Software Testing

Lecture 1

Prepared by

Soha Makady

s.makady@fci-cu.edu.eg

Outline

- Course Organization
- Software testing
- Testing versus Quality Assurance
- Is Software Testing trivial?
- Is Software Testing important?

Course Organization

(Tentative) Course Structure

- Part 1: Introduction
  - Introduction to Software Testing
  - Defects lifecycle and defect tracking
  - Black box testing techniques
  - White box testing techniques
  - Test adequacy assessment

Course Structure (Cont'd)

- Part 2: Various kinds of testing
  - Configuration testing
  - Compatibility testing
  - GUI testing
  - Web application testing
  - Performance testing

Course Structure (Cont'd)

- Part 3: depends on Schedule ?
  - Test Driven Development
  - Behavior Driven Development

Course Structure (Cont'd)

- Are we going to study testing/QA tools?
  - Yes, yet within the context of the course
  - The main focus is to learn the science behind testing and QA, not to study tools
  - Tools provide their own tutorials!

Course Access Code

- Enroll access code TBD
- Course ID: TBD

(Tentative) Evaluation

- Final exam (60)
- Coursework (40)
  - Midterm (20)
  - (Pop up) Quizzes
  - Assignments
  - Project

Evaluation (Cont'd)

- Cheating Policy
  - There will be ZERO tolerance for any sort of cheating.
  - COPYING your code from online resources IS CHEATING
  - Discussing the details of your solution with your colleague is CHEATING
  - In case of cheating, a faculty-based regulation will be taken.

Course Goals

- By the end of this course, you will have learnt:
  - Fundamental concepts and terminologies related to software testing.
  - Software test planning and the defect life cycle.
  - Basic software test design techniques.
  - Various types of software testing, and how they related to the studied test design techniques.

Course Material

- Textbooks:
  - Ron Patton. 2005. Software Testing (2nd edition).
  - Burnstein, Ilene. *Practical software testing: a process-oriented approach*. Springer Science & Business Media, 2006.
  - Paul Ammann & Jeff Offutt, 2017 “Introduction to Software Testing” 2nd Edition
  - Other sources: Online sources: will be posted on the course’s website.
- Other topics: Will be covered in details within the lecture notes, or posted as additional required readings.

Course Pre-requisites

- Software Engineering 1
- Software Engineering 2

“Program testing can show the presence of bugs, but never their absence” **(Edsger Dijkstra, 1930-2002)**

SOFTWARE TESTING

Software Testing vs. Quality Assurance

- The goal of a software tester is to find bugs, find them as early as possible, and make sure they get fixed.
- A software quality assurance person's main responsibility is to create and enforce standards and methods to improve the development process and to prevent bugs from ever occurring.

Is Software Testing Trivial?

Example 1:

- Divide into teams of 4 members each.
- Write some test cases (i.e. sets of data) to test a program.
- Such data should be handled properly by the program, to be considered a successful program.
- The program reads three integer values from an input dialog, representing the sides of a triangle.
- The program should state if the triangle is scalene, isosceles, or equilateral.
- You have 10 minutes.

Is Software Testing Trivial?

- The program reads three integer values from an input dialog, representing the sides of a triangle.
- The program should state if the triangle is scalene, isosceles, or equilateral.
- Each team should mention:
  - How many cases (tests) did they come up with (i.e., write down your total number of cases).
  - A case (with numerical data) that they covered

… till we consolidate the answers.

Is Software Testing Trivial?

- How do you evaluate this kind of program?
  - Trivial
  - Moderate
  - Complex
- How about we consider another example?

Is Software Testing Trivial?

Example 2:

- Consider a software system with 30 variables. Each variable has two possible values.
- We need to test all combinations.
- How much time would it take to test all combinations, if takes one second to execute 5 tests (in an automated manner)?
  - 1 minute
  - 1 hour
  - 1 day
  - 1 year
  - Something else?

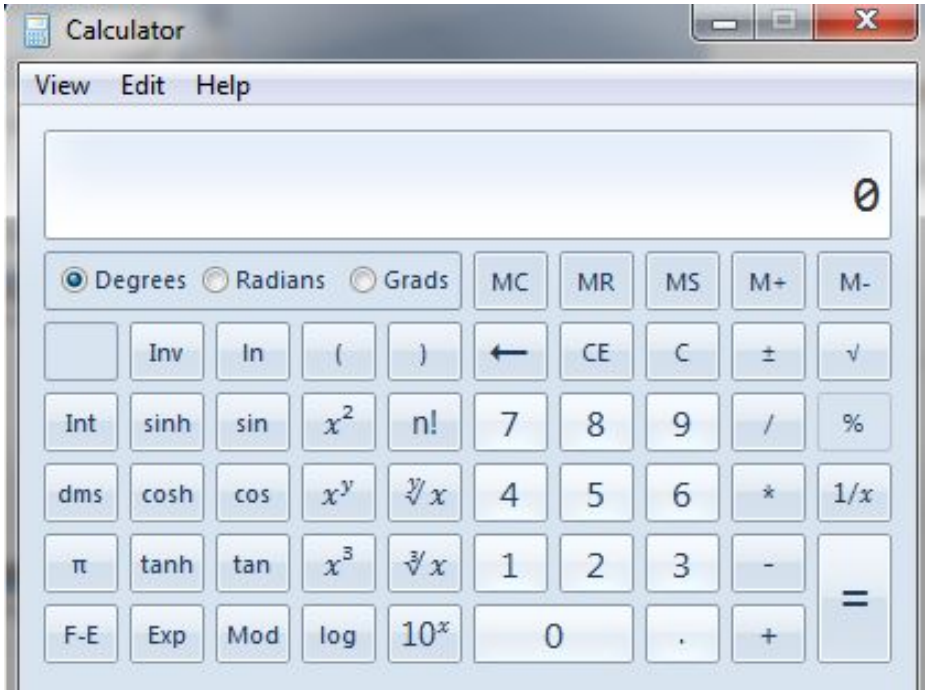
Is Software Testing Trivial?

- Consider a software system with 30 variables. Each variable has two possible values.
- We need to test all possible combinations.
- How much time would it take to test all combinations, if takes one second to execute 5 tests (in an automated manner)?
  - 6.8 years of testing!
- How do you evaluate this kind of program?
  - Trivial, moderate, or complex
- How about testing 100,000-statement air traffic control system?

Is Software Testing Trivial?

- Consider a calculator program
- The number of inputs is very large.
- The number of outputs is very large.
- The number of paths through the software is very large.
- When shall we stop ?
- Test-to-pass or

  Test-to-fail?



Is Software Testing Important?

Disney's Lion King, 1994-1995

- Disney released its first multimedia CD-ROM game for children "The Lion King Animated Storybook".
- Sales were huge before Christmas Day.
- But ….on December 26th, phone support technicians received endless calls from angry parents.
- "The software didn't work. Disney hadn't bothered to properly test their CD-ROM with most video drivers, nor had it shipped needed drivers"
- Disney failed to test the software on the different PC models.

Is Software Testing Important?

Disney's Lion King, 1994-1995

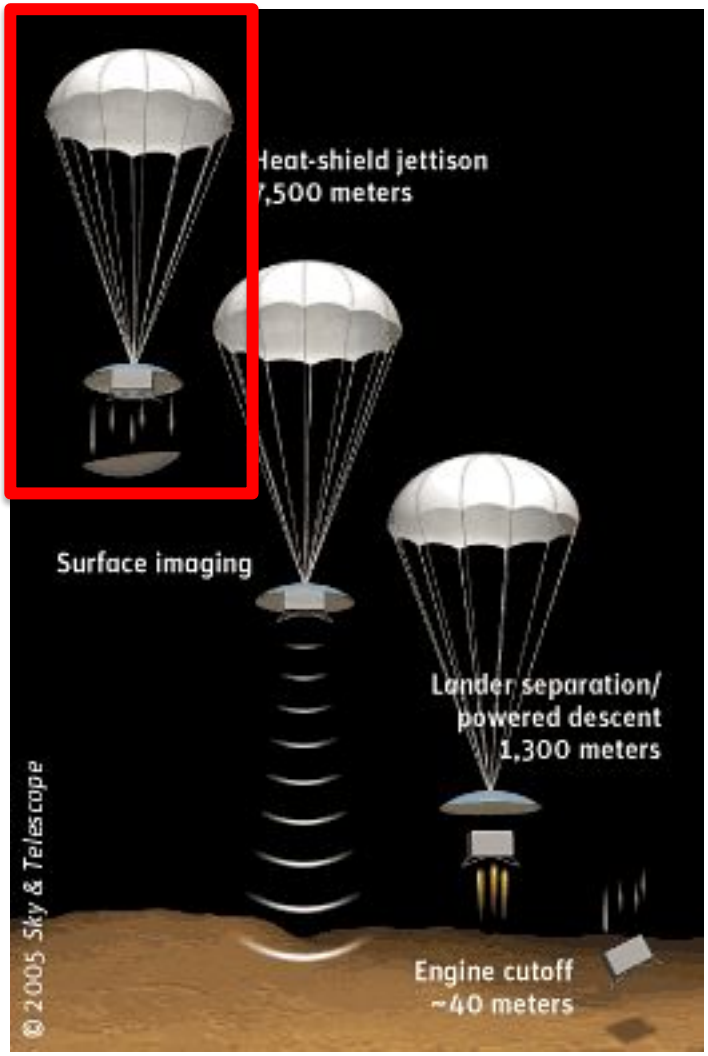
- Disney has hired two employees for their support desk 😐
- Accordingly, Disney had to:
  - Hire a telephone customer support company
  - Run through another debug, fix and test cycle.
  - Provide thousands of replacements CDs

Is Software Testing Important?

NASA Mars Polar Lander, 1999.

The landing plan was:

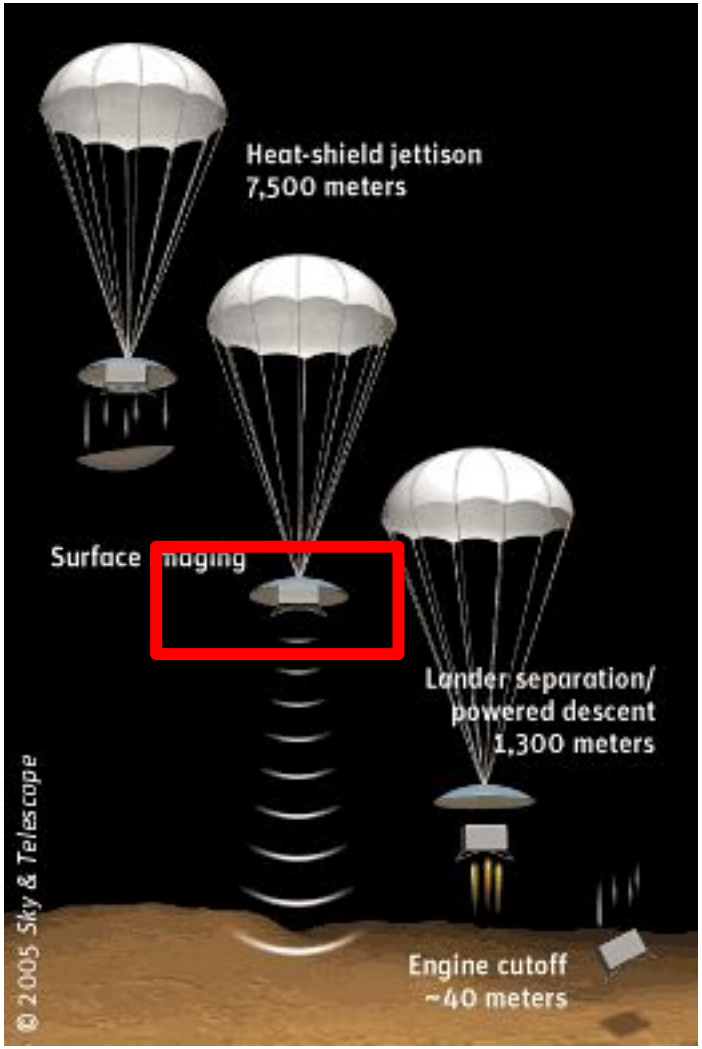
- As the lander fell to the surface it would open a parachute to slow its descent.



Is Software Testing Important?

NASA Mars Polar Lander, 1999.
The landing plan was:

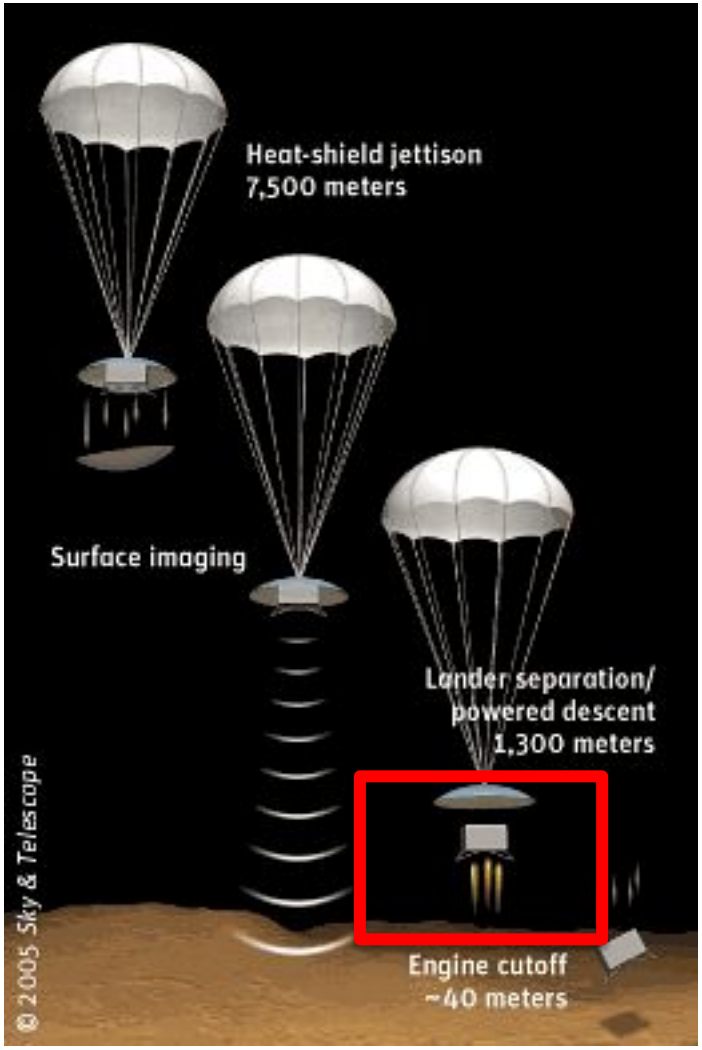
- As the lander fell to the surface it would open a parachute to slow its descent.
- When the lander approaches to a certain height, it snaps open its three legs (in a landing position



Is Software Testing Important?

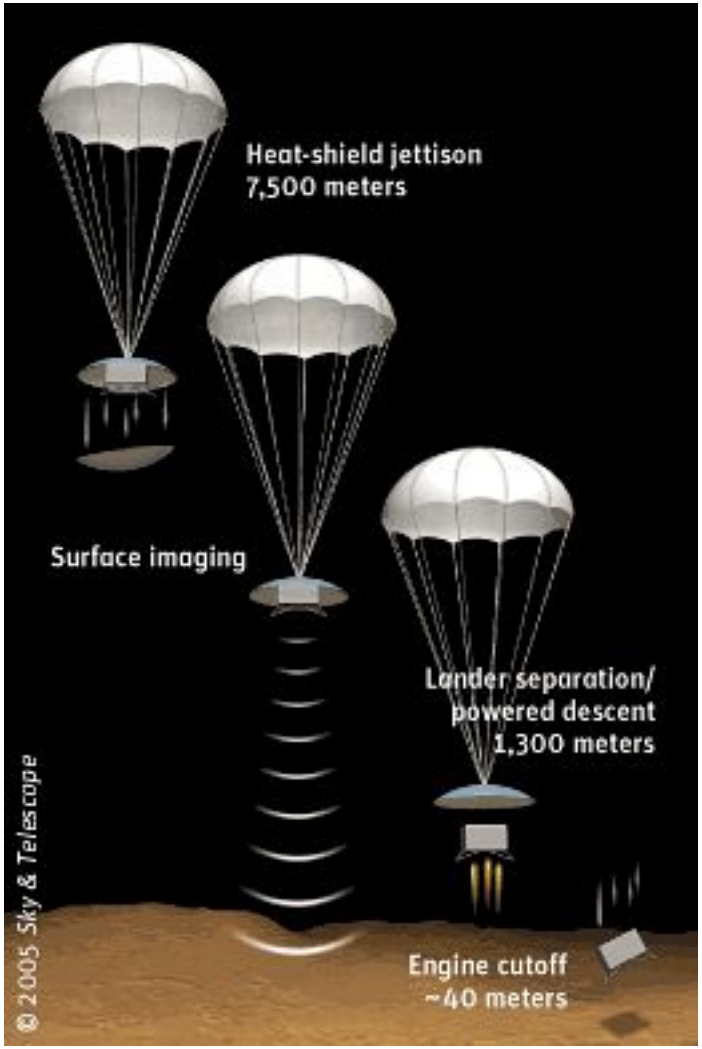
NASA Mars Polar Lander, 1999.
The landing plan was:

- As the lander fell to the surface it would open a parachute to slow its descent.
- When the lander approaches to a certain height, it snaps open its three legs (in a landing position.
- At a lower height, it would release the parachute, and ignite its landing thrusters



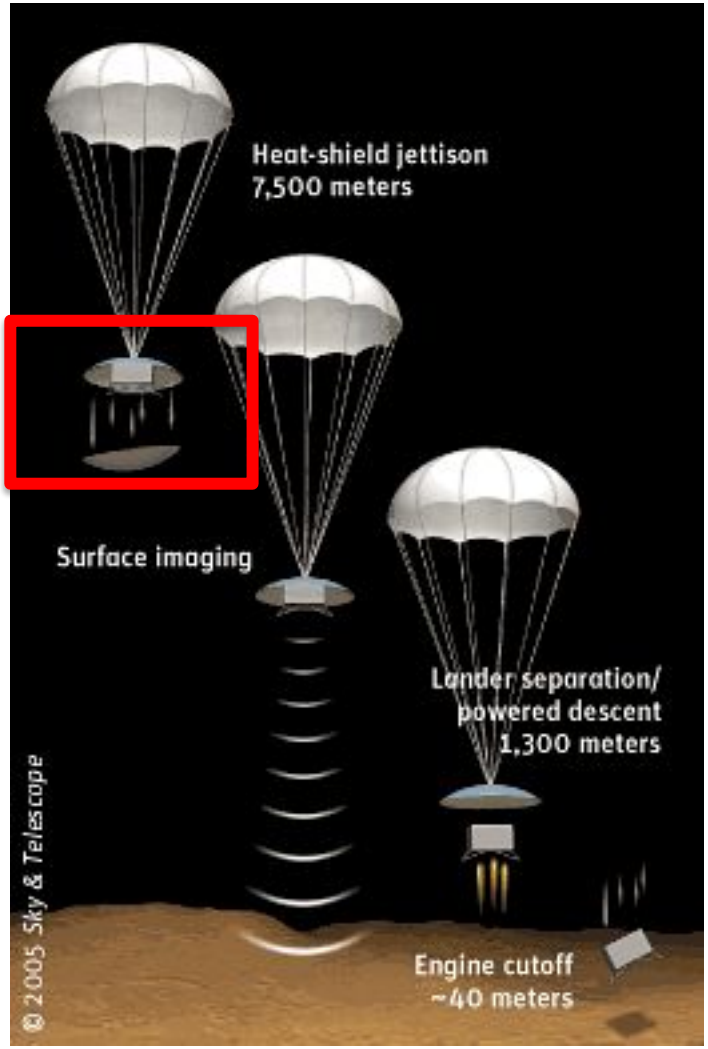
Is Software Testing Important?

- NASA Mars Polar Lander, 1999.
- However, it disappeared!
- NASA had put an inexpensive contact switch on one of the 3 legs to shut off the fuel of the thrusters.
- Hence, the engines would burn until the legs “touched down”



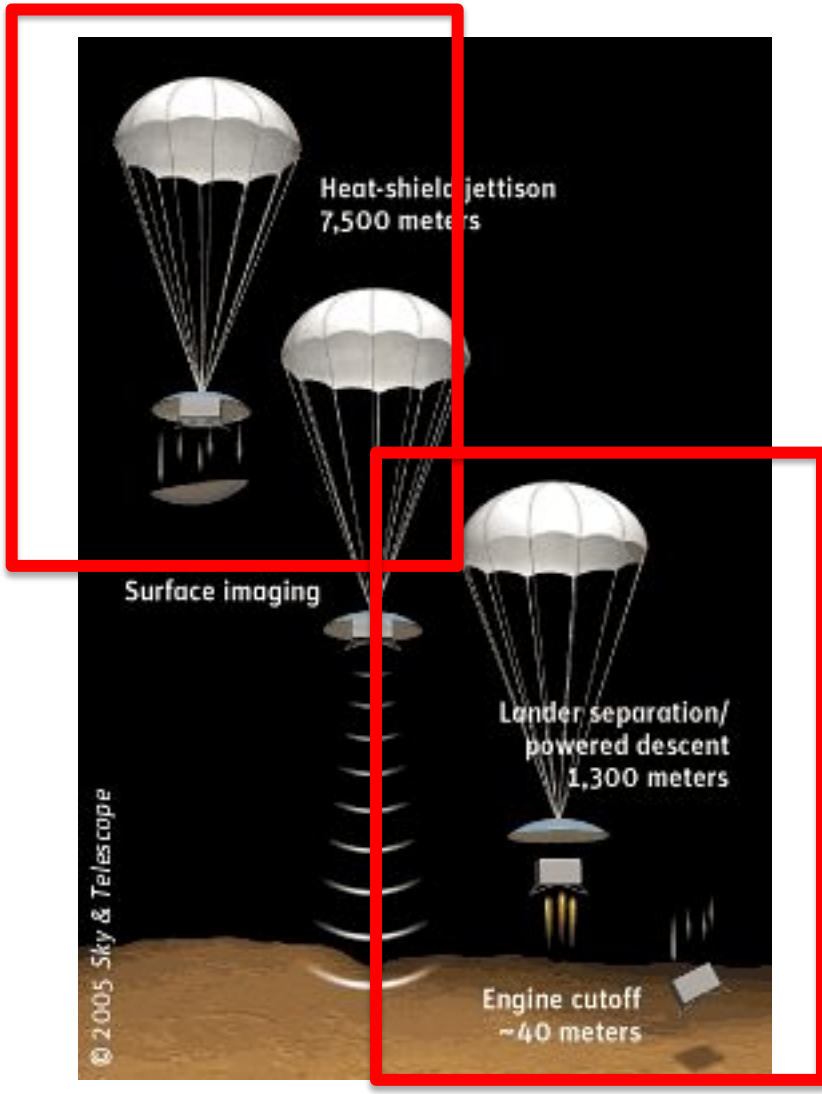
Is Software Testing Important?

- NASA Mars Polar Lander, 1999.
- However, it disappeared.
- Opening its legs had accidentally set a bit that shut off its fuel at an early stage.

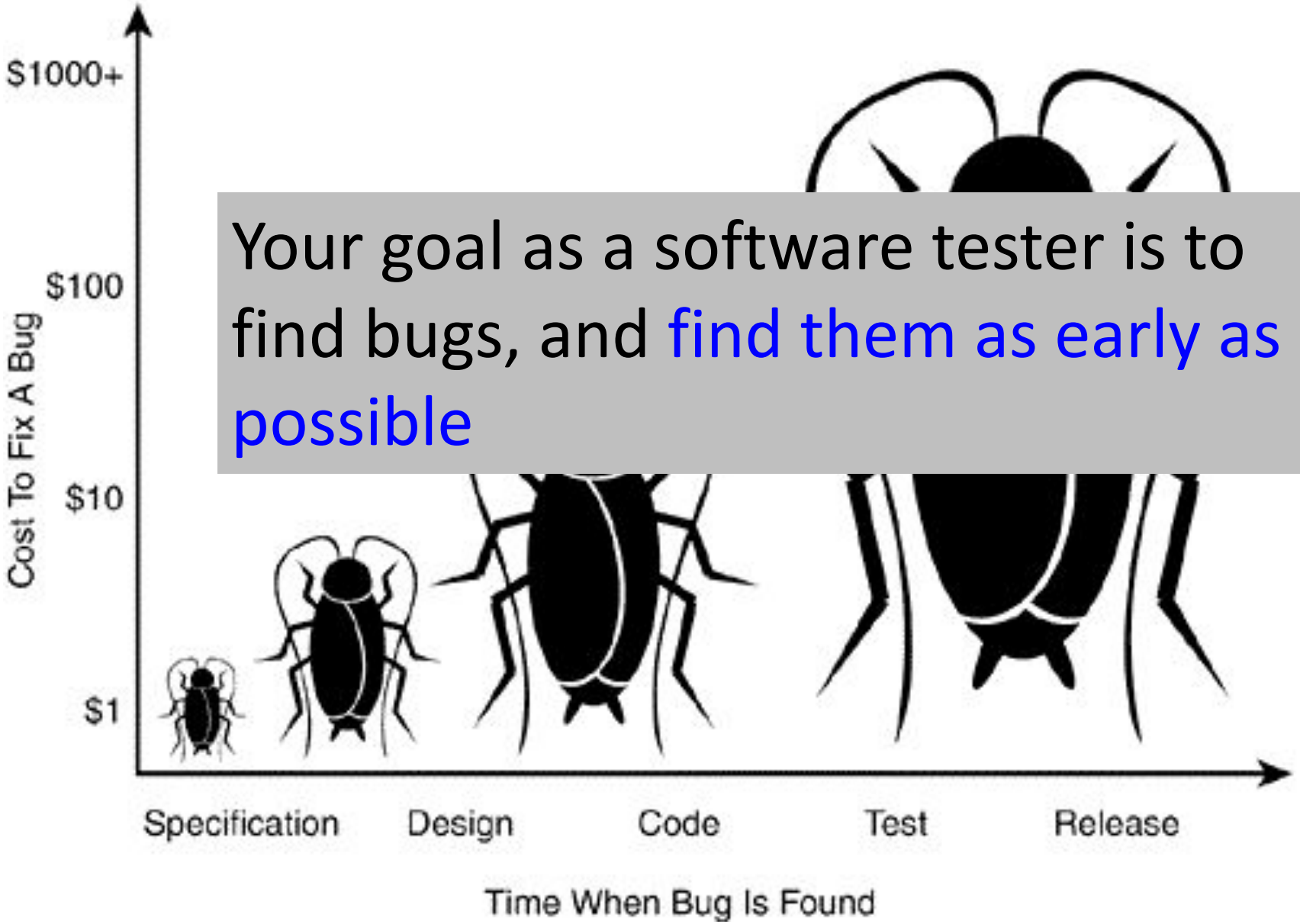


Is Software Testing Important?

- NASA Mars Polar Lander, 1999.
- The testing was done for each stage separately, hence the defect was never noticed.



The Cost of Bugs



Your goal as a software tester is to find bugs, and find them as early as possible

© Figure 1.2 – Ron Patton, Software Testing, 2nd Edition, 2005

Covered Material

- Chapter 1 from: Ron Patton. 2005. Software Testing (2nd edition).