# Function Testing

## Software Testing

Tóth Zsolt

University of Miskolc

2018

# Table of Contents

1. Introduction

2. Functional Testing
   - Techniques
   - Test Design
   - Example

# Introduction

Types of Testing

- Functional Testing
  - Automated Testing
  - Unit Testing
  - Component Testing
  - Integration Testing
  - Alpha/Beta Testing
- Non-Functional Testing
  - Stress Testing
  - Security Testing
- Structural testing
  - Software Analysis
  - Code Metrics
- Manual Testing

# White–Box

- Source Code is given
- Structural Testing
- Software Analysis
- Code Review
- Techniques
  - Data Stream Monitoring
  - Control Flow Testing
  - Instructions graph
  - Code Coverage
    - Statements
    - Conditions

Levels

- Unit Testing
- Integration Testing
- System Testing

Advantages

- Testing in early stages
- Thorough testing (every details)

Disadvantages

- Programming knowledge is required
- Test maintenance

# Black–Box

- Functional and Non-Functional Testing
- Specification based
- Programming skill not required
- Manual & Automated Testing
- Techniques
  - Boundary Value Analysis
  - Actual and Expected Output Comparison
- Behavioral Testing
- Typical Error Categories
  - Incorrect functions
  - Data structure errors (DTO)
  - Data source errors (DAO)
  - Performance errors

Levels

- Integration
- System
- Acceptance

Advantages

- User's point of view
- Independent from the developers (could be)
- As soon as Specification

Disadvantages

- Superficial Testing (few cases)
- Clear Specification is required
- Overlap White–Box Tests

# Table of Contents

1 Introduction

2 Functional Testing
- Techniques
- Test Design
- Example

# Goal

- Testing the behavior of the software.
- Software verification
- Specification based
- White–box and Black–box Techniques
- Levels
  1. Unit
  2. Component
  3. Integration
  4. System
- Techniques
  - Random Value Generation
  - Partitioning
  - Boundary Value Analysis

# Random Value Generation

- State Space is huge
- Exhaustive search is impossible
- Lot of similar test cases
  - Same error
  - Similar properties
- How to select a test case?
  - Random value generation
  - Design

- Difficult design
- Vague specification
- Statistical approach
  - More tests $\rightarrow$ better coverage
- Probability of errors is different
  - Quadratic Equation Solver $ax^2 + bx + c = 0$ .
  - $P(2 \text{ Solutions}) \approx 0.5$
  - $P(1 \text{ Solution}) \approx 0.0$
  - $P(\text{No Solution}) \approx 0.5$
  - Why?
- A needle in a haystack.

# Partitioning

- Split state space based on input of the software or system
- Partition a.k.a. Equivalence Relation
  - Reflexive $a \rho a$
  - Symmetric $a \rho b \rightarrow b \rho a$
  - Transitive $a \rho b, b \rho c \rightarrow a \rho c$
- Input is sorted into equivalence classes
- Each class is represented by one test case
- Similar test cases in same partition
- Typical Test Cases
  - In/Out partition
  - Boundary values

# Boundary Value Analysis

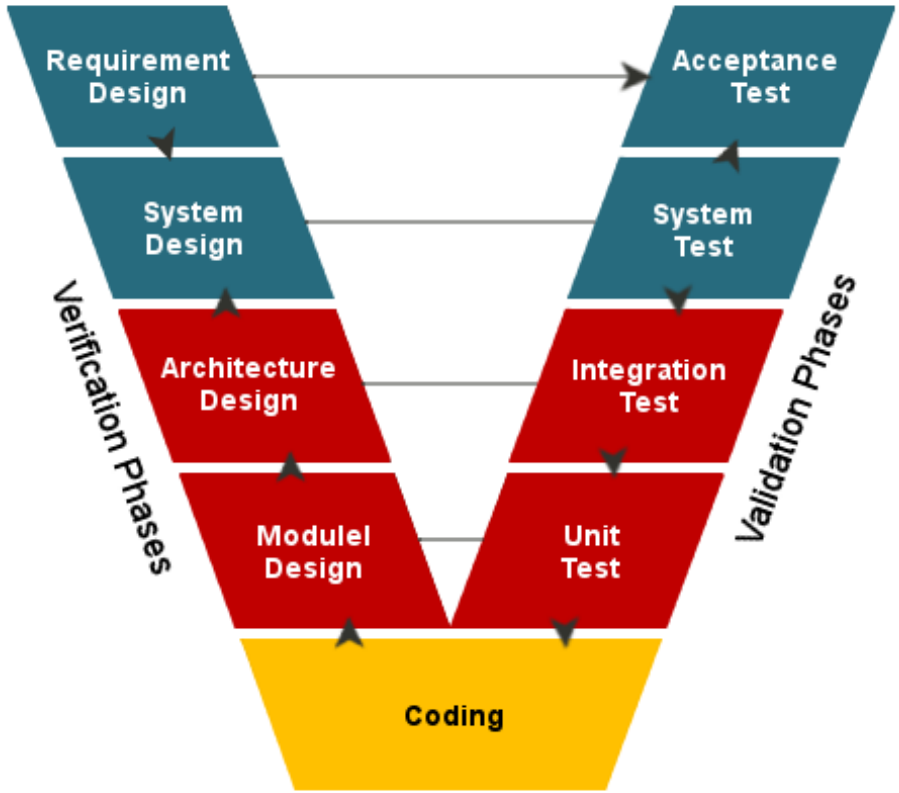
- Part of Partition Testing
- Boundaries of Partitions
  - Operators $<$, $\leq$,, $=$, $\geq$, $>$
  - String length
- Usually 3 test cases
  - Outside
  - Boundary value
  - Inside

Examples

- Can you buy a beer at your 18th birthday?
- Password should be at least 6 characters.
- During cardio exercises the hearth rate is above 152 at age 30.
- Exam has 4 tasks for 10 points:
  - 4 point per task is required.
  - Minimum 20 points are required.

# Test Design

- Parallel with Software Design
- Top–Down vs Bottom–Up
- Levels of Testing
  - Unit
    - ★ Classes separately
  - Component
    - ★ Modules separately
  - Integration
    - ★ Modules together
    - ★ Interfaces & Communication



# Unit Test

- Lowest level
- Each class separately
- Dependency Injection
  - Separation is costly but required
- Faults spread quickly.
- Technique
  - Class and TestClass in same package
  - `protected` access modifier

Shopping Cart

- Shopping cart contains Products
- Shopping cart knows its total value
- Products return with wrong value.
- Shopping cart calculates wrong total value.

# Component Test

- Run with Unit tests
  - Not always distinguished.
  - Component is the development unit
- Restricted to a single module
- Behavior of the module
- Internal dependencies tested together
- External dependencies are separated
- Same Maven goal

Unit test failure

- Unit test fails
- Component test fails

Component test failure

- Unit test is successful
- Component test fails

# Integration Test

- Modules tested together
- No separation
- Number of modules may vary
- System test
  - Integration test
  - Cover all modules
- Interfaces
  - End–point of Components
  - Exceptions
- Communication
  - Distributed Systems
  - Delay, Time Outs
  - Data Formats
- Maven goal `verify`

- Abstract
- Behavioral Test
- Functional
- Black–box approach
- Slow
  - building environment
  - waiting for time outs

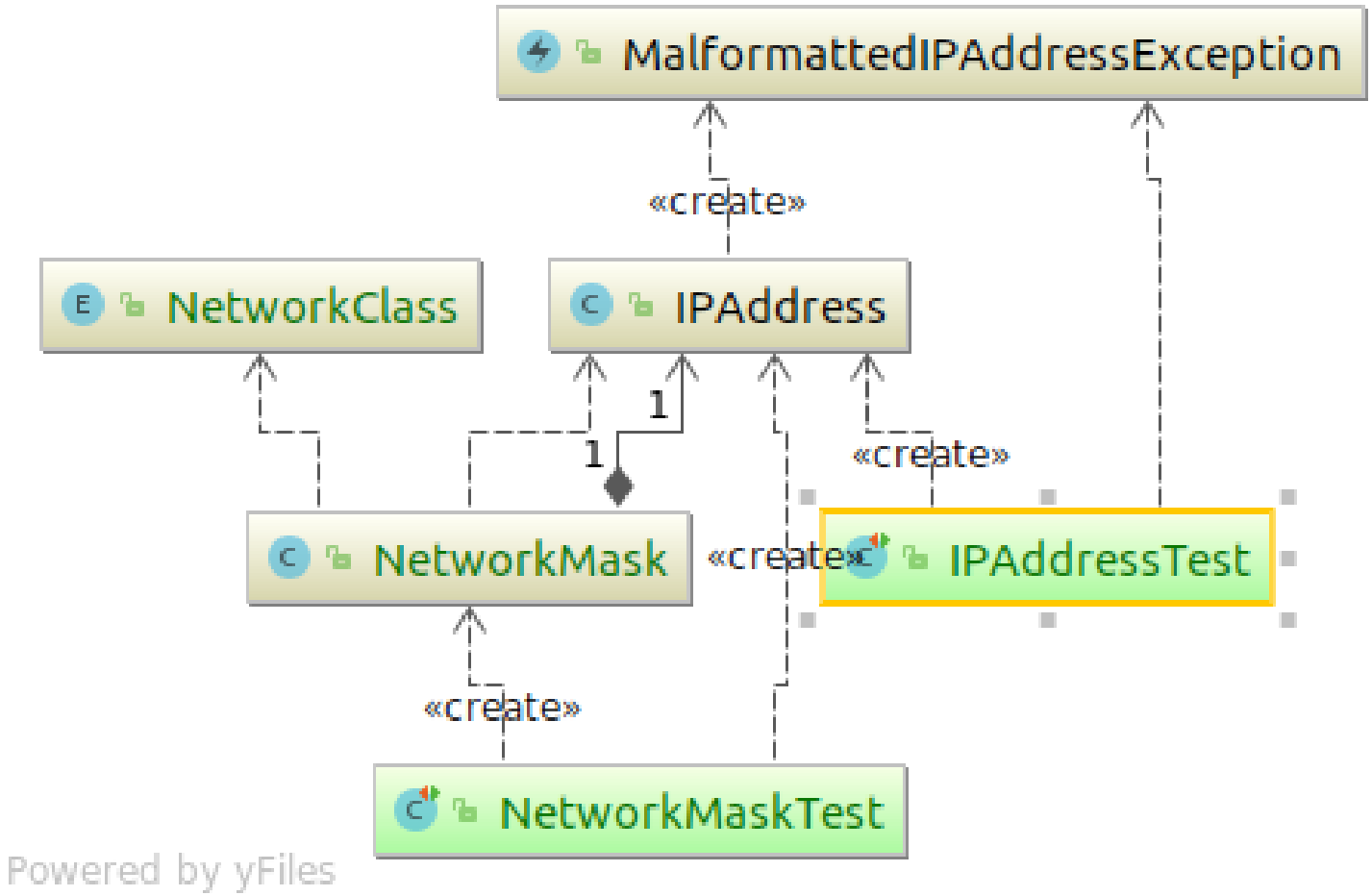
# Separation (Mocking)

- Mocking of behavior
- Substitution of object with trained behavior.
- Errors won't spread

- Programming skill is required
- Mocking Frameworks
  - EasyMock
  - Mockito

# Network Simulator Example

- Representation of IPv4 addresses.
- Checking whether an IP Address is in a network or not.



# IPAddress Class

- Represent an IPv4 address
- 32bit
- Different formats
  - 4 numbers
  - String
- Parsing
  - Length
  - Separation
  - Formation
- Validation
  - Range Check
  - Possible Values

Testing

- Expected Faults
  - Invalid value at any position
  - Length of String
- Boundary Values
  - 0, 255
- Implementation Patterns
  - Method names
  - Condition checking order
  - Condition definition

# NetworkMask

- IPv4 address & Length
- `isInNetwork` method
- `NetworkClass` enum
  - A,B,C
  - enum with fields

Testing

- IPAddres is a dependency
  - Mocking
- Construction
  - Exceptions
- `isInNetwork` method
  - In, out
  - Network Address