# Rational Functional Tester

Dan Yahav
Yael Grumbach
Eitan Kovacs

# Computer system failures
## caused by bugs

- Ariane 5
  - On June 4 1996 the first flight of the European Space Agency's new Ariane 5 rocket failed shortly after launching.
  - It was reportedly due to the lack of exception handling of a floating-point error in a conversion from a 64-bit integer to a 16-bit signed integer.

# Computer system failures (cont.)

- Banking bugs
  - Software bugs caused the bank accounts of 823customers of a major U.S. bank to be credited with $924,844,208.32 each in May of 1996.

# Some facts…

- About US$250 billion spent per year in the US on application development. Of this, about US$140 billion wasted due to the projects getting abandoned or reworked.
- 20% of costs are development costs. 80% are testing costs.

# Organization of this Lecture

- Introduction to Software Engineering.
- Software testing.
- Products.
- Demo.

# Introduction to Software Engineering

# What is Software Engineering?

- *“The whole trouble comes from the fact that there is so much tinkering with software. It is not made in a clean fabrication process, which it should be. What we need, is software engineering”* (F.L. Bauer, 1968)

# What is Software Engineering? (cont.)

- Hybrid of:
  - Scientific.
  - Technical principles.
  - Management principles.

# Software process

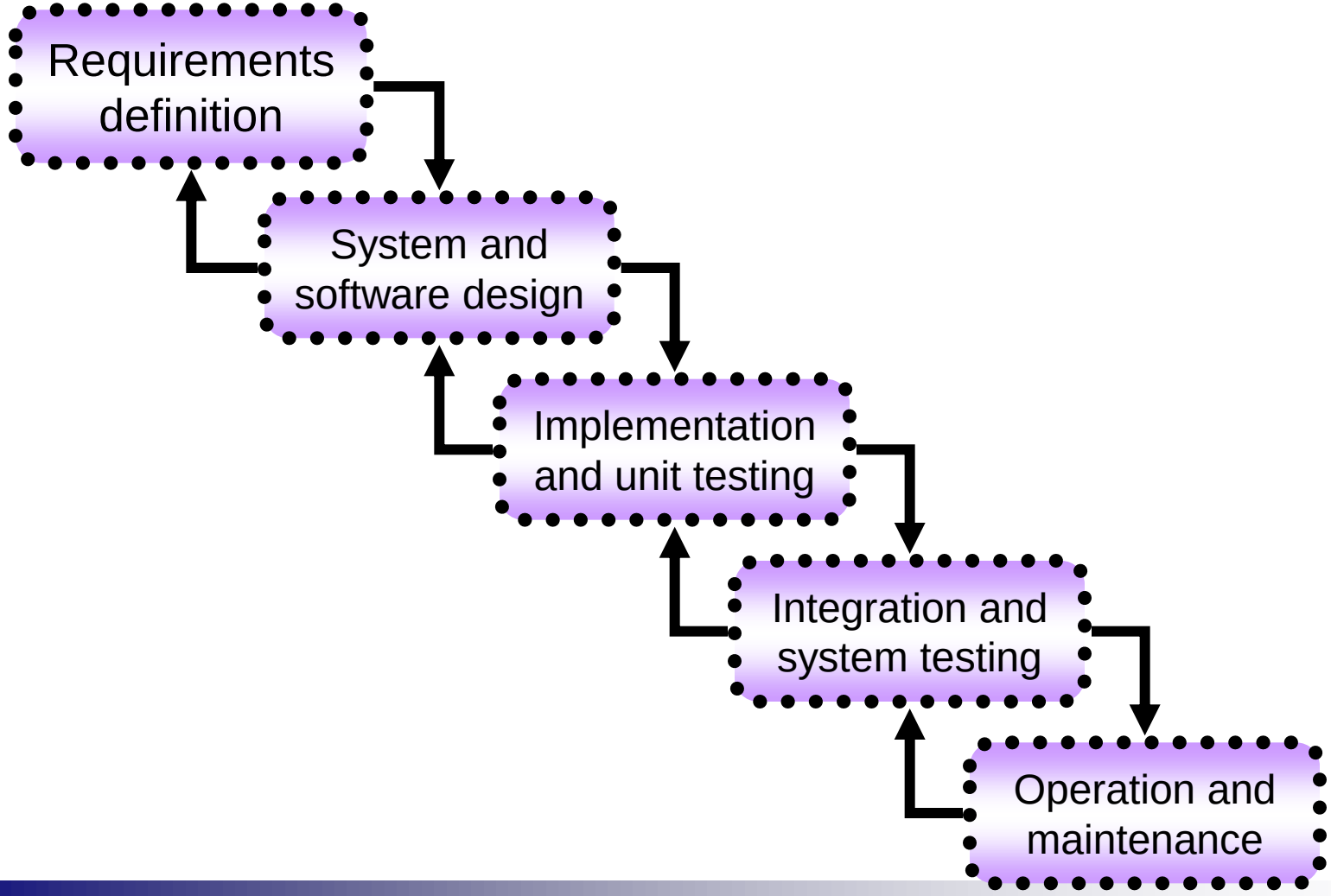
- A **set of activities** whose goal is the development or evolution of software.
- Generic activities in all software processes are:
  - **Specification** - what the system should do and its development constraints.
  - **Development** - production of the software system.
  - **Validation** - checking that the software is what the customer wants.
  - **Evolution** - changing the software in response to changing demands.

# Software process model

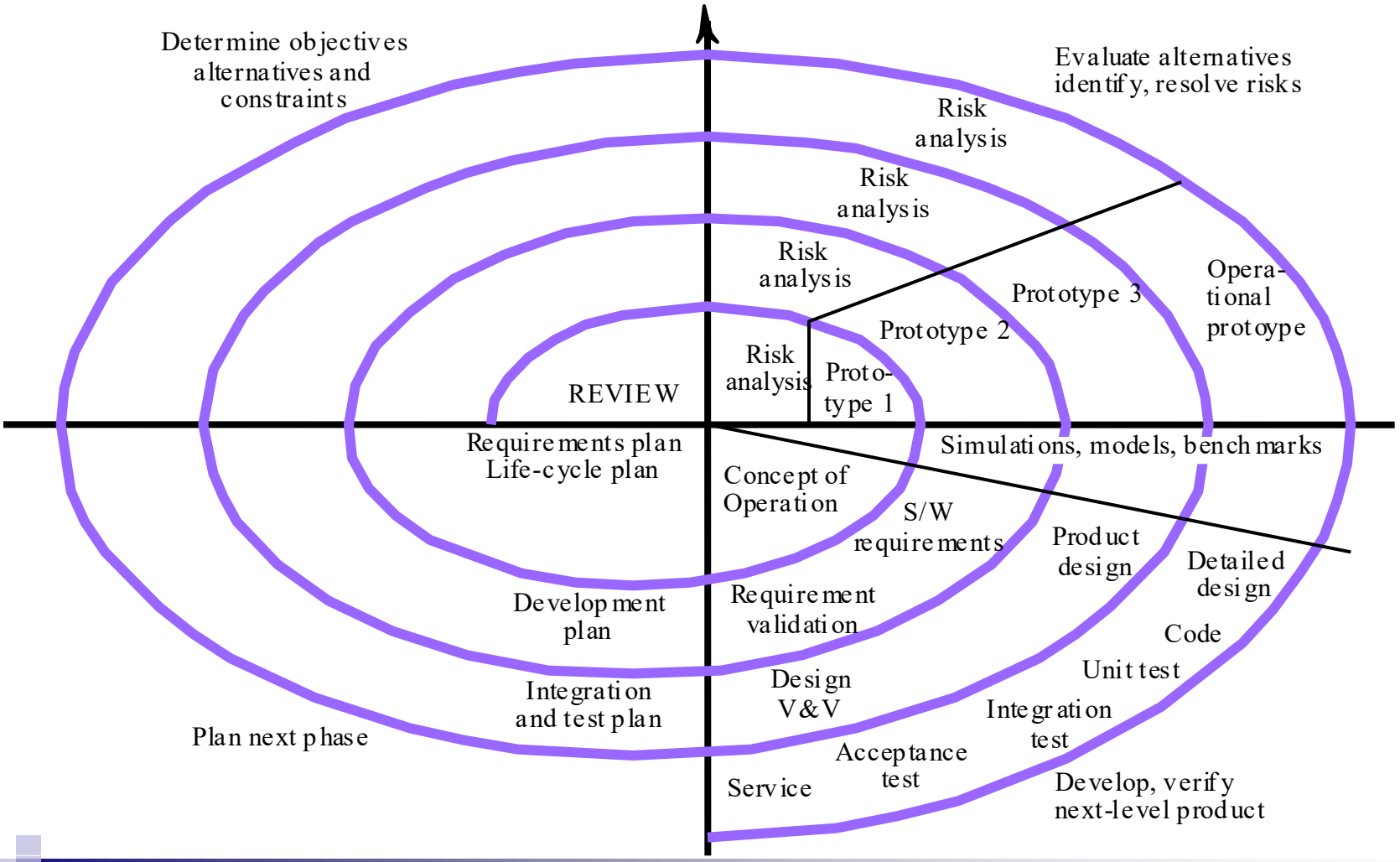
- A simplified representation of a software process, presented from a specific perspective.
- Generic process models
  - Waterfall
  - Spiral model

# Software Processes

## waterfall model



# Spiral model



# Software Testing

# Software Testing

- Definition - operation of a system or application under controlled conditions and evaluating the results.
- The controlled conditions should include:
  - Normal conditions.
  - Abnormal conditions.

# Software Testing

## Organization viewpoint

- Combined responsibility of one group or individual.

Or

- Project teams.

***It depends on what best fits an organization's size and business structure…***

# Why does software have bugs?

- **Programming errors**
  - Programmers, like anyone else, can make mistakes.
- **Changing requirements**
  - Redesign, rescheduling of engineers.
- **Miscommunication or no communication**

# Why does software have bugs? (cont.)

- **Software complexity**
  - 5 faults/1000 LOC
  - 1M LOC will have 5000 faults
  - Windows XP has 45M LOC ⇨ 45 * 5000 = 225,000
  - UNIX has 4M LOC ⇨ 4 * 5000 = 20,000
- **Time pressures**
- **Egos**
  - People prefer to say things like:

# people prefer to say things like:

**Instead Of**

No Problem.

…us a lot of
…ity and we could
… making a lot of
…nistakes.

# people prefer to say things like:

piece of cake …

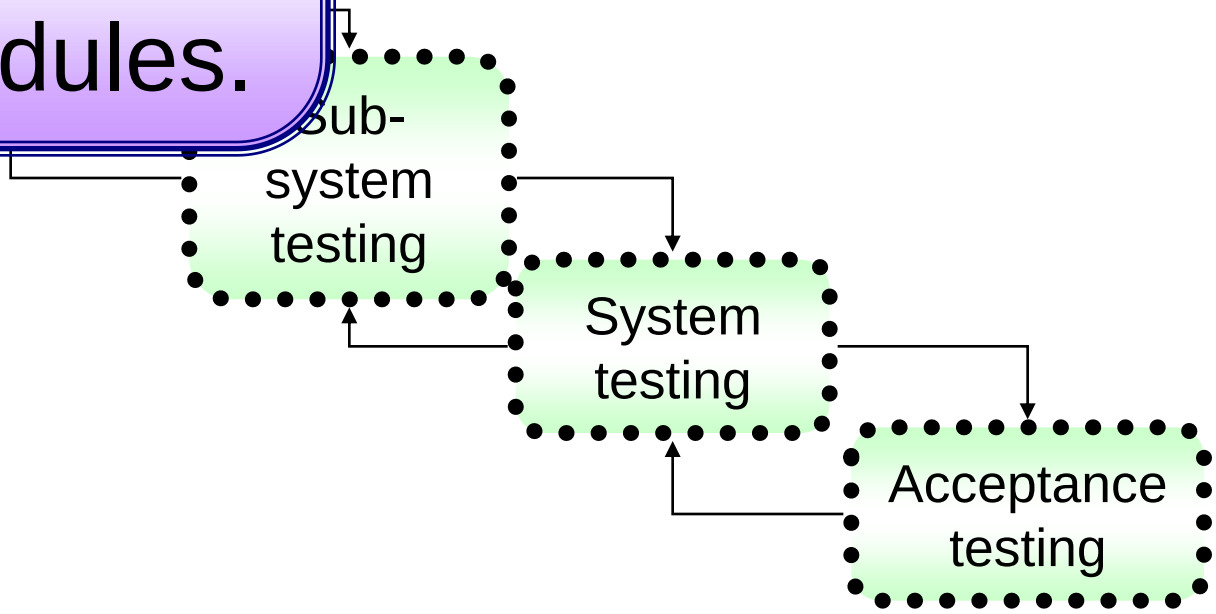
**Instead Of**

…t figure out that
…aghetti code.

# The testing process

**Unit Testing**

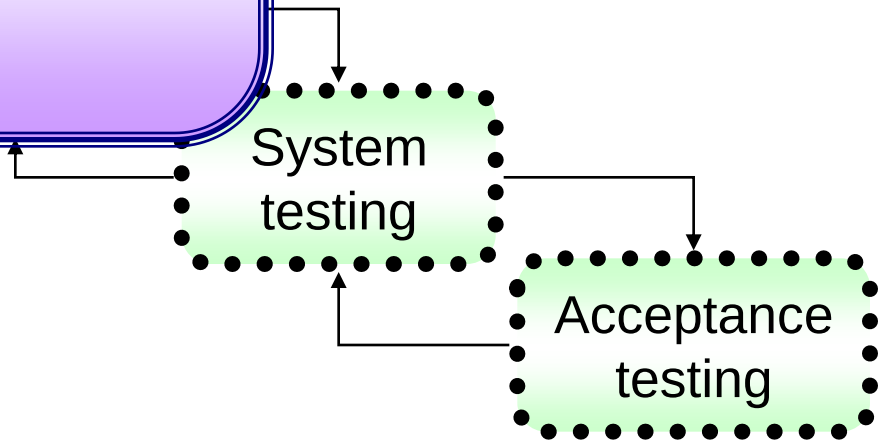
most 'micro' scale of
to test particular
ns or code modules.



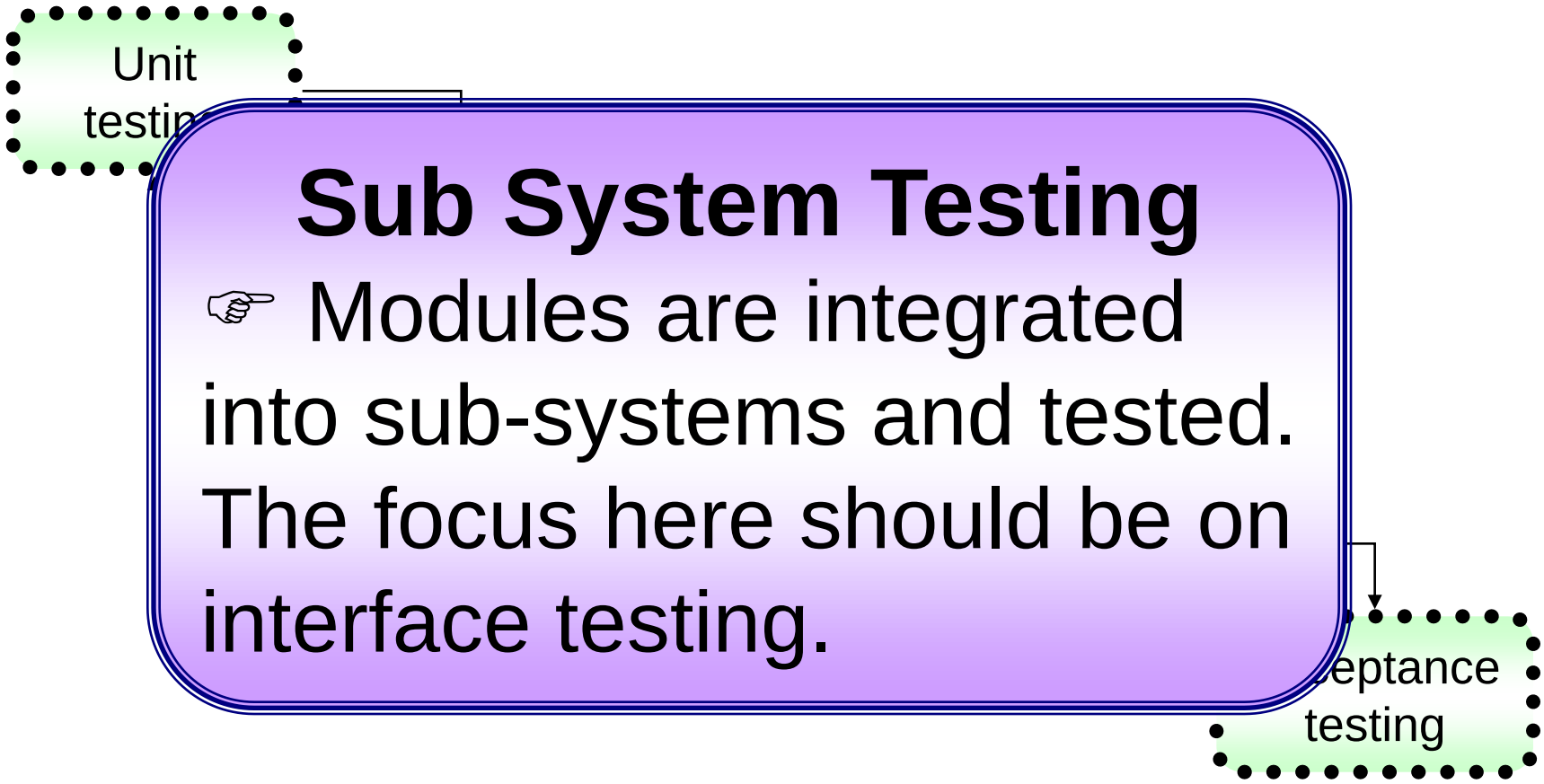
# The testing process

## Module Testing

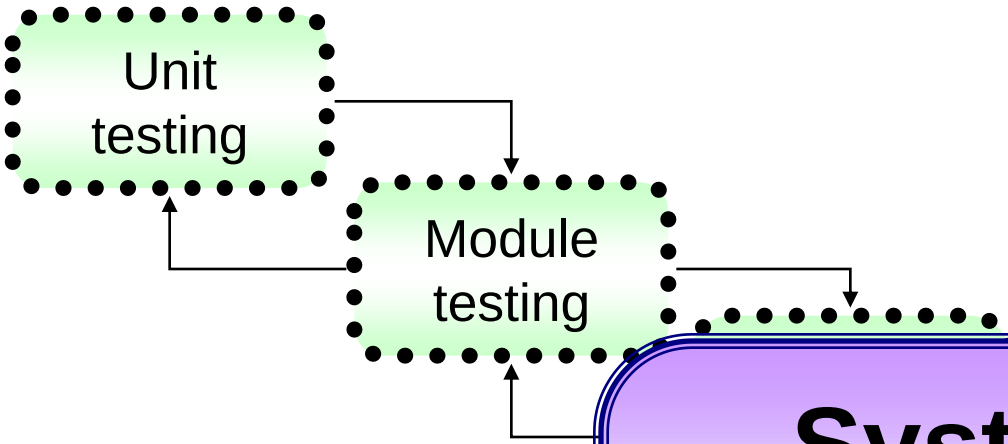
☞ Related collections of dependent components are tested.



# The testing process



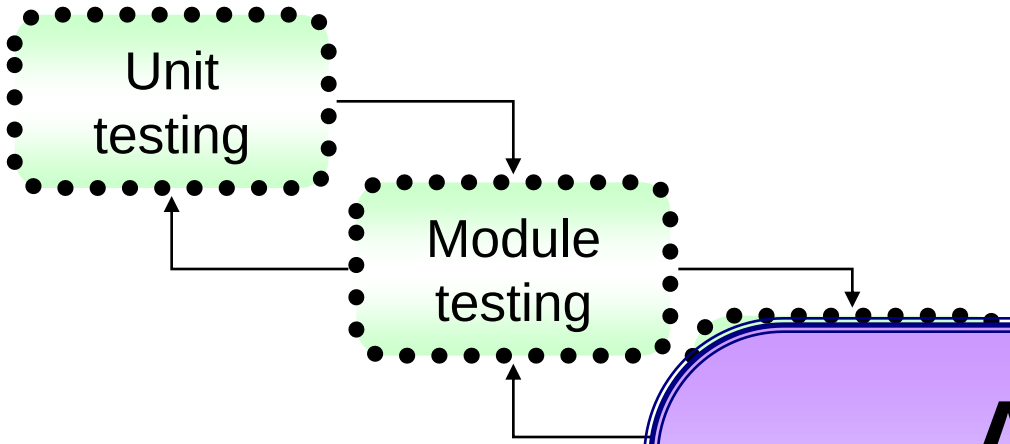
# The testing process




**System Testing**

☞ Testing of the system as a whole. Testing of emergent properties.

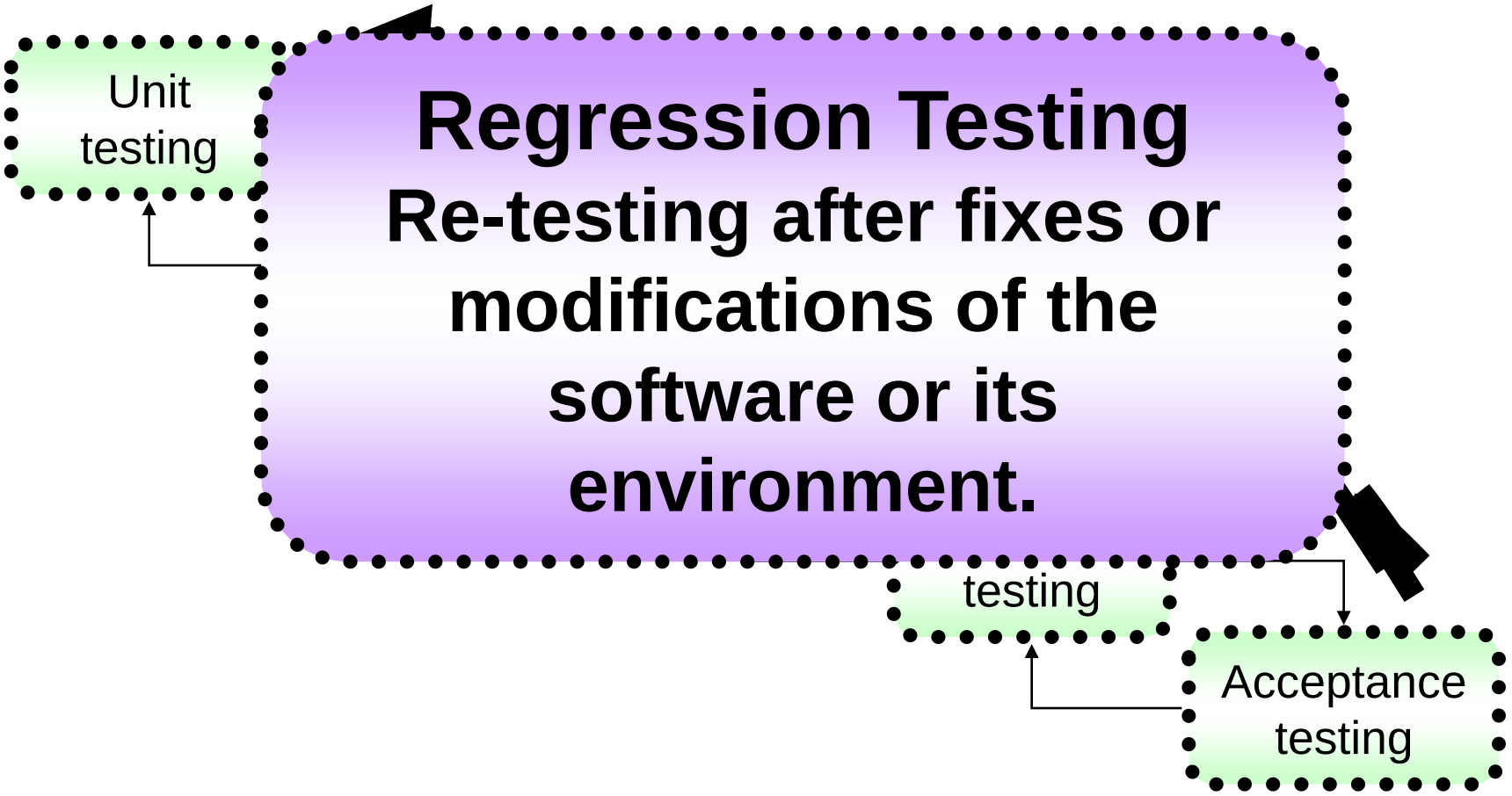
# The testing process




**Acceptance Testi**

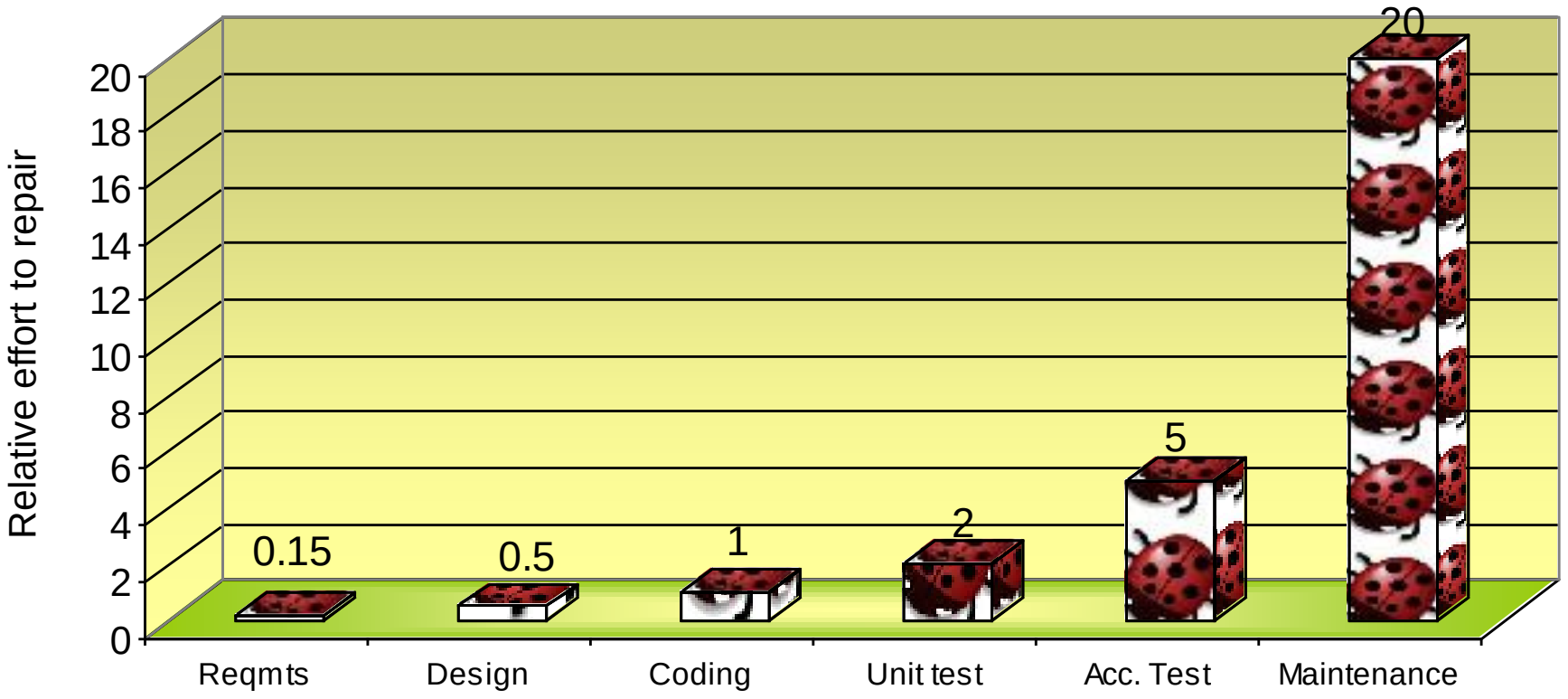
☞ Formal testing conducte
enable a user, customer or
authorized entity to determ
whether to accept a system
component

# The testing process



# Effort to Repair Software

## (when defects are detected at different stages)



# Testing types

- **White box testing.**
- **Black-box Testing.**
  - Also called Functional Testing.
  - An abstraction of a device or system in which only its externally visible behavior is considered and not its implementation or "inner workings".

# Black-box advantages

1. Unbiased test.
2. Specific programming languages knowledge is not required.
3. Will help to expose any ambiguities or inconsistencies in the specifications.
4. Early design of tests.

# Black-box disadvantages

1. Tests redundancy.
2. Difficult to design without clear specifications.
3. Cannot be directed toward specific segments of code which may be very complex.

# Products for Automated Testing

# Mercury QuickTest

**Supported Environments** - Web applications, Win32 / MFC applications.

- .NET and JAVA Add-in.
- SAP & Siebel.
- Oracle.

- **Operating System** - Windows 2000 and further.

# Rational Functional Tester

# Rational Functional Tester Features

1. Support for testing of Java, Web, Visual Studio .NET WinForm-based applications and Siebel.
2. Choice of language - Java or Visual Basic .NET - for test script customization.
3. Native Java and Visual Basic .NET editor and debugger for advanced testers.

# Rational Functional Tester Features (cont.)

4. ScriptAssure technology to accommodate frequent UI modifications.
5. Automated data-driven testing eliminate need for manual coding.
6. Multiple verification points with regular expression pattern matching support.
7. Advanced object map maintenance capabilities.

# Rational Functional Tester Features (cont.)

8. Ships with IBM Rational ClearCase LT for automated version control.

# System requirements

- Linux
  - Red hat version 9.0 (All functions except recording)
  - Red Hat Enterprise Linux WS version 3
  - SUSE Linux Enterprise Server 9
- Windows
  - Windows 2000 and further.
- Hardware
  - 500MHz Intel® Pentium® III
  - Minimum: 256MB
  - 500MB installation directory per product

# Products Comparison

| product | Range of supported application | Recommended for technical users | Recommended for non-technical users | Life cycle tool integration |
|---|---|---|---|---|
| IBM Rational Functional Tester |  |  |  |  |
| Mercury QuickTest |  |  |  |  |

# Automatic testing Rational XDE Tester

Part2

# White Box testing

- Also: glass box, *structural*, clear box and open box testing.
- Tests
  - the source code
  - the implementation logic.
- Requires knowledge
  - to select the test data
  - to examine outputs.

# White Box testing

- **Advantages:**
  - Wise input can help in testing the application effectively.
  - Helps in optimizing the code.
  - Helps in removing the extra lines of code.
- **Disadvantages:**
  - Requires a skilled tester.
  - Cannot look into every bit of code to find out hidden errors.

# Unit testing

- Type of testing where a developer proves that a code module meets its requirements.
  - Most ‘micro’ scale testing.
  - Contrast with “system test”.
- Typically done by the programmer.
- Usually associated with structural test design.

# Benefits

- Facilitates change
- Simplifies integration
- Documentation

# Limitations

- Will not catch every error in the program
- Can only show the presence of errors
- Responsibility of the developer

# Techniques & Applications

- Often conducted in an automated environment.
- The unit is executed outside of its natural environment.

- Building block to Test Driven Development (TDD).
- xUnit.

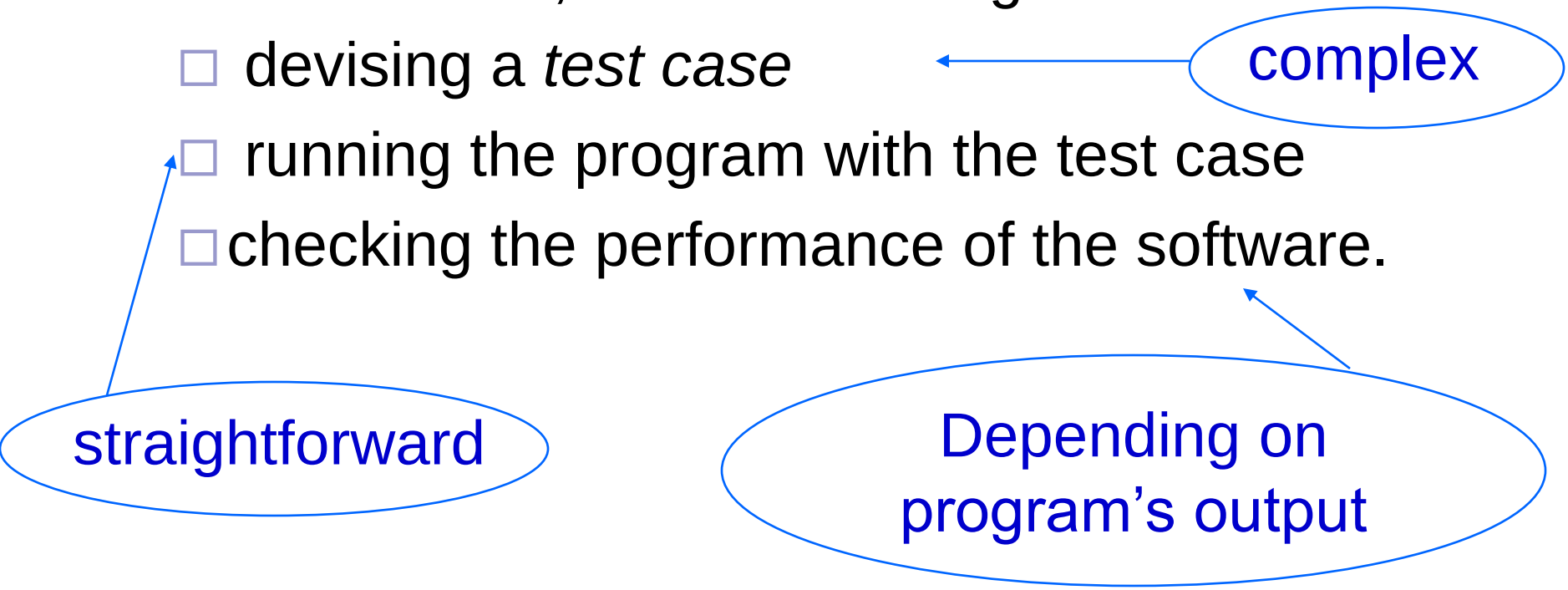
# Automatic testing

- Testing which is performed, to a greater or lesser extent, by a computer.

- Motivation:
  - Increasing demands from testers.
  - Regression tests.

# Automatic testing

- In the abstract, software testing involves:
  - devising a *test case*
  - running the program with the test case
  - checking the performance of the software.




- Partial test automation.

# The principle

- A program runs the application with proper input and checks its output against the expected.
- Once the *test suite* is written, no human intervention is needed.
- Test suites help:
  - before a new version is released.
  - software internally different for environments, but with the same external behavior.

# What's a 'test plan'?

- A document that describes *all* of a software testing effort.
  - Useful way to think through the efforts needed to validate a product.
  - Help people outside understand the 'why' & 'how'.
- Thorough, but not too much!

# Test plan template, IEEE 829 format

- Test Plan Identifier
- References
- Test Items
- Approach
- Item Pass/Fail Criteria
- Responsibilities
- Schedule
- Approvals …

# What's a 'test case'?

- A set of conditions under which a tester will determine if a requirement upon an application is partially or fully satisfied.
  - Known input
  - Expected output
- At least one per requirement.
- Help finding problems in the application design.
- Usually collected into Test Suites.

| No | Action | Expected result |
|---|---|---|
| 1 | Open application | The GUI is open. There are 10 buttons with number from 0 to 9. There are 5 operation buttons (+, -, *, /, =) and a clear button. There is also a text field with a zero number (0). |
| 2 | Press clear | The text field contains zero. |
| 3 | Press number button 0 | The text field shows the value 0. |
| 4 | Repeat actions 2,3 for all numbers between 0 to 9 | The same as above. |
| 5 | Press Clear | The text field shows 0 |
| 6 | Press button number 4 | The value 4 appears. |
| 7 | Press the operator button + | None |
| 8 | Press button number 8 | The value 8 appears. |
| 9 | Press button = | The value 12 should appear |
| 10 | … all operations | |

# GUI automation tools

- **Record/playback**: The user records a set of actions on the GUI under test, and the tool is able to replay those actions later.
- **Programmatic**: The user writes code describing the interaction with the GUI under test.
  - Also: ‘data-driven’ or ‘keyword-driven’.

# Record/Playback

- **Advantages:**
  - Simplistic
- **Disadvantages:**
  - Fragile
  - How do you properly determine delay factors between events being synthesized?
  - Often have to re-record tests

# Programmatic

- **Advantages:**
  - Can adjust to changes in GUI
  - Can determine when to send the next event more correctly
  - The tester has tons of flexibility available
- **Disadvantages:**
  - The test developer has to be a programmer

# Automatic testing - yes or no?

- **Pros**
  - Eliminate repetition
  - Reduce error
  - Quicker results
- **Cons**
  - Effort needed for automation
  - Number of releases expected for testing
  - Maturity of the product

# Rational Functional Tester

installation

# Rational Functional Tester

Example for unskilled programmers ;)