Testing Techniques and Methods
SFWR ENG 3S03: Software Testing

Alicia Marinache

Department of Computing and Software, McMaster University
Canada L8S 4L7, Hamilton, Ontario

Acknowledgments: Slides adapted from Dr. R. Khedri and Dr. R. Paige. Material based on [Per00]

Objectives

- Discuss main testing techniques and methods
- Understand their goals, and how and when to use them

- Testing Methods: high level strategy for conducting testing
- Testing Techniques: procedures to design test cases
- Classified by:
  - System State
  - Code Visibility
  - System Requirements
  - Level of System Detail
  - System Execution
  - . . . and more

Definition (Static)

Testing the code **without** executing it

Goals

- Detect Defects Early
- Ensure Compliance with Standards

Definition (Dynamic)

Testing the code by executing it

Goals

- Verify Functional Behavior
- Identify Runtime Issues
- Validate Performance and Usability

Example: Static/Dynamic testing: come up with an example for testing

- Web Server
- Smartphone App
- A Satellite System

Definition (White Box)

Testing that systematically exploits knowledge of the system to design tests for it

Sometimes, you don't have access to the source code:

- outsourcing or crowdsourcing the testing
- using a COTS product
- integrating an API

Definition (Black Box)

Testing the system without knowledge of its internal design or implementation

Definition (Grey Box)

Testing the system with some knowledge of its internal structure and design

Examples

- Black Box: COTS
- Grey Box: API testing, Web app

Goals

- Black Box:
  - Validate Requirements
  - Ensure User Satisfaction (Acceptance)
  - How: Test externally without code knowledge
  - When: functional and acceptance phases
- White Box:
  - Improve Code Quality
  - Ensure Code Coverage
  - How: Analyze internal logic
  - When: During development or for security and edge cases
- Grey Box:
  - Identify Integration Issues
  - Troubleshoot Complex Bugs
  - How: Combine both approaches
  - When: Integration and system-level

Examples Black/Grey/White Box: come up with an example for testing

- Web Server
- Smartphone App
- A Satellite System

Hint: are they Static or Dynamic?

Definition (Functional)

Testing that the system performs its function(s) correctly and the correctness can be sustained over a continuous period of time

- Goals: Ensure the software meets functional requirements; Identify functional defects
- How: Define test cases based on the functional specifications; Execute tests with input data to verify expected outputs
- When: During development and system testing: To validate individual features or modules; Before release: To ensure the system meets all functional requirements

Definition (Non-Functional)

Testing how the system performs (not what it does)

- Goals: Validate system quality attributes; Ensure a seamless user experience
- How: Simulate scenarios to measure non-functional aspects, like response time or resource usage; Use tools to automate tests and gather data
- When: Post-functional testing: To validate the system's readiness for deployment; During maintenance: To check for regressions in quality attributes after updates

Examples FT/NFT: come up with an example for testing

- Web Server
- Smartphone App
- A Satellite System

Functional Testing Techniques

- (Functional) Requirements testing
- Acceptance testing
- Regression testing
- Error-handling testing
- Manual-support testing
- Integration testing
- Parallel testing

Non-Functional Testing Techniques

- Stress testing
- Execution testing
- Recovery testing
- Operation testing
- Compliance testing
- Security testing

Definition (Manual Testing)

Human testers executing test cases without the help of automation tools

- Goal: Identify any issues that the automated testing cannot; Validate usability and design
- Who: Testers (manual testing team); End-users
- How: Exploratory testing; Predefined test cases

Definition (Automated Testing)

Using tools and scripts to execute test cases automatically

- Goal: Ensure consistency in repetitive tasks; Increase speed of regression testing
- Who: Developers or automation engineers
- How: Test scripts using tools; Integration with testing frameworks

Where do you start testing? One option

Method:

```
int increment(int i){
        return i+1;
}
```

Test

Test:

```
@Test
public void test_1(){
    int j = 0;
    assertEquals(1, increment(j));
}
```

Definition (Unit Testing)

Testing individual components in isolation (to ensure they function as required)

- Goal: Verify individual components
- Who: Usually done by the developer
- When: Done during the development process
- How: Sometimes by creating the tests BEFORE the code (TDD)

Challenges:

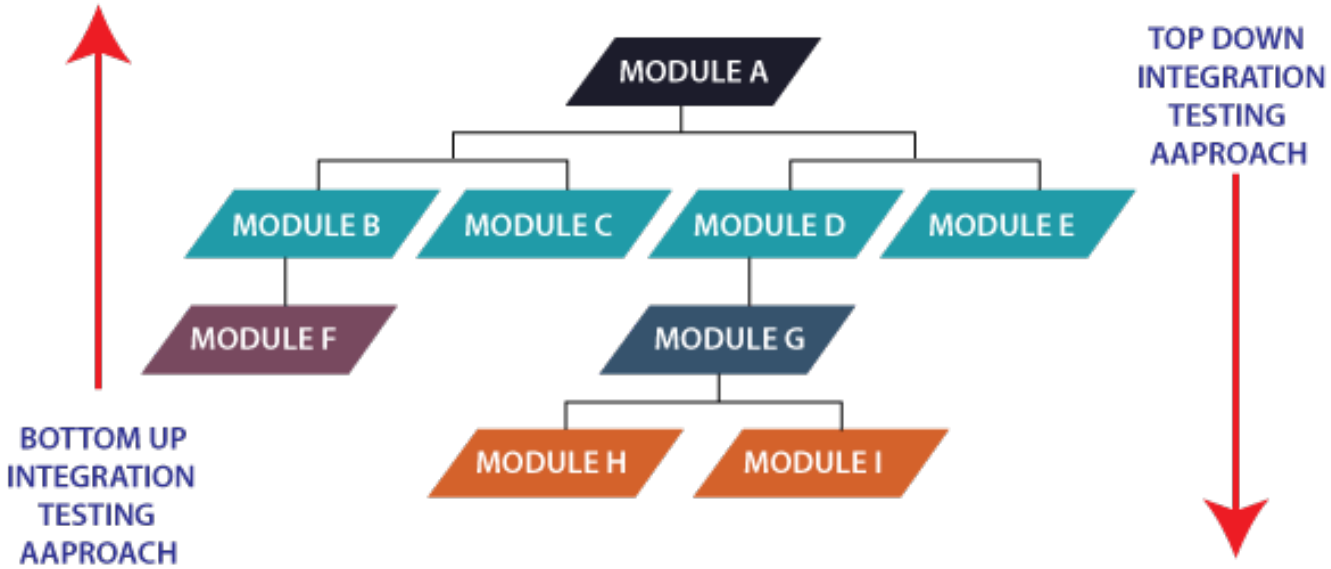
- What is a "unit"?
- Can it catch all defects?

Solution...

Definition (Integration Testing)

The process that evaluates the interaction and data flow between integrated modules or components to ensure they work together as expected

- Goal: Detect integration defects
- Who: Could be engineers who have deep knowledge of the components
- How: Execute integration nodes
- When: Generally after modules have passed Unit Testing



Are we done yet? No, because...

©https://www.javatpoint.com/

Definition (System Testing)

The comprehensive process that tests the system as a whole, to ensure it mets its specification as intended in real-world scenario

- Goal: Verify compliance with requirements
- Who: Usually expert team (not the developers)
- How: Run tests that cover the entire system and use realistic scenarios

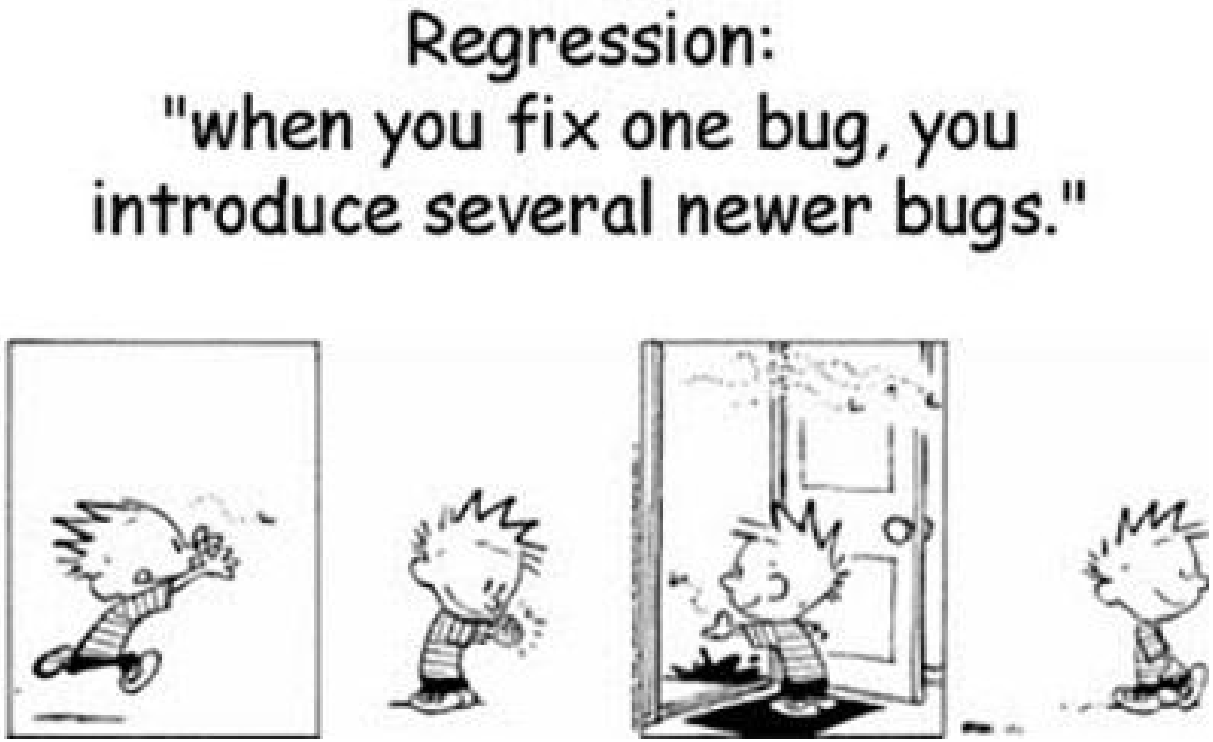
Surely we are done now! Well...

Definition (Acceptance)

The final phase of testing, perform to validate the system against business requirements

- Goal: Business Approval
- Who: End-users, client representatives, or stakeholders
- How: Plan . . . Execute . . . Review Results

You said FINAL, we ARE done, right?



Definition (Regression)

Testing previously tested (segments of a) system to ensure they still function properly after a change has been made

- Goal: System documentation and test data remain current
- Who: It depends on the change
- How: By re-running the original tests

JUnit Example of Regression Testing

For the following examples, think of as many **types** of tests you can perform

Hint: Always ask questions AND record your assumptions

Application: Calculator

Requirements:

- Perform basic arithmetic operations (+, -, *, /)
- Division should throw an error when dividing by zero

Application: Bank ATM Interface

Requirements:

- Withdraw funds if the balance is sufficient
- Deny withdrawal for insufficient funds
- Allow balance inquiry and display correct amounts
- V2: Add Online banking

Application: Online Ticket Booking System

Requirements:

- Handle up to 1,000 concurrent users during peak times
- Ensure response time is under 2 seconds for any request

Application: Social Media App

Requirements:

- Existing features: Posting text, uploading images, liking posts
- New feature: Stories that disappear after 24 hours

Before we end: `bit.ly/4fRCW0k`

- What is the **main point I learned** in this set of lectures?
- What is the **main point that was left out** in this set of lectures?

References I

 William E. Perry, *Effective methods for software testing*, second edition ed., John Wiley & Sons, Inc., 2000.