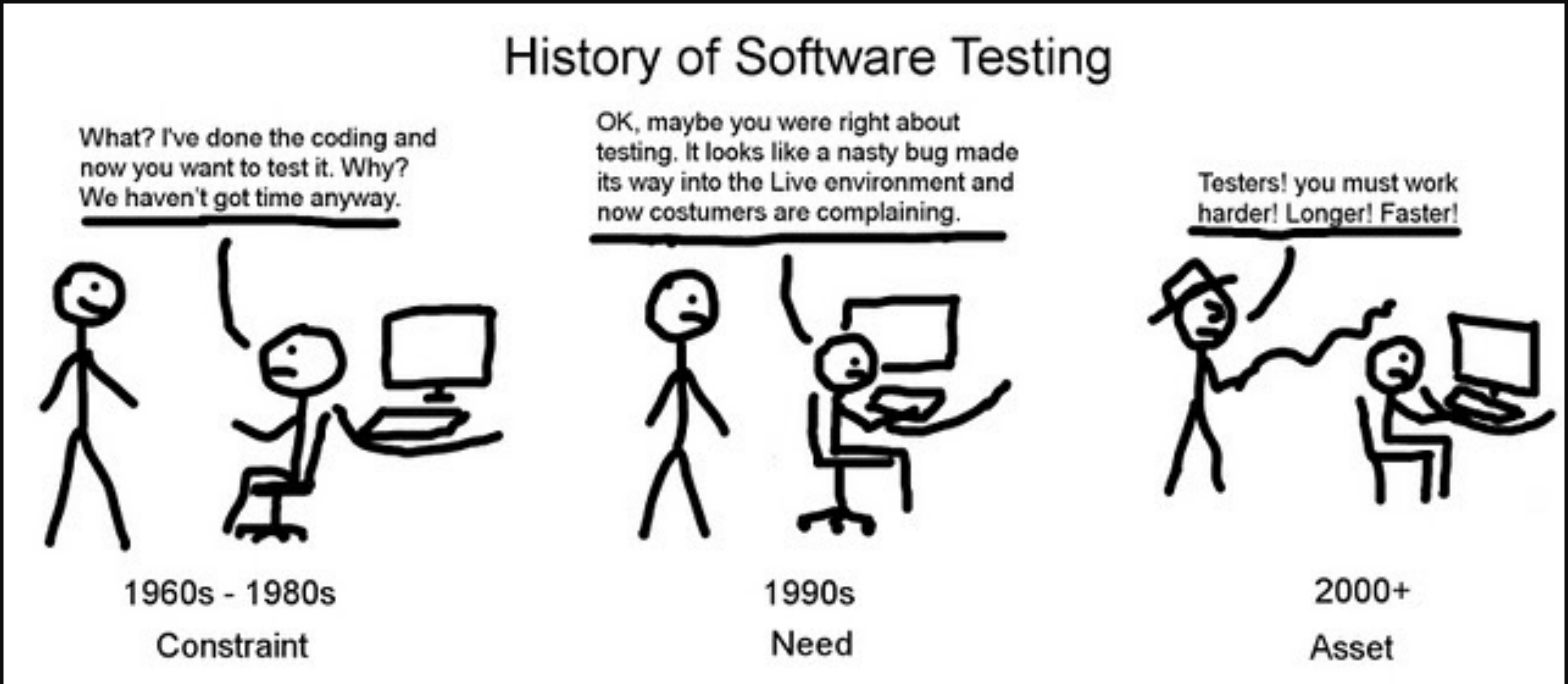


Software Engineering Institute; Carnegie Mellon University; Handbook CMU/SEI-96-HB-002

Introduction to Software Testing, Edition 2 (Ch 1) © Ammann & Offutt 25

[Chart illustrated by Ammann & Offutt
Source: Software Engineering Institute; Carnegie Mellon University; Handbook CMU/SEI-96-HB-002; page 56-58]

History of Software Testing


History of Software Testing
What? I've done the coding and now you want to test it. Why? We haven't got time anyway.
OK, maybe you were right about testing. It looks like a nasty bug made its way into the Live environment and now costumers are complaining.
Testers! you must work harder! Longer! Faster!
1960s - 1980s
Constraint
1990s
Need
2000+
Asset

[image: http://ashishqa.blogspot.com/2012/12/history-of-software-testing.html]

Why Do We Test?

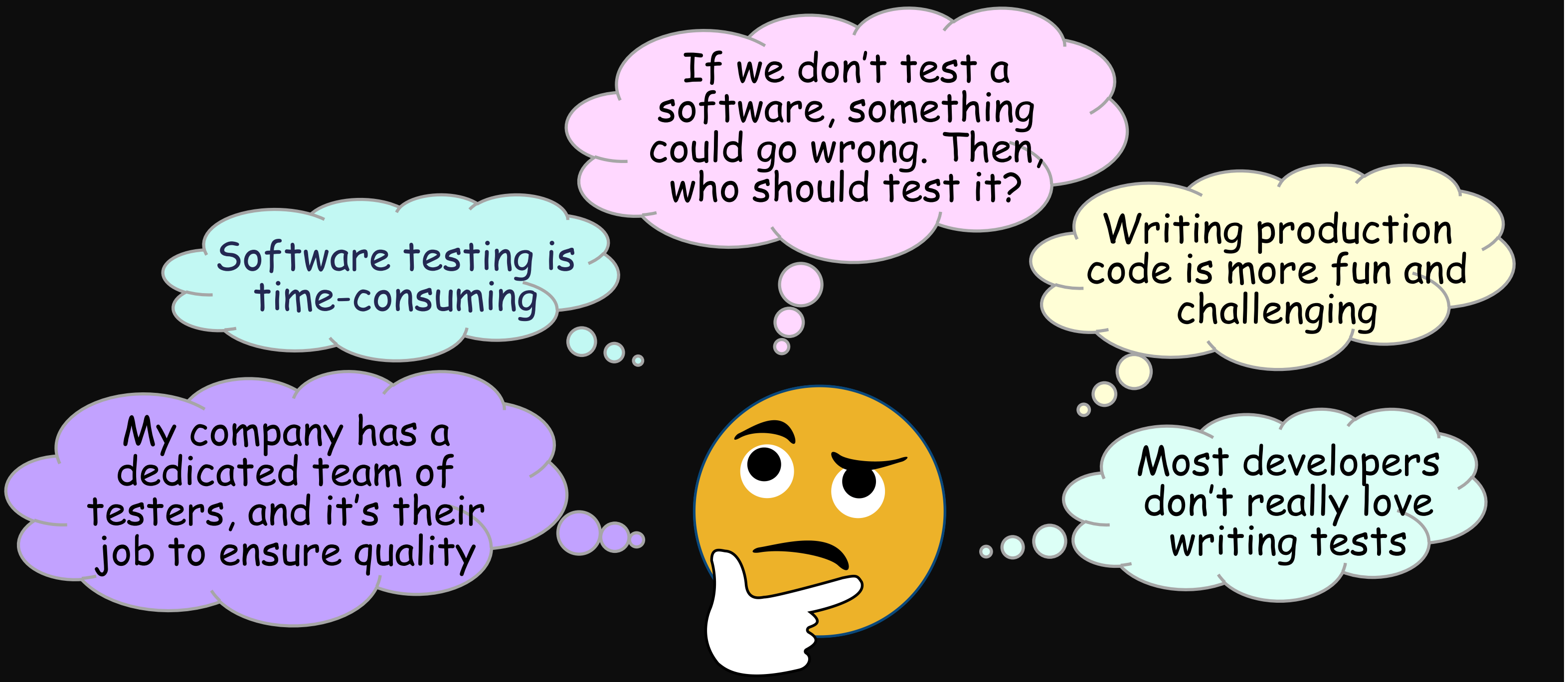
- Guard software from regression
- Improve the code quality
- Reduce uncertainty
- Increase the development pace
- Enhance the specification density
- Boost confidence and courage

Increase confidence for anyone who is affected
through some forms of evidence

Software Testing – Who Cares?

Software testing is not just for testers. As a developer:

- It is also your responsibility to ensure the quality of your product.
- Tests are the tool to help you with that responsibility.
- If you design tests properly, you can test your code in an effective and systematic way.


If we don't test a software, something could go wrong. Then, who should test it?
Software testing is time-consuming
Writing production code is more fun and challenging
My company has a dedicated team of testers, and it's their job to ensure quality
Most developers don't really love writing tests

[Ref: emoji by Ekarin Apirakthanakorn]

The Essence of Testing

Technical	Models (ISP, graph, logic, syntax), tools or test automation frameworks
investigation	An organized and thorough search for information (~run tests and look carefully at the results)

to expose quality-related information

- Find sources or problems to get them fixed
- Check intraoperability and interoperability
- Help in decision making (release/no-release)
- Minimize technical support costs
- Assess conformance and compliance
- Minimize safety-related lawsuit risk
- Determine safe scenarios for use of the product

about the project or software under test

Testing in the 21st Century

- Safety critical, real-time software
- Embedded software
- Enterprise applications
- Security
- Web
- Mobile

Software testing becomes more important

We need reliable software.
Testing is one way to assess reliability and thus improve quality of software

Wrap-up

- Testing is the most time consuming and expensive part of software development
- Not testing is even more expensive
- Having too little testing effort early increases the testing cost
- Planning for testing after development is prohibitively expensive
- A tester's goal is to eliminate faults as early as possible
- **What's next?**
  - Getting started – intro to software testing