Software College Northeastern University

# Software Quality Assurance and Testing

## Chapter 1 Software Quality and Testing Overview

wuchenni@qq.com

# Contents

Chapter 1

## Software Testing Introduction

- 1.1 About Software Quality
- 1.2 About Software Testing
- 1.3 Software Testing Rules

# 1.1 About Software Quality

- What is Quality?
  - – The totality of features and characteristics of a product or service that bears on its ability to satisfy given needs.

# 1.1 About Software Quality

- What is Quality?
  - Quality from the
    - Customer’s viewpoint fitness for use, or other customer needs
    - Producer’s viewpoint meeting requirements

# 1.1 About Software Quality

- In the software world, a metric definition of the term “quality ” has been projected as follows:
  - Quality: the degree to which software conforms to quality criteria.
  - Quality criteria include but are not limited to:

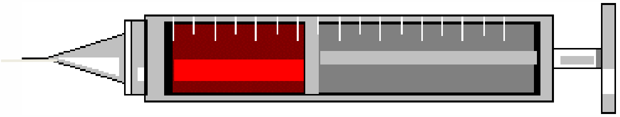
**Economy**
**Integrity**
**Documantation**
**Understandability**
**Flexibility**
**Interoperablity**
**Modularity**
**Correctness**
**Reliability**
**Modifiability**
**Validity**
**Generality**
**Testability**
**Reusability**
**Resilience**
**Usability**
**Clarity**
**Maintainability**
**Portability**
**Efficiency**
**...**

# 1.1 About Software Quality

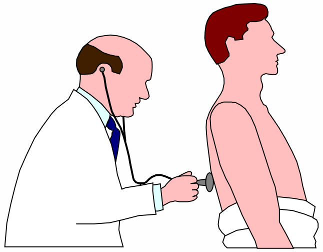
- Quality-Prevention and Detection

Prevention is better than cure . . .

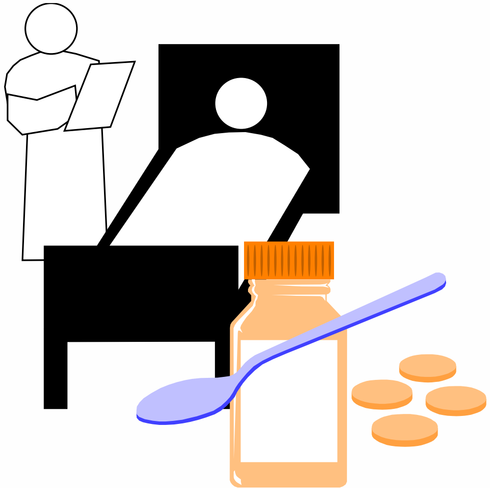
. . . but not everything can be prevented!



Prevention



Detection



Cure

# 1.1 About Software Quality

- Software quality includes activities related to
  - Process
  - Product
- Quality assurance is about the work process
- Quality control is about the product

# 1.1 About Software Quality

## Software Quality

1.满足用户需求；2.建立合理的进度、成本与功能的关系； 3.具备扩展性和灵活性； 4.能有效的处理例外情况； 5.保持成本和性能的平衡

## Software Quality Assurance

为了确保软件开发过程和结果符合预期的要求，而建立的一系列规程，以及依照规程和计划采取的一系列活动，以及结果评价。

## Quality Control

为了达到产品的品质要求所采取的作业技术和活动。

# 1.1 About Software Quality

- QA and QC Broad Difference

QC

Control

Product

Detective

Testing

QA

Assurance

Process

Preventive

Quality Audit

# 1.1 About Software Quality

- It’s all about the End-user

Does this software product work as advertised?

*Functionality, Performance, System & User Acceptance ... Testing*

Will the users be able to do their jobs using this product?

*Installability, Compatibility, Load/Stress ... Testing*

Can they bet their business on this software product?

*Reliability, Security, Scalability ... testing*

# 1.2 About Software Testing

- Software is everywhere

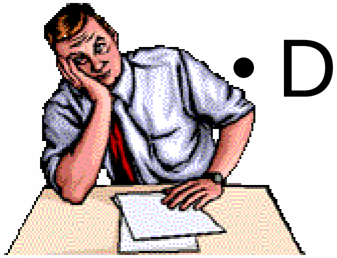
# 1.2 About Software Testing

- 编程大师说：“任何一个程序，无 它多 小，总存在着错误。”
- 初学者不相信大 得只 执行一个简单的
- “这样的 这样 的程序存 个错 误。”
- 但初学 不失效，那么 会怎
- “没有 大师说，“但如果这样的操作系统存在的话，硬件最后将失效，产生一个错误。”
- 初学者仍不满足，再问：“如果硬件不失效，那么会怎样？”
- 大师长叹一声道：“没有不失效的硬件。但如果这样的硬件存在的话，用户就会想让那个程序做一件不同的事，这件事也是一个错误。”

没有错误的程序世间难求

# 1.2 About Software Testing

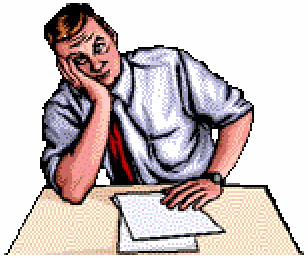
- Qualified rate of products
  - Qualified rate of airplane manufacturing industry
    - “Aerobus747-400” are made up of **1000,000** parts
    - Qualified rate of every part: **99.9999%**
    - Do you know the qualified rate？



$$(99.9999\%)^{1000000} = 36.79\%$$

# 1.2 About Software Testing

- Qualified rate of products
  - Qualified rate of software
    - Coding line: **99%**
    - Write **10,000** lines code
    - Do you know the qualified rate?



$$(99\%)^{10000} = 2.25 \times 10^{-44}$$

# 1.2 About Software Testing

## What is the next ?

### Northeast Blackout (2003)

508 generating units and 256 power plants shut down

### Ariane 5 (1996)

The rocket self-destructed 37 seconds after launch.

### Mars Polar Lander(1999)

Sensor signal falsely indicated that the craft had touched down when it was 130-feet above the surface.

**Infamous software error case studies**

# 1.2 About Software Testing

- Why do we test ?
  - Most security vulnerabilities are due to faulty software.
  - World-wide monetary loss due to poor software is staggering.
  - Stronger testing could solve most of these problems.

# 1.2 About Software Testing

- Why do we test ?
  - We want our programs to be reliable.

# 1.2 About Software Testing

- What is software defect（缺陷）?
- The history of software testing

# 1.2 About Software Testing

- Error（错误）： occurs in the process of writing a program.
- Fault（故障）： is t 表现） of one or more errors.
- Failure（失效） aulty piece of code is executed leading o an incorrect state that propagates （传播） to the program’s output.
- Incident（事故）: no message is displayed when failure occurs.

Defect

# 1.2 About Software Testing

- Software defect（缺陷） definition from IEEE 1983 of IEEE Standard 729

软件产品中所存在的问题，最终表现为用户所需要的功能没有完全实现，没有满足用户的要求。

# 1.2 About Software Testing

- Software defect（缺陷） definition
  - Out of accord with user expectancy（期望）
  - Software function can be executed incorrectly
  - All kinds of software problems
    - E.g. Inconsistency , user interface fault
  - Defect of Software=Bug

# 1.2 About Software Testing

- Defect examples：
  - Shortcoming: running slowly.
  - Inconsistency: Ctrl+S can’t save all applications.
  - User interface design defect: an button should show 5 words on it ,but only 3 words could be seen.

# 1.2 About Software Testing

- The source of defects
  - Requirements definition
  - Design
  - Implementation
  - Support systems
  - Inadequate testing of software
  - Evolution

# 1.2 About Software Testing

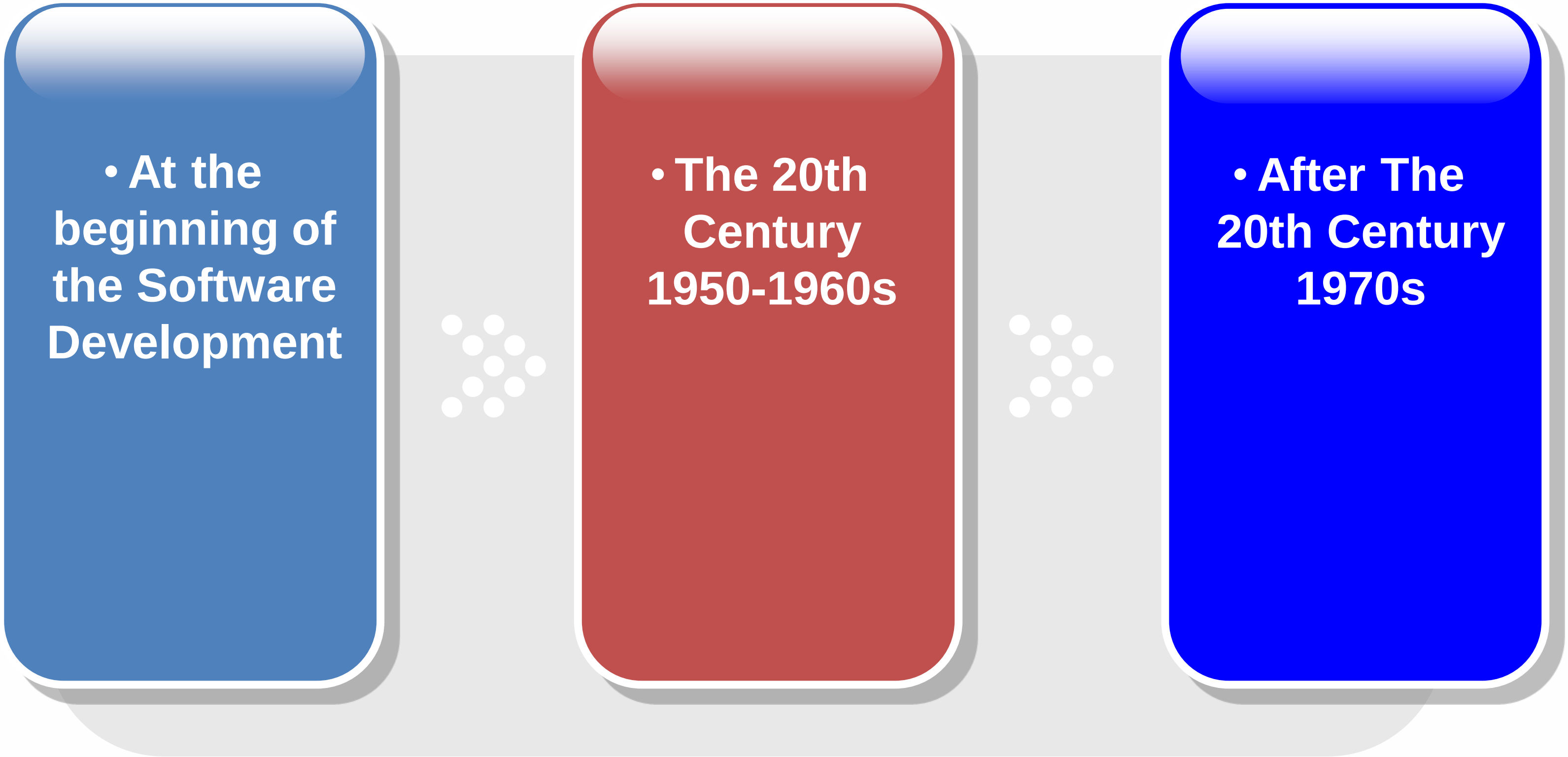
- When does defect occur?
  - – The software does not do something that the specification says it should do.
  - – The software does something that the specification says it should not do.
  - – The software does something that the specification does not mention.
  - – The software does not do something that the product specification does not mention but should.
  - – The software is difficult to understand, hard to use, slow …

# 1.2 About Software Testing

- Correct program
  - No syntax error
  - No obvious errors during running
  - No improper statements
  - Valid input - correct output
  - Invalid input - correct output
  - Any possible input - correct output

# 1.2 About Software Testing

## The History of Software Testing



- At the beginning of the Software Development

- The 20th Century 1950-1960s

- After The 20th Century 1970s

# 1.2 About Software Testing

- Software testing definition
- Verification(验证）&Validation(确认)
- Test & debug
- Purpose of software testing
- Types of testing

# 1.2 About Software Testing

- Software testing definition
  - Software testing is the essential step which is planned and systematic.
  - It is an empirical investigation conducted to provide stakeholders with information about the quality of the product or service under test.
  - We can know whether the users expectations are realized.
  - Software testing is the key step of software quality assurance.

# 1.2 About Software Testing

- Software testing definition
  - The process of exercising or evaluating a system or system components by manual or automated means to verify that is satisfies specified requirements or to identify differences between expected and actual results.

# 1.2 About Software Testing

- Attention：
  - *The essential function of software testing is* *verification*(*验证*）*and* *validation*(*确认*).
    - *•Verification（验证）:  The software should conform to its specification (Are we building the product right?)*
    - *•Validation（确认）: The software should do what the user really requires (Are we building the right product?)*

# 1.2 About Software Testing

- Attention：
  - Verification: The process of determining whether or not the products of a given phase of the software development cycle fulfill the requirements established during the previous phase.
  - Validation: The process of evaluating software at the end of the software development process to ensure compliance with software requirements.

# 1.2 About Software Testing

- Attention：
  - Test & debug
    - Automated test vs. manual operate
    - Don’t know details are OK vs. must know details
    - Correctness proof and how to do with failure vs. correctness proof only
    - Checking vs. reasoning
    - Plan , under control vs. out of control

# 1.2 About Software Testing

- Purpose of software testing, according to the view of G.J.Myers.
  - Find errors of software
  - Decrease the risk of software doesn’t work

# 1.2 About Software Testing

- Types of testing
  - C1:Source of test generation
  - C2:Life cycle phase in which testing takes place
  - C3:Goal of a specific testing activity
  - C4:Characteristics of the artifact under test
  - C5:Test process models

# 1.2 About Software Testing

- C1:Source of test generation

| Artifact | Technique |
|---|---|
| Requirements (informal) | Black-box testing |
| Code | White-box testing |
| Requirements and code | Black-box testing and white-box testing |
| Formal model: graphical or mathematical specification | Model-based specification |
| Component’s interface | Interface testing |

# 1.2 About Software Testing

- C2:Life cycle phase

| Phase | Technique |
|---|---|
| Coding | Unit testing |
| Integration | Integration testing |
| System integration | System testing |
| Maintenance | Regression testing |
| Post system, pre-release | Beta-testing |

# 1.2

- C3:

| Goal | Technique |
|---|---|
| Advertised features | Functional |
| Security | Security |
| Invalid inputs | Robustness |
| Vulnerabilities | Vulnerability |
| Errors in GUI | GUI |
| Operational correctness | Operational |
| Reliability assessment | Reliability |
| Resistance to penetration | Penetration |
| System performance | Performance |
| Customer acceptability | Acceptance |
| Business compatibility | Compatibility |
| Peripherals compatibility | Configuration |
| Foreign language compatibility | Foreign language |

# 1.2 About Software Testing

- C4:Artifact under test

| Characteristic | Technique |
|---|---|
| Application component | Component testing |
| Batch processing | Equivalence partitioning, finite-state model-based testing, … |
| Client and server | Client-server testing |
| Compiler | Compiler testing |
| Design | Design testing |
| Code | Code testing |

# 1.2 About Software Testing

- C4:Artifact under test

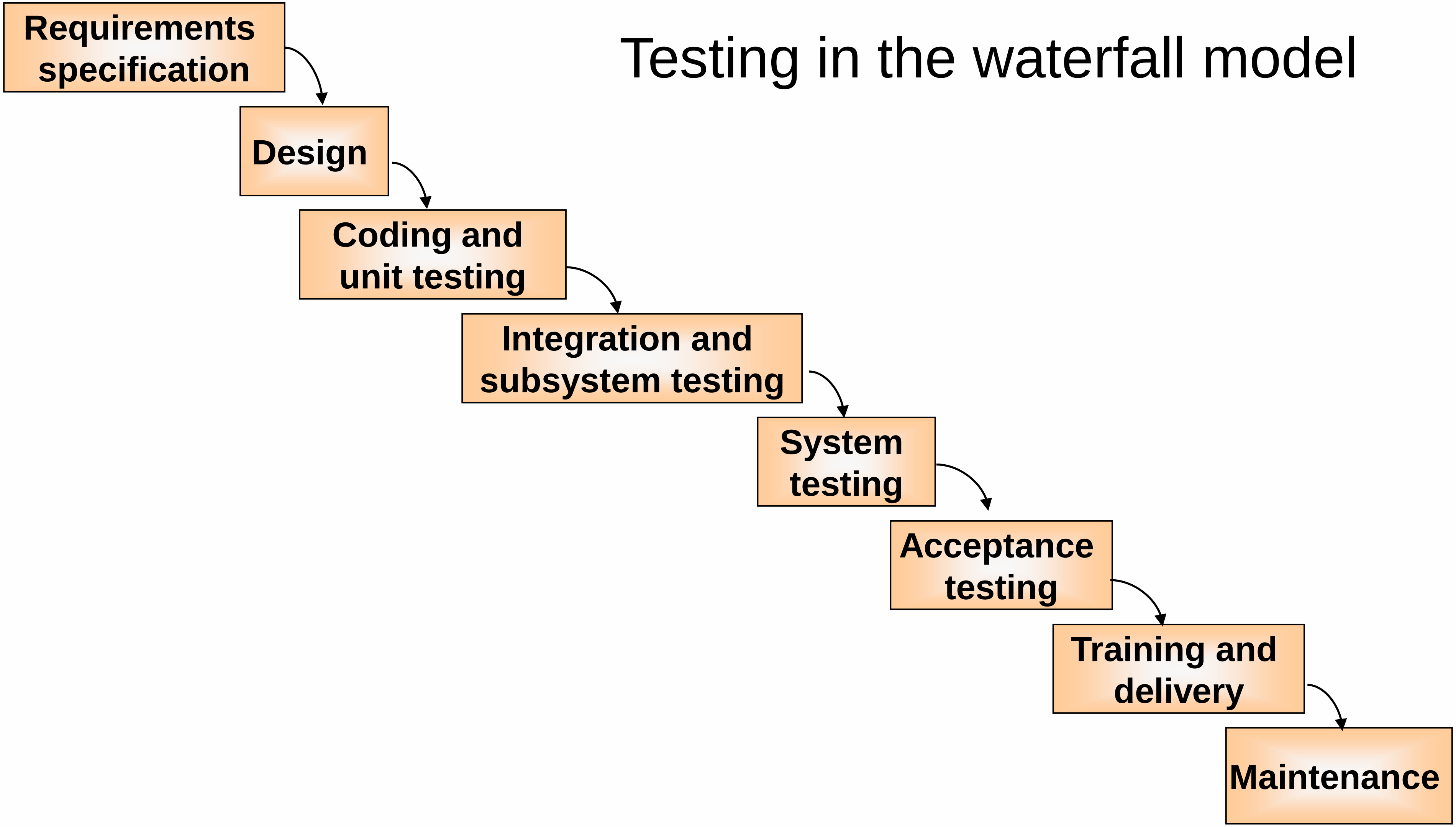
| Characteristic | Technique |
|---|---|
| Database system | Transaction-flow testing |
| OO software | OO-testing |
| Operating system | OS testing |
| Real-time software | Real-time testing |
| Requirements | Requirement testing |
| Software | Software testing |
| Web service | Web-service testing |

# 1.2 About Software Testing

- C5:Test process models
  - Testing in waterfall model
  - Testing in V-model
  - Spiral testing
  - Agile testing
  - Test-driven development (TDD)
    - Requirements specified as tests

# 1.2 About Software Testing

Testing in the waterfall model


Requirements specification

Design

Coding and unit testing

Integration and subsystem testing

System testing

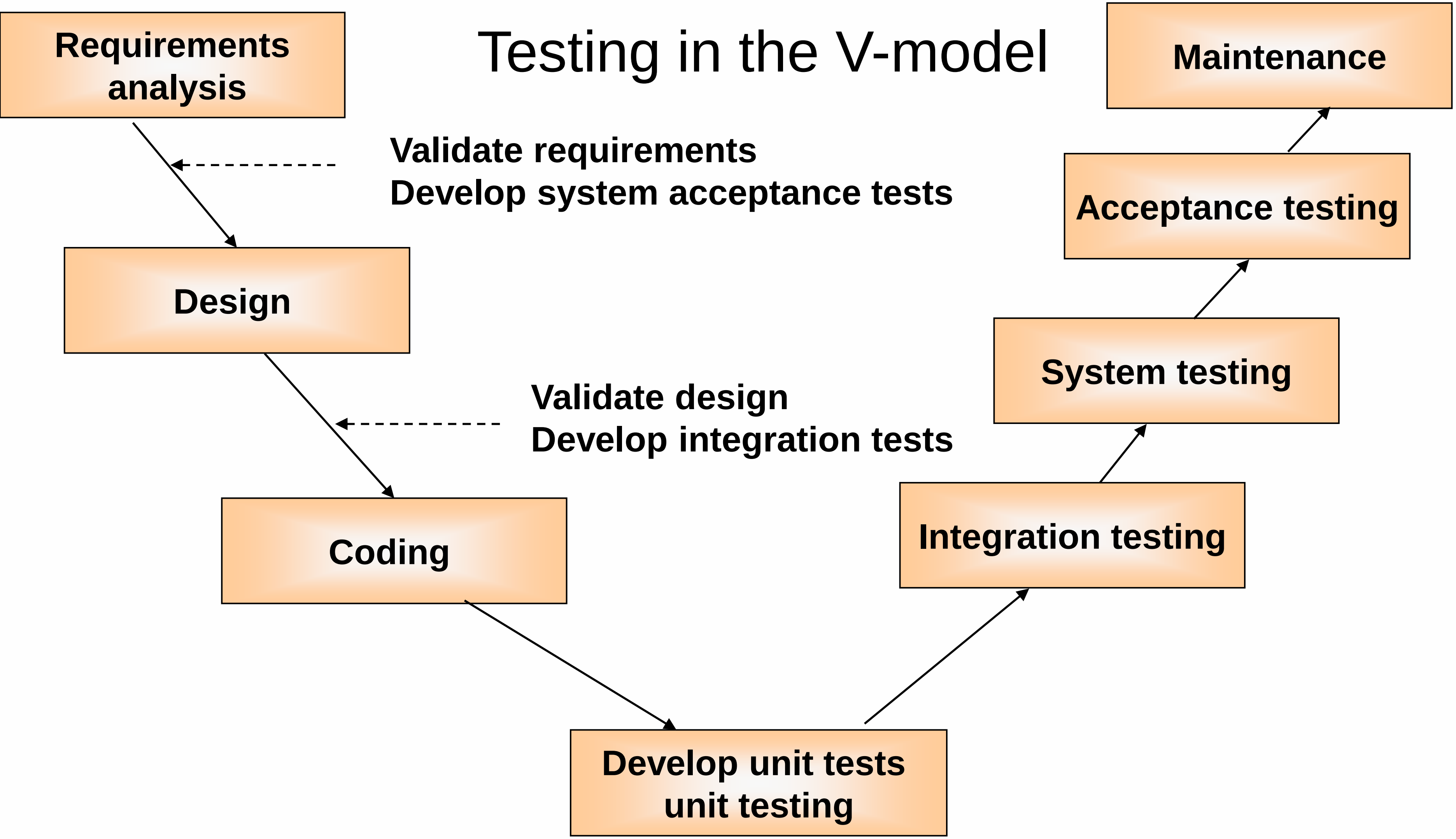
Acceptance testing

Training and delivery

Maintenance

# 1.2 About Software Testing

## Testing in the V-model



Requirements analysis

Validate requirements
Develop system acceptance tests

Design

Validate design
Develop integration tests

Coding

Develop unit tests
unit testing

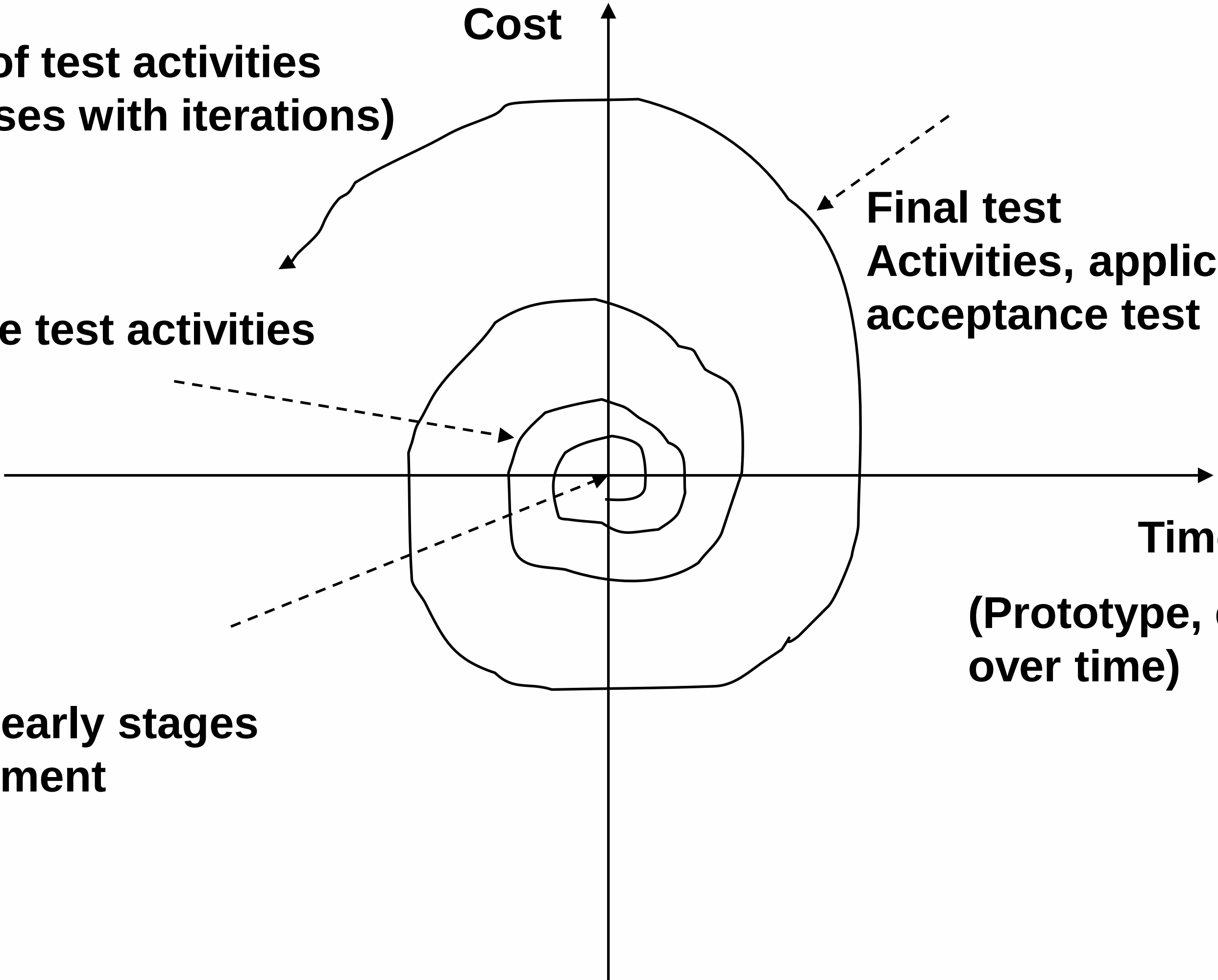
Integration testing

System testing

Acceptance testing

Maintenance

# 1.2 About Software Testing



Cost

(Cost of test activities increases with iterations)

Final test
Activities, application ready for acceptance test

Intermediate test activities

Time

(Prototype, evolves over time)

Initial test
Activities, early stages of development

Spiral testing

# 1.2 About Software Testing

- Agile testing promotes the following ideas
  - Include testing-related activities throughout a development project starting from the requirements phase
  - Work collaboratively with the customer who specifies requirements in terms of tests
  - Testers and developers must collaborate with each other rather than serve as adversaries
  - Test often and in small chunks

# 1.2 About Software Testing

- Example
  - Consider a Web service W to be tested. When executed, W converts a given value of temperature from one scale to another, for example from Fahrenheit scale to the Celsius scale.
    - Regardless of the technique used for test generation, we can refer to the testing of W as Web-services testing(C4).
  - Let’s examine various types of test-generation techniques that could be used for testing W.

# 1.2 About Software Testing

- Supposing tester A tests W by supplying sample inputs and checking the outputs. No specific method is used to generate the inputs.
  - C1 : black-box testing
  - C2: unit testing
  - C3: GUI testing (If W has a GUI to interface with a user)

# 1.2 About Software Testing

- Supposing tester B writes a set of formal specifications for W using the Z notation. The tester generates, and uses, tests from the specification.
  - C1: black-box testing

# 1.2 About Software Testing

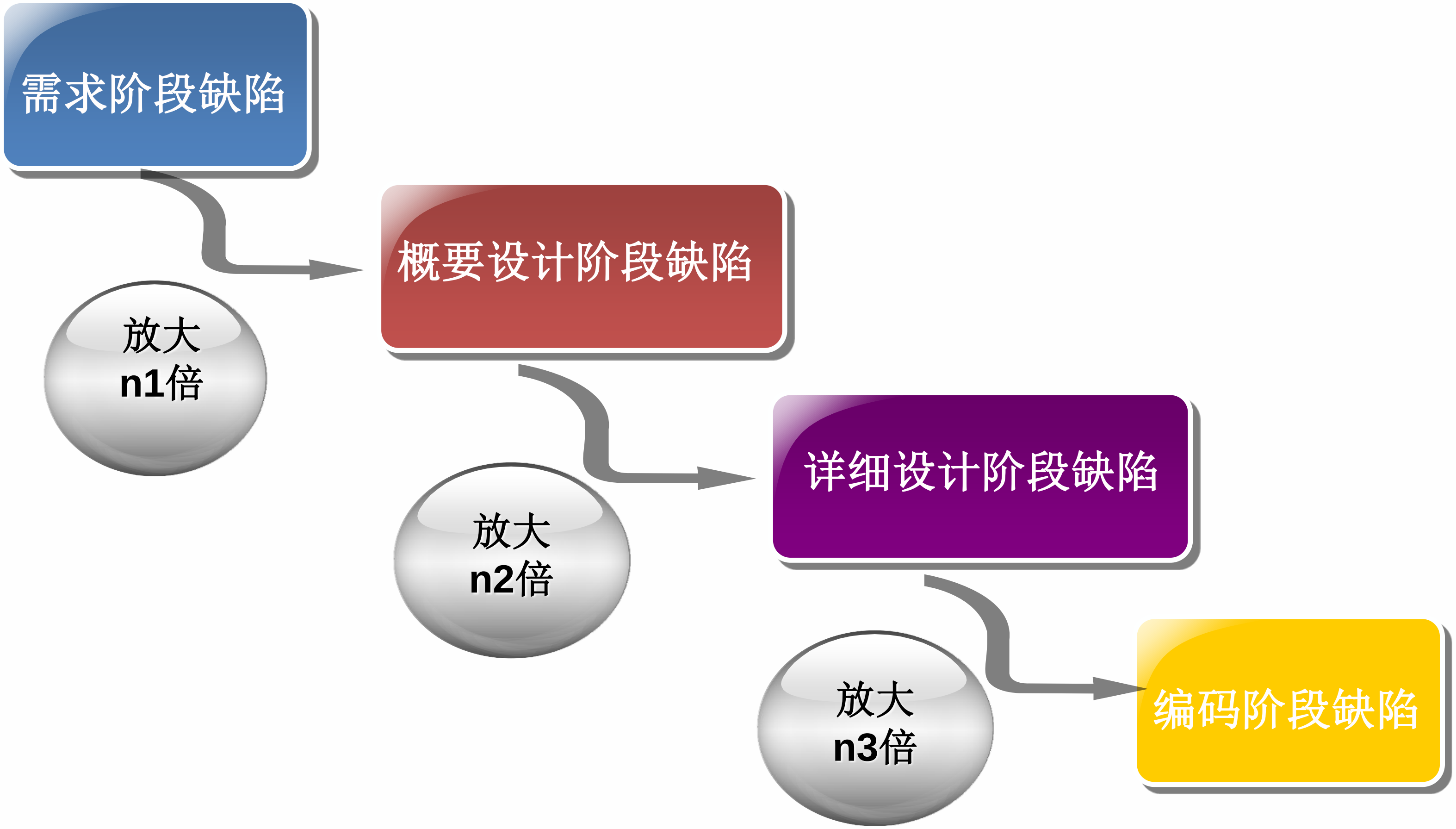
- Supposing tester C generates tests using the formal specifications for W. C then tests W and evaluates the code coverage using one of the several code-coverage criteria. C finds that the code coverage is not 100%
  - C1: black-box testing and white-box testing

# 1.2 About Software Testing

- Supposing tester D tests W as a component of a larger application. Tester D does not have access to the code for W and hence uses only its interface, and interface mutation, to generate tests.
  - C1: black-box testing.

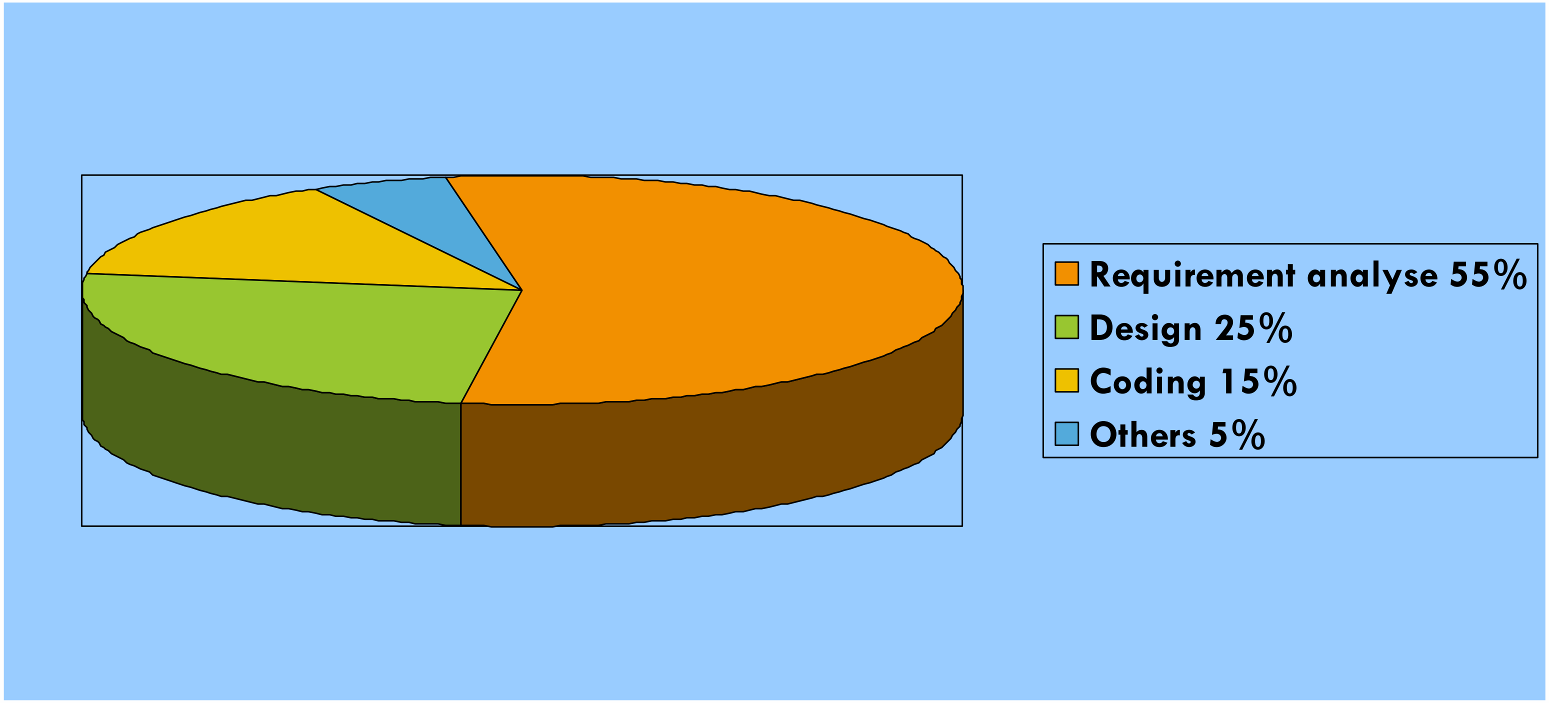
# 1.2 About Software Testing

- IBM研究结果表明：缺陷存在放大的趋势



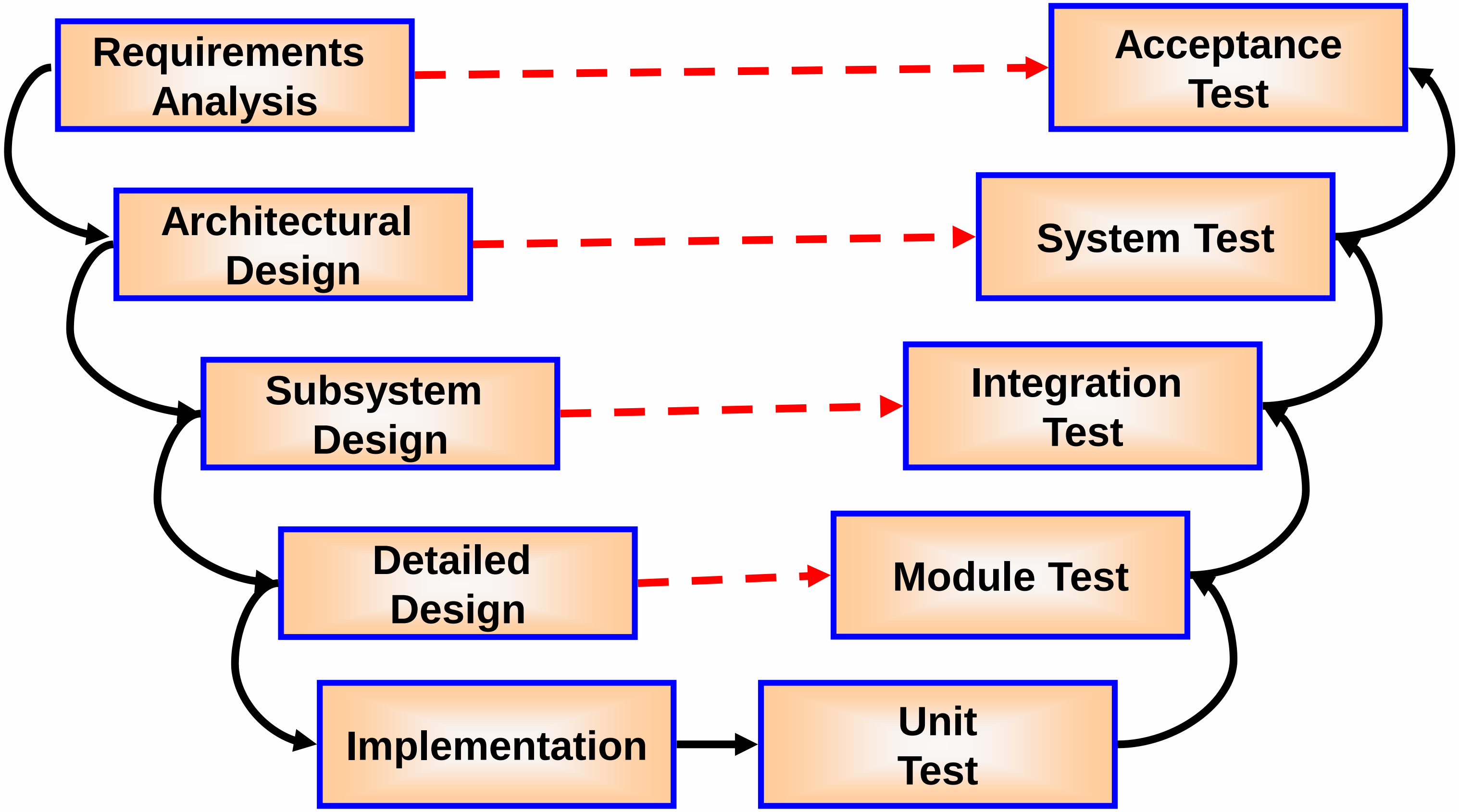
# 1.2 About Software Testing

- Defects can be imported from any phrase during software development, and they will be amplified.



# 1.2 About Software Testing

- The V model of software testing



Requirements Analysis

Acceptance Test

Architectural Design

System Test

Subsystem Design

Integration Test

Detailed Design

Module Test

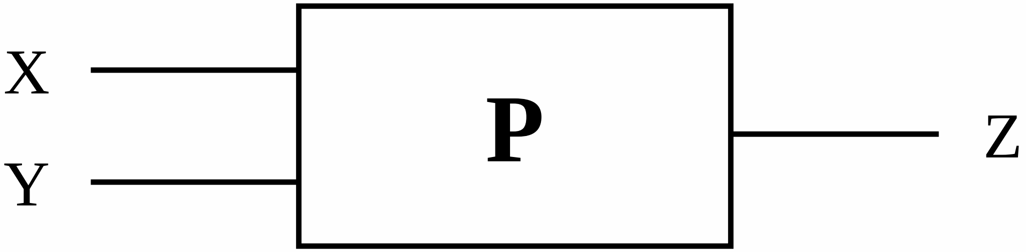
Implementation

Unit Test

# 1.3 Software Testing Rules

- **Infinite test is impossible**

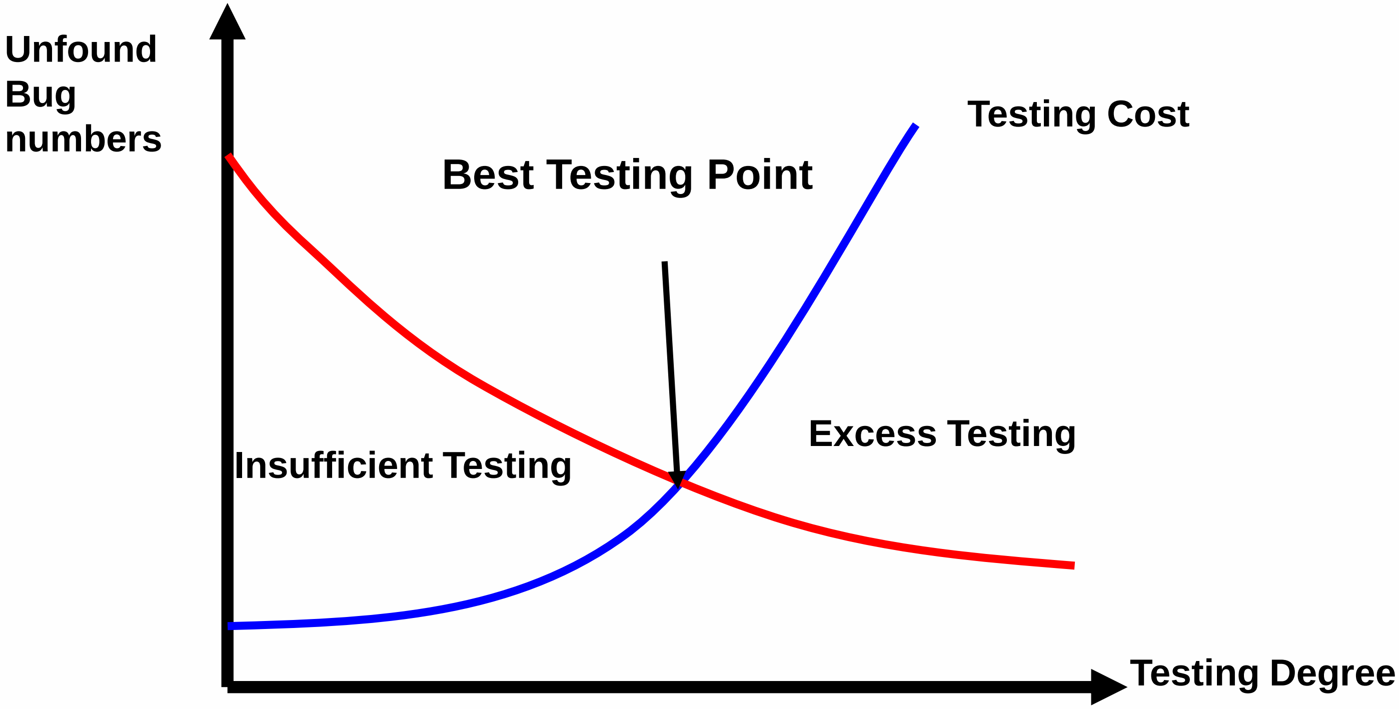
Suppose a program *P* has inputs( *X* and *Y*) and output (*Z*). It run at a computer with 32 bits. if *X* and *Y* are integer, how many time is needed?(1 group data/ms)



$2^{32}$×$2^{32}$ / 365 × 24 × 60 × 60 × 1000 = 5亿年

# 1.3 Software Testing Rules

- **Curved line : testing and cost**



## 1.3 Software Testing Rules

- Good-enough
- 80-20

# 1.3 Software Testing Rules

- Test early and test often.
- Integrate the application development and testing life cycles.
- Formalize a testing methodology.
- Develop a comprehensive test plan.
- Use both static and dynamic testing.

# 1.3 Software Testing Rules

- Static testing

- Dynamic testing

# 1.3 Software Testing Rules

- Define your expected results.
- Understand the business reason behind the application.
- Use multiple levels and types of testing.
- Review and inspect the work, it will lower costs.
- Don't let your programmers check their own work.

# 1.3 Software Testing Rules

- Test passing rules
  - Whether all test cases are executed.
  - Whether function design is finished.
  - Whether we get enough bugs.

# 1.3 Software Testing Rules

- What is test case？
  - –Inputs to test the system and the predicted outputs from these inputs if the system operates according to its specification.

# 1.3 Software Testing Rules

- Test case design rules
  - –Avoid vague test case.
  - –Similar functions should be abstracted and classified.
  - –Avoid complicated test case.

# 1.3 Software Testing Rules

- Capability for Software Tester
  - Technique ability
  - Communication ability
  - Suspicion
  - Confidence
  - Patience
  - Analysis ability
  - Cooperation

# Keystone

- Definition of Software Quality
- Concept of QA & QC
- Definition of Software Defect
- Software Testing Definition
- Types of Testing
- The Roles of  Software  Testing in Software Development
- Testing Rules

# Thank you !