

# Testing Foundation



Global CyberSoft

*A World of Difference*

By **Son Pham**

*July, 2012*

*Duration: 4 hours*

[www.globalcybersoft.com](http://www.globalcybersoft.com)



**Microsoft**  
**GOLD CERTIFIED**  
*Partner*

# Contents

---

- ❑ Course Objectives
- ❑ Basic of Testing Services at GCS
- ❑ Basic of Software Testing
  - What is software testing?
  - Test Principles
  - Fundamental test process
  - The Psychology of testing
  - Test Concepts
- ❑ Basic of SW development process
- ❑ Testing Levels & Test types
- ❑ GCS Testing Process
- ❑ GCS Testing Templates
- ❑ Practices/ Exercises

# Course Objectives

---

At the end of the course, you will be able to understand...

- The basic of Testing Services at GCS
- The basic of software development process
- The testing stages and test types
- The basic Testing process, templates at GCS
- The test managing and escalation mechanism in project
- The documents/tools associated with test activities

# Testing Services at Global CyberSoft

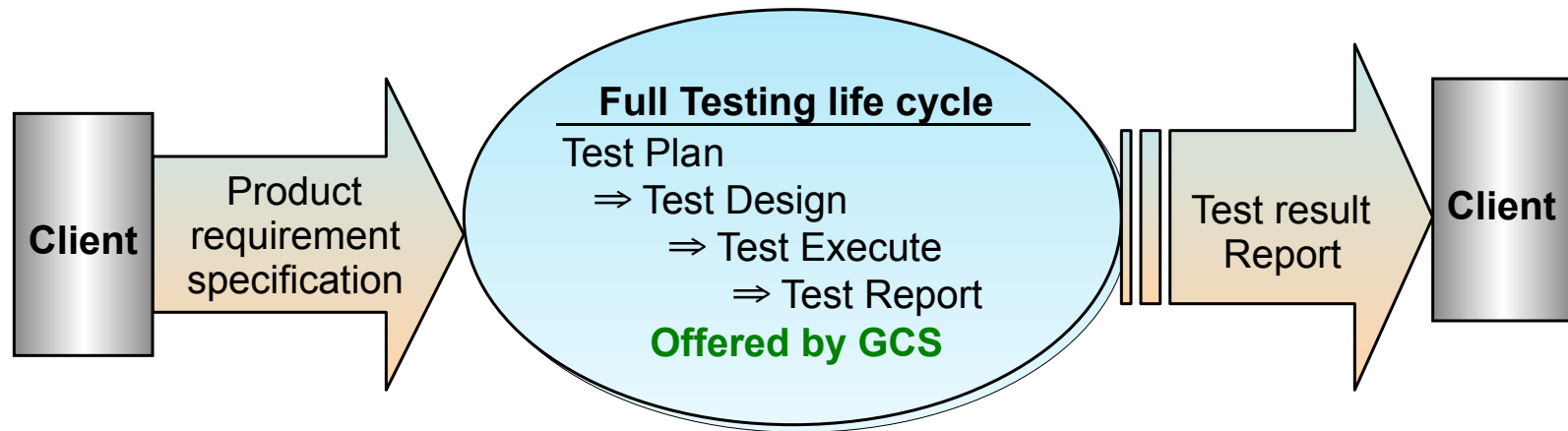
---

## The basic of Testing Services at Global CyberSoft

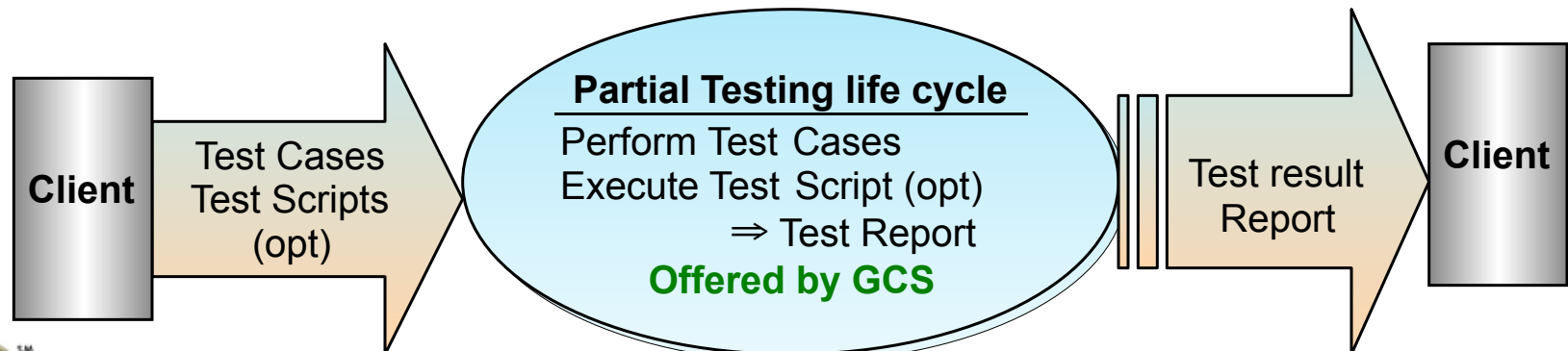
# Four basic testing packages

**There are four most success packages GCS can offer:**

1. Completed package:

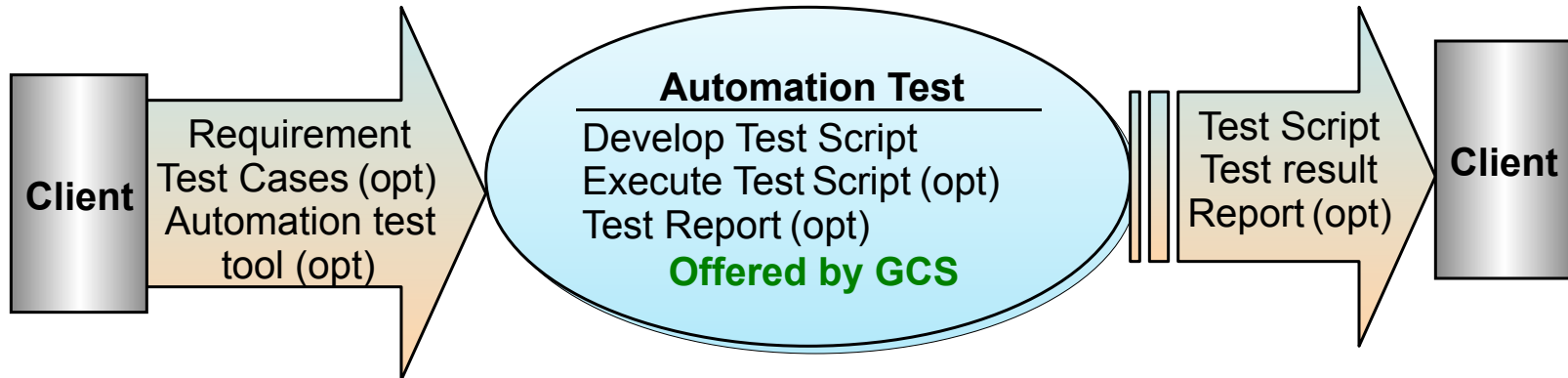


2. Execution package:

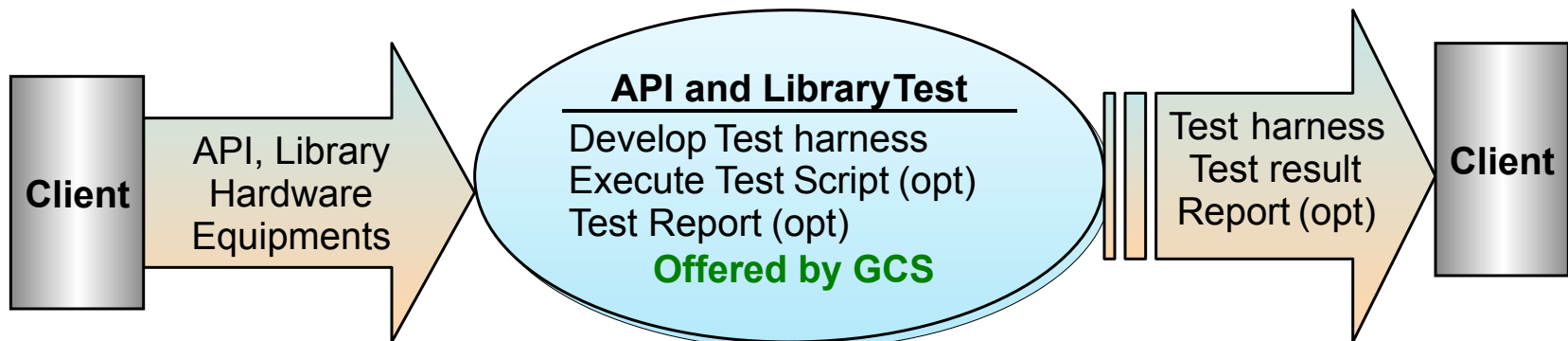


## Four basic testing packages, cont

### 3. Automation testing package:



### 4. API and Library testing package:



### 5. Combination package:

# Things GCS tests

## **Web sites and services**

Web pages, site design  
Web applications, tools  
Web related Plug-ins  
Web-based software package

## **Software**

Internet applications  
Multi-tier software  
E-commerce  
Database  
Client/server applications  
Desktop applications  
Enterprise Resource Planning  
Business Automation  
eLearning solutions  
Network support software

Things GCS can test do  
**not limit** in typical lists  
below

## **Hardware and Network**

Servers desktop/notebook systems  
Security (encryption, certificate, etc.)  
Firewall  
Web appliances  
Wireless  
Network infrastructure products  
Hubs  
Switches & Routers  
LAN modems, Analog modems  
Communications equipment  
ISDN products  
Peripherals  
Printers  
Scanners  
Monitors  
PC cards  
Etc



# GCS testing solutions

## Onshore

- Testing professionals work at client site
- Testing labs on or near client campus
- Testers are part of client teams

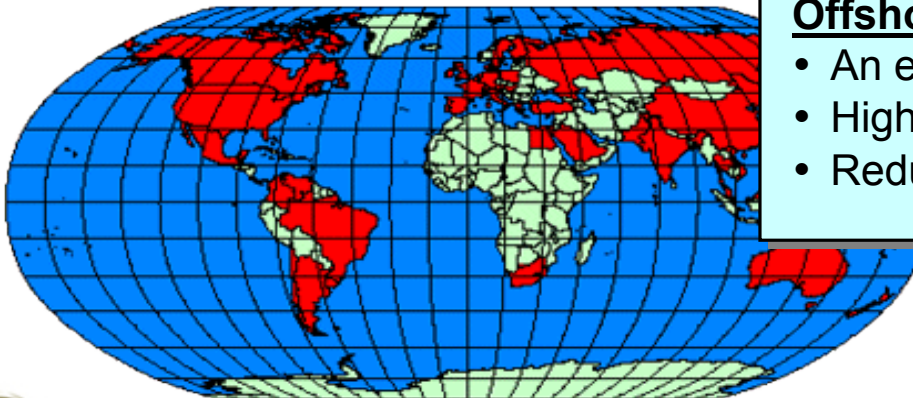
## Offshore

- Testing professionals work either at onsite or/and GCS site
- Testing labs at GCS
- Lowest possible prices

Depending on client needs, GCS can deliver testing services at appropriate solution

## Offshore Test Center

- An extension of Offshore solution
- High security and IP protection
- Reduced cost and maximized efficiency





# Testing project models

## □ Project-based model

- GCS team has full control in planning, designing and executing tests based on test and product requirements.
- A PM or TL can be dedicated by GCS.
- Process for running project is proposed and executed by the GCS, plus appropriate customization to adapt to client needs and project specific factors.

## □ ODC model

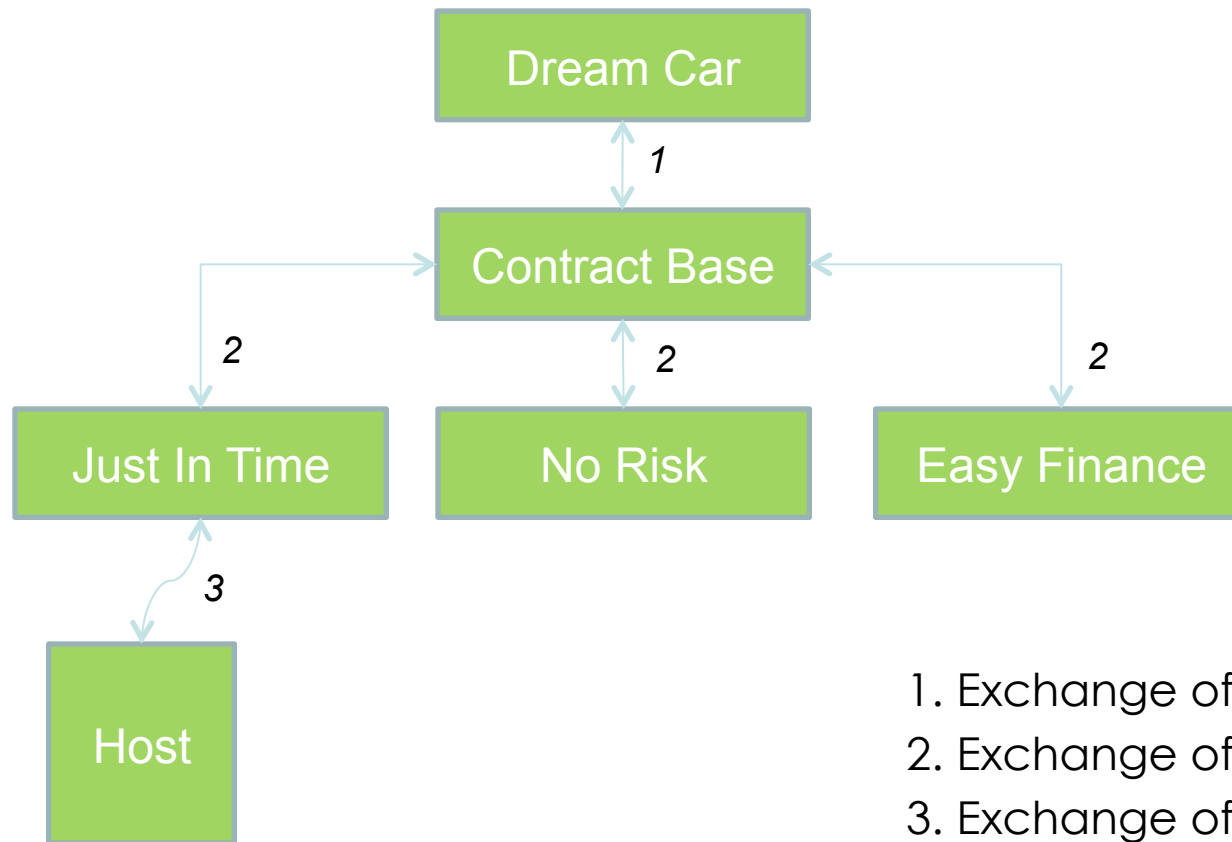
- Client holds the right to manage project members and to fully control all stages of project. The project team executes project activities following task assigned by client.
- A PM or TL mostly come from client, a TL may be from GCS.
- Process for running project is totally proposed by client site, it may combine some processes from GCS

# Basic of Software Testing

- ❑ Why is testing necessary?
- ❑ What is testing?
- ❑ Testing principles
- ❑ Fundamental test process
- ❑ The Psychology of testing
- ❑ Test Concepts

# Case study – Virtual Show Room - VSR

**VSR** is a new electronic sales support system to allow who is interested in buying a new car will be able to configure their favorite model (model , type, color extras..)



1. Exchange of car data
2. Exchange of contract data
3. Exchange of order data

# Basic of Software Testing

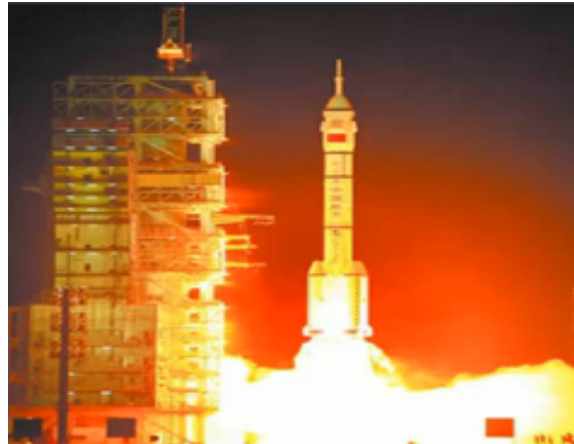
- ❑ **Why is testing necessary?**
- ❑ What is testing?
- ❑ Testing principles
- ❑ Fundamental test process
- ❑ The Psychology of testing
- ❑ Test Concepts

# Why is testing necessary?

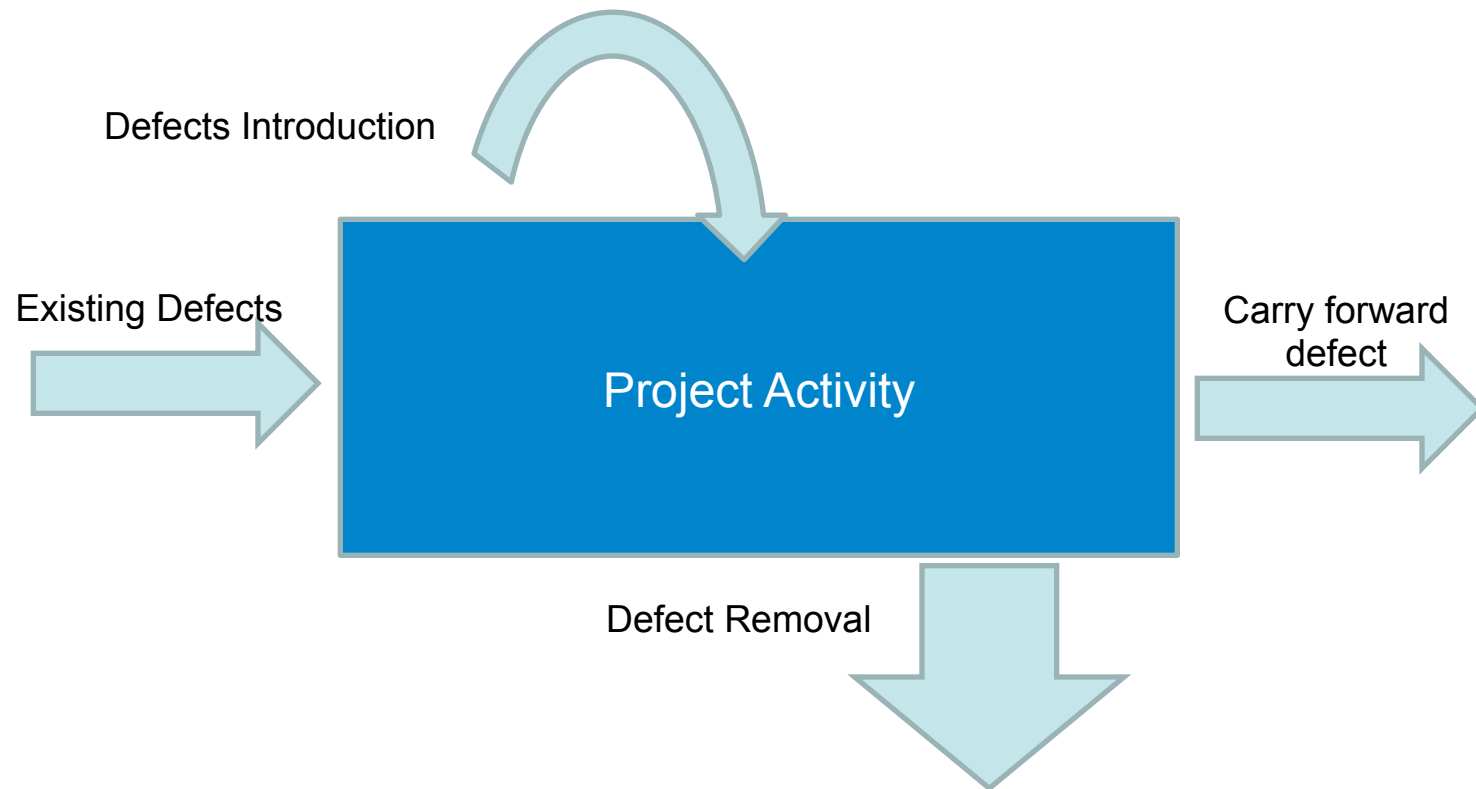
- ❑ How defects happens?
- ❑ Cause of defects
- ❑ Cost of defect and quality
- ❑ Why testing is necessary?
- ❑ Testing and Quality
- ❑ How much testing is enough?

## Test Concepts:

- **Error**, Mistake
- **Defect**, Bug, Fault
- **Incident**, Problem, Failure
- **Risk**
- **Software quality**
- **Testing** and **exhaustive testing**
- **Reliability**
- **Defect Mask**

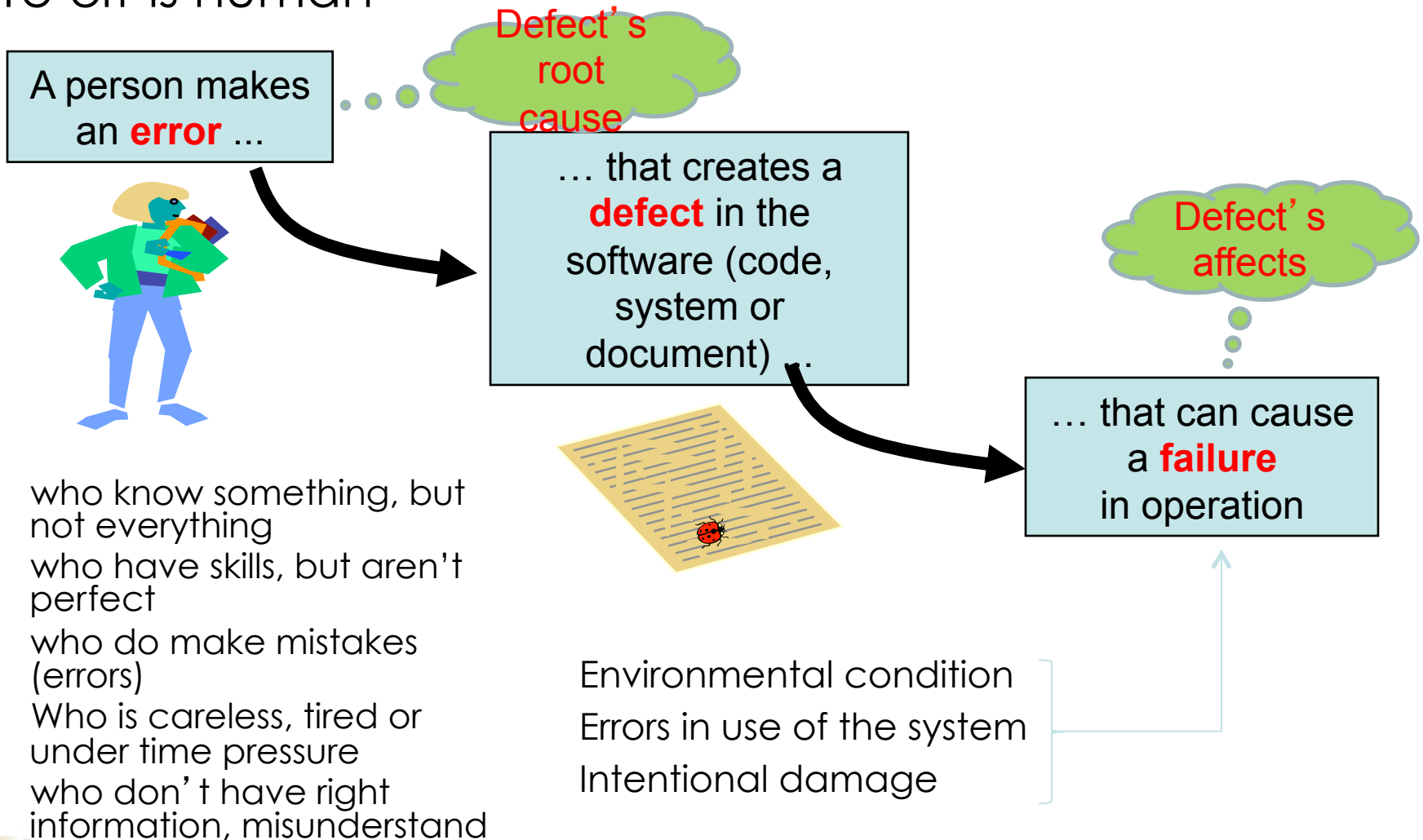


# How defects happen?



# How defects happens?

## □ To err is human



A World of Difference

# Causes of defects

---

□ Some causes may includes:

- Unclear requirement
- Incorrect architecture
- Incomplete design
- Unfamiliarity with programming language
- Inadequate understanding of interfaces, standards (e.g. ODBD, SQL, TCP/IP...)
- Poor document for legacy system
- Design based on outdated requirement
- Poor design for future expansion (e.g hardwired constants, buries assumptions...)
- Code written to outdated design
- Bad fix of a previous problem



# Terms

---

## ❑ Terms:

### ■ Error/ Mistake

- An human action that produce an incorrect result

### ■ Defect/ Bug/Fault:

- A flaw in a system that cause the system to fail to perform it's function.
- Caused by an error or mistake by a person

### ■ Failure/ Incident/ Problem:

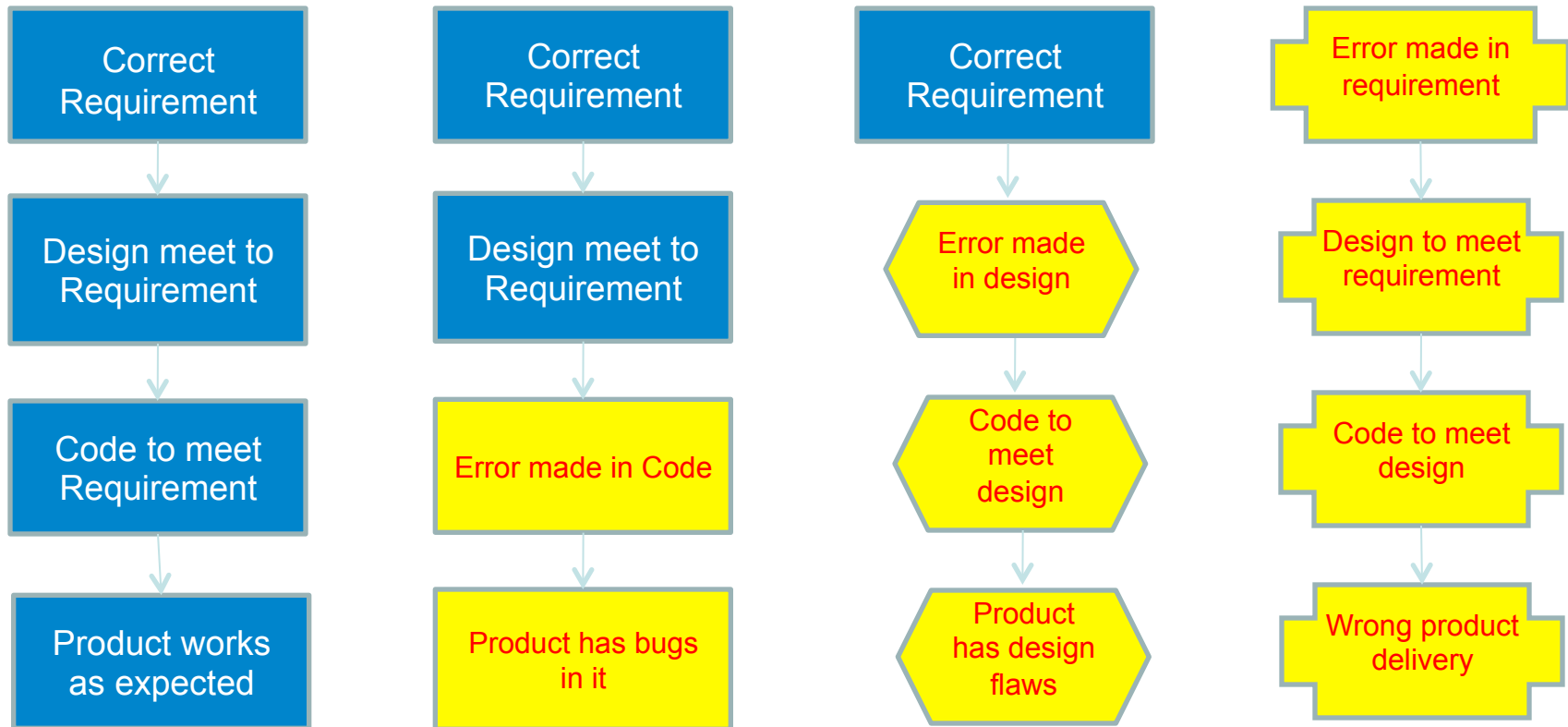
- A non-fulfillment of given requirement; a discrepancy between the actual behavior and user expected behavior.
- Caused by a defect in software or environmental conditions

### ■ Fault masking

Definition: refer to ISTQB Glossary.

# Types of Errors, Defects and failures

## □ Types of errors and defects



# Error, Defect and Failure (cont)

---

## In case study - VSR

### □ User requirement:

- Function 1: build a dream car

**Manufacturer:** {"HONDA", "INNOVA", "KIA MOTORS"}

**Model:** {"Honda Civic 2010", "Honda Civic 2011", "Toyota CAMRY", "Toyota INNOVA", "Kia Motors Cerato Forte 2010"}

**Type:** {"2.0 2008", "2010", "2011"}

**Color:** {"Black", "Silver", "Dark Blue"}

**Extra:** {"item extra 1", "item extra 2".. "item extra 10"}

- Function 2: calculate the total price of dream car

**basePrice:** Basic price of the car

**specialPrice:** a special price

**extraPrice:** price for extra equipment items

**extras:** number of extra items

**discount:** apply only to the base price

+ extra items is more than 3 items: discount of 10% on these particular items only

+ extra items is more than 5: discount of 15% on these particular items only

with extra items which are not a part of the special model chosen

- Environment: using Win XP, Oracle 9i or above

# Error, Defect and Failure (cont)

---

## In case study – VSR

### □ Software Requirement:

#### - UC1 - Build Dream car: basic flow

1. Select Manufacturer
2. Select Model
3. Select Color
4. Select list of extra items
5. Save car configuration

#### - UC2 – Calculate total price of the car with valid values: Basic flow

Precondition: UC1: passed

1. Get basePrice for this car
2. If this model is special Model: get special price
3. Calculate Price:

If number of extralltems  $\geq 3$ : discount = 10

If number of extralltems  $\geq 5$ : discount = 15

If number of extralltems  $< 3$ : discount = 0

### □ Environment: using Win XP, Oracle 8i or above



*A World of Difference*

# Error, Defect and Failure (cont)

## ❑ Coding

```
void calPrice( double basePrice, specialPrice, double extraPrice, int extra, double
discount)
{
    double addon_discount;
    double result;
    If {extras >3} addon_discount = 10;
    else if {extras > 5} addon_discount = 15;
    else addon_discount = 0;

    if (discount > addon_discount)
        addon_discount = discount;

    result = basePrice/100.0 * (100 - addon_discount)
        + specialPrice
        + extraPrice/100.0 / (100 - addon_discount);

    return result;
}
Double test_calPrice ()
{
    double price;
    bool test_result = TRUE;
    // Discount in the case of five or more extra items
    price = calPrice(10000.00, 2000.00, 1000.00, 7, 10);
    return price;
}
```

// Some of the seldom-occurring insurance prices can not be calculated at all b/c implementing the corresponding calculation was forgotten in the insurance component



What do you think were errors and defects, potential failures in VSR example?

The errors are.....

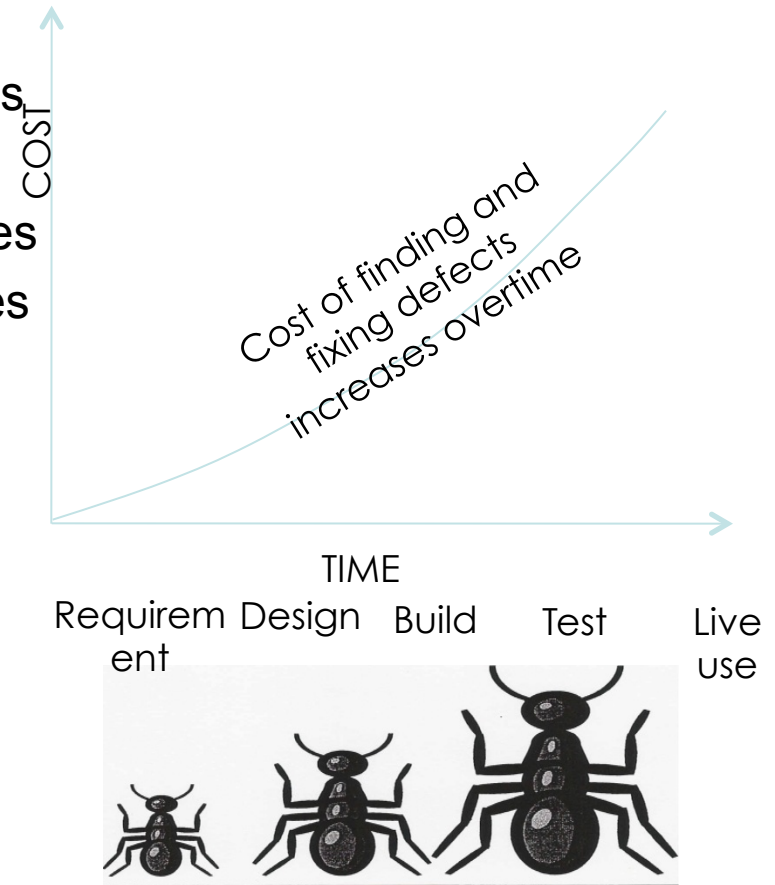
The faults are.....

The failures are.....



# Cost of defects

- ❑ Minor impact:
  - Likelihood of customers to demand refunds
- ❑ Moderate impact
  - Downstream effects of poor quality on sales
  - Larger impact to sales based on references
- ❑ Critical financial or human life impact
  - Automotive software failure (\$7.5 million)
  - Kill or be harmful to person



**In VSR:** If calculation formula is wrong, the system will calculate a wrong price and the customer can insist on that price and business can loss money depending on how much the price was miscalculated by the VSR system for each car.



# Why is testing necessary? - Best examples

---

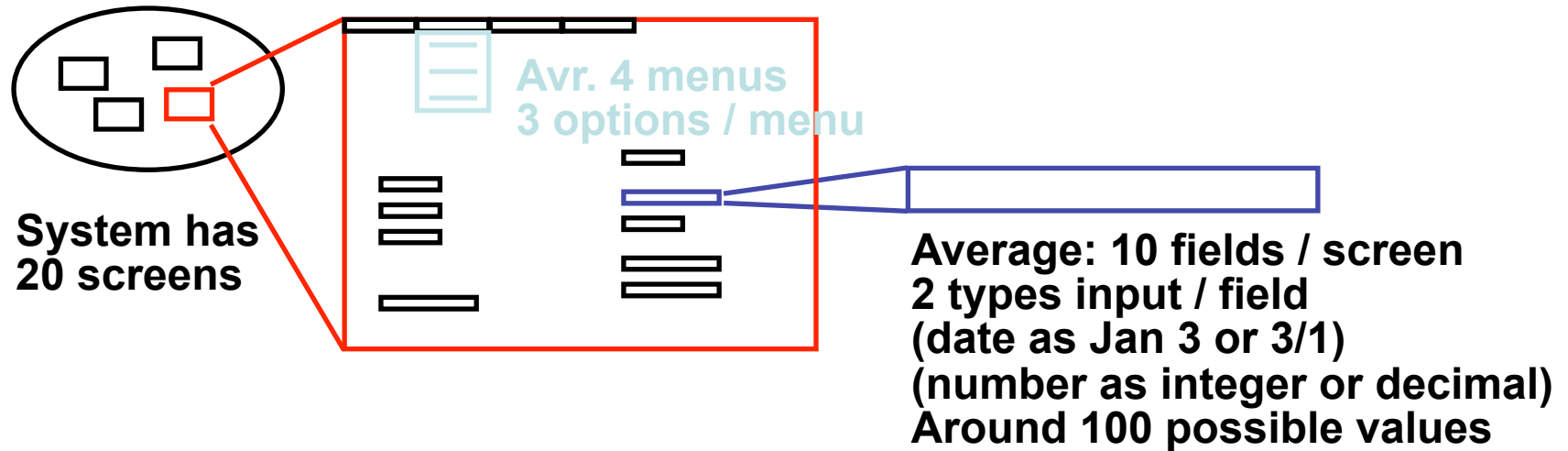
- ❑ Save money
- ❑ Save time
- ❑ Improve quality

## Examples:

- ❑ Safety critical systems:
  - Aircraft producer
  - Radiation treatment (Therac-25)
  - Satellite launch
  - Bank systems
    - A bank which makes 5% of its revenue on ATM fees suffers an ATM network outage due to a software defect
- ❑ Modern software can contain over a million lines of code. According to industry studies, the average C program introduces one bug for every 25 lines of code.



# Why not just “test everything”?



❑ Total for '**exhaustive**' testing:

$$20 \times 4 \times 3 \times 10 \times 2 \times 100 = 480,000 \text{ tests}$$

If 1 second per test, 8000 mins, 133 hrs, **17.7 days**

(not counting finger trouble, faults or retest)

❑ 10 secs = 34 wks, 1 min = 4 yrs, 10 min = 40 yrs



# How much testing is enough?

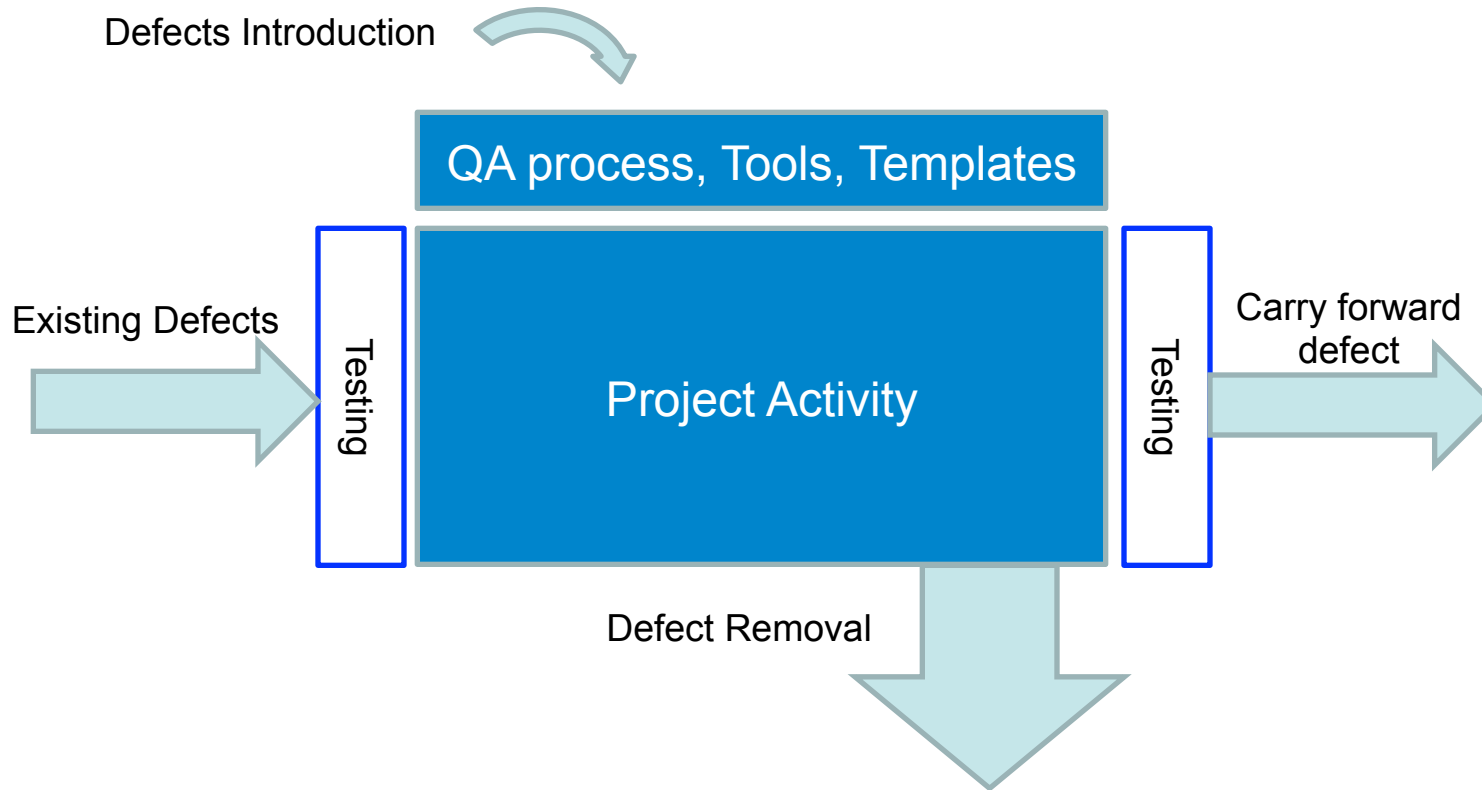
- ❑ It depends on RISK
  - risk of missing important faults
  - risk of incurring failure costs
  - risk of releasing untested or under-tested software
  - risk of losing credibility and market share
  - risk of missing a market window
  - risk of over-testing, ineffective testing
- ❑ Use RISK to determine:
  - what to test first
  - what to test most
  - how thoroughly to test each item
  - What not to test (this time)
- ❑ Use RISK to allocate the time available for testing by prioritizing testing...

**It is difficult to determine  
how much testing is enough  
but it is not impossible**

# Basic of Software Testing

- ❑ Why is testing necessary?
- ❑ **What is testing?**
- ❑ Testing principles
- ❑ Fundamental test process
- ❑ The Psychology of testing
- ❑ Test Concepts

# What is testing?



## Test Concepts:

- **Quality Assurance**
- **Quality Control**
- **Software Quality**
- **Testing process**

# What is testing?

---

- ❑ Testing is a process consisting of all lifecycle activities, both static and dynamic:
  - Planning
  - Preparation and evaluation of software products and related work products
  - Determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects.
  
- ❑ **Purposes of Software testing**
  - To find defect → The most important
  - To measure quality
  - To provide confident
  - To prevent defects by analyzing document

# Testing Objectives

---

- ❑ Development testing (e.g. component, integration and system testing):
  - to cause as many failures as possible so that defects in the software are identified and can be fixed.
- ❑ Acceptance testing
  - to confirm that the system works as expected, to gain confidence that it has met the requirements.
- ❑ Maintenance testing
  - no new errors have been introduced during development of the changes.

# Testing Objectives

---

- ❑ Operational testing
  - to assess system characteristics such as reliability or availability.
  
- ❑ In some cases
  - to assess the quality of the software (with no intention of fixing defects), to give information to stakeholders of the risk of releasing the system at a given time.

# Testing Terms

---

- ❑ Testing is not Debugging
  - Testing: Show failures that are caused by defects.
  - Debugging: the development activity that identifies the cause of a defect, repairs the code and checks that the defect has been fixed correctly.
  
- ❑ The responsibility for each activity
  - Testers test
  - developers debug

# Testing Terms

---

## ❑ Software Quality

- Quality is “**fitness for use**”

(Joseph Juran)

- Quality is “**conformance to requirements**”

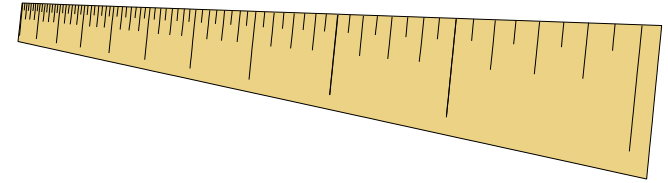
(Philip B. Crosby)

- Quality of a product or services is its ability to satisfy the needs and expectations of the customer



# Testing and Quality

- ❑ Testing measures software quality in terms of the number of defects found, the tests run and the system covered by the tests
- ❑ Testing can find faults, software quality is improved
- ❑ Software Quality is measured based on its attributes

**Functionality:**

- Suitability
- Accuracy
- Interoperability
- Compliance
- Security

**Efficiency:**

- Time Behavior
- Resource Behavior

**Reliability:**

- Maturity
- Recoverability
- Fault Tolerance

**Maintainability:**

- Analyzability
- Scalability/  
Changeability/
- Stability
- Testability

**Usability:**

- Learn ability
- Understandability
- Operability
- Attract ability

**Portability:**

- Adaptability
- Ease of Installation
- Conformity
- Interchangeability
- Replace ability



# Basic of Software Testing

- ❑ Why is testing necessary?
- ❑ What is testing?
- ❑ Testing principles
- ❑ Fundamental test process
- ❑ The Psychology of testing
- ❑ Test Concepts

# Testing principles

---

## ❑ Testing show presence of defects

- Show defects are presence but cannot prove there are no defects
- Even if no defect are found, this is no proof that there are no defects

## ❑ Exhaustive testing is impossible

- Test all combinations of inputs and preconditions is not feasible:  
Execution time and Costs will rise exponentially
- Instead of exhaustive testing ? use risk and priorities to focus testing efforts

## ❑ Early testing

- Testing activities should start as early as possible in the development lifecycle and should be focused on defined objectives

# Testing principles

---

## ❑ Defect clustering

- A small number of modules contain most of the defects discovered during prerelease testing or show the most operational failures

## ❑ Pesticide paradox

- Repeat over and over same set of test case will no longer find any new bugs
- To overcome this “pesticide paradox” ? the test cases need review and revised, new and different tests need to be written to exercise different parts of software under test

# Testing principles

---

## ❑ **Testing is context dependent**

- Testing must be adapted to the risk inherent in the use and environment of the application
- No two system should be tested in the exactly same way

## ❑ **Absence-of-errors fallacy**

- Fallacy of assuming that no failures means a useful system
- Finding and fixing defects does not help
  - if the system built is unusable
  - does not fulfill the users' needs and expectations

# Basic of Software Testing

- ❑ Why is testing necessary?
- ❑ What is testing?
- ❑ Testing principles
- ❑ Fundamental test process
- ❑ The Psychology of testing
- ❑ Test Concepts

# Fundamental Test Process

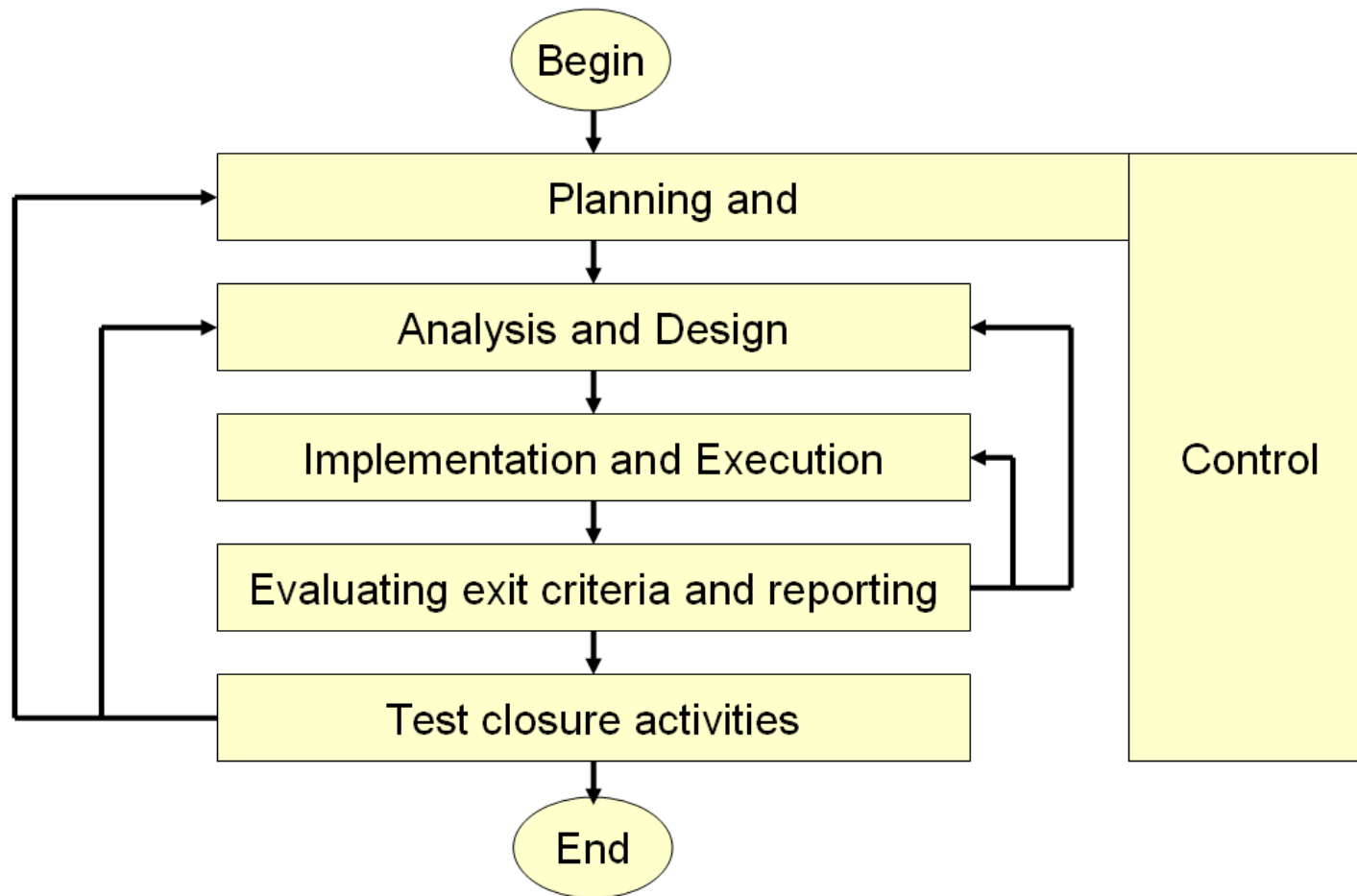
---

## Test Concepts:

- Test policy
- Test Plan
- Test Strategy
- Test Approach
- Test Coverage
- Testware
- Test Basic
- Test condition
- Test case
- Test design
- Test execution
- Test Log
- Test summary report
- Incident

# Fundamental Test Process

## □ Testing process and activities





# Fundamental Test Process

- ❑ Test planning and control
  - Set goal and objectives for testing, and derive and approach and plan for the test
- ❑ Test specification can be broken down into three distinct tasks:
  1. **Test analyst:** determine 'what' is to be tested
    - review test basis
    - **identify test condition**
  2. **Test design:** determine 'how' the 'what' is to be tested
    - **Design the test case, test input, expected result (using test design techniques)**
    - Design the test environment set-up, required
  3. **Test implementation**
    - **Develop and prioritize test cases**
    - Develop test data
    - Develop test script
    - Create test suite
    - Implement and verify the test environment

Why specifying expected results in advance is important?

How to prioritize the test?



# Fundamental Test Process

---

- ❑ Test execution
  - Execution the test suite and test cases
  - most important ones first
  - would not execute all test cases if
    - testing only fault fixes
    - too many faults found by early test cases
    - time pressure
  - can be performed manually or automated
- ❑ Evaluating exit criteria and reporting
  - Check test logs against the exit criteria specified in test plan
  - Assess if more tests are needed or exit criteria should be changed
  - Write a test summary report
- ❑ Test closure activities
  - Check which planned deliverables we delivered and ensure all defects are resolved.
  - Finalize and archive testware
  - Handover testware to maintenance team if needed
  - Evaluate how testing went and analyze lessons learned

# Fundamental Test Process

---



In case study VSR, we have 5 sub systems.

1. What test strategy for each?
2. Create test specification for the DreamCar sub system
  - Identify test conditions and prioritize the test condition
  - Design test cases
  - Build test cases (Opt)



- What is a good test case?

# The Psychology of testing

---

- ❑ Some psychological factors that influence testing and its success
  - Clear objective for testing
  - The proper roles (PM, test lead, test designer, tester...)
  - Balance of self-testing and independent testing
  - Clear and courteous communication and feedback on defects

# The Psychology of testing

---

- ❑ Need good interpersonal skills to communicate factual information about defects, progress and risks in a constructive way.
  - Without criticizing the person who created it
    - Don't goad – you are not perfect either
    - Don't blame
    - Be constructive critical and discuss
  - Explain we can work round it so the delivered system is better for client
  - Start with collaboration rather than battles.

# Summary: Key Points

---

- ❑ Testing is necessary because people make errors
- ❑ The test process: planning, specification, execution, recording, checking completion
- ❑ Independence & relationships are important in testing
- ❑ Re-test fixes; regression test for the unexpected
- ❑ Expected results from a specification in advance
- ❑ Prioritise to do the best testing in the time you have

# Question?

---



# Basic of SW development process

- ❑ Sequential development model
- ❑ Iterative
- ❑ Incremental

## Test Concepts:

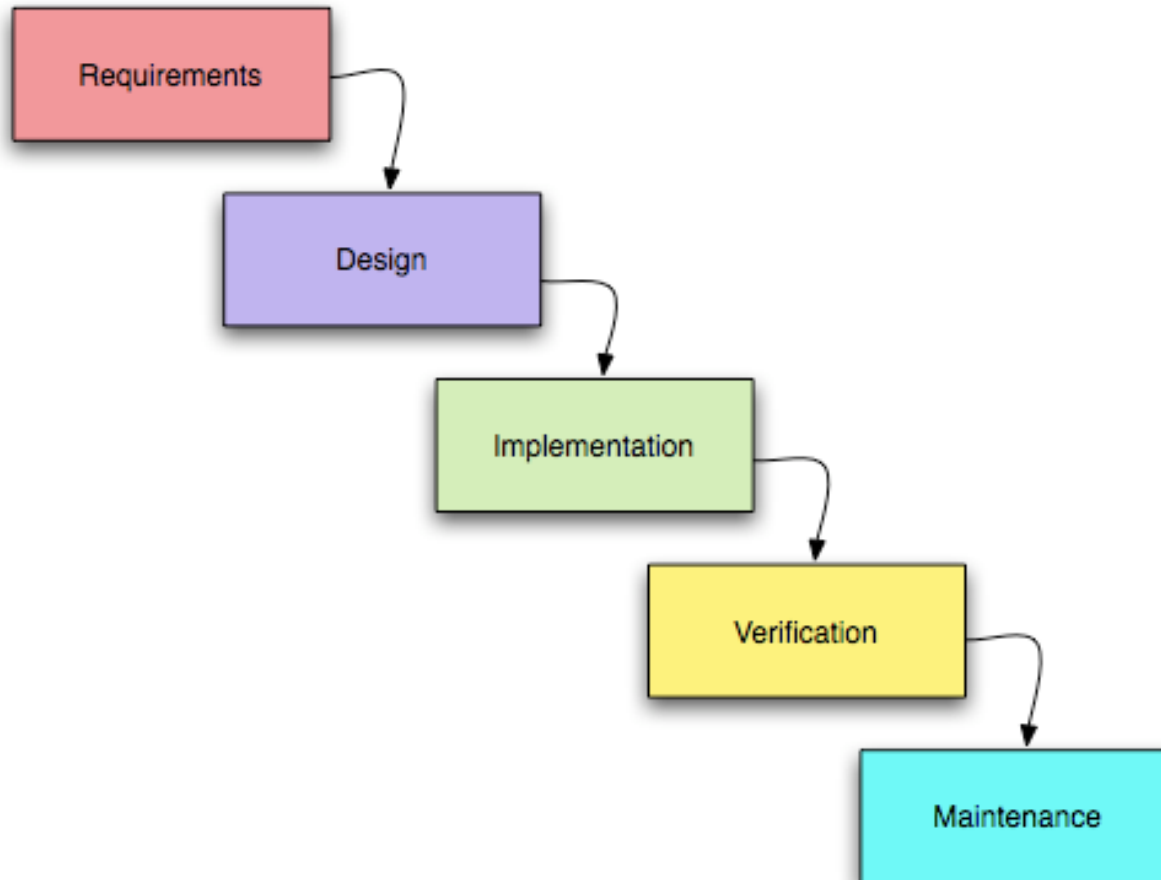
- **Validation**
- **Verification**



# Sequential development model (1)

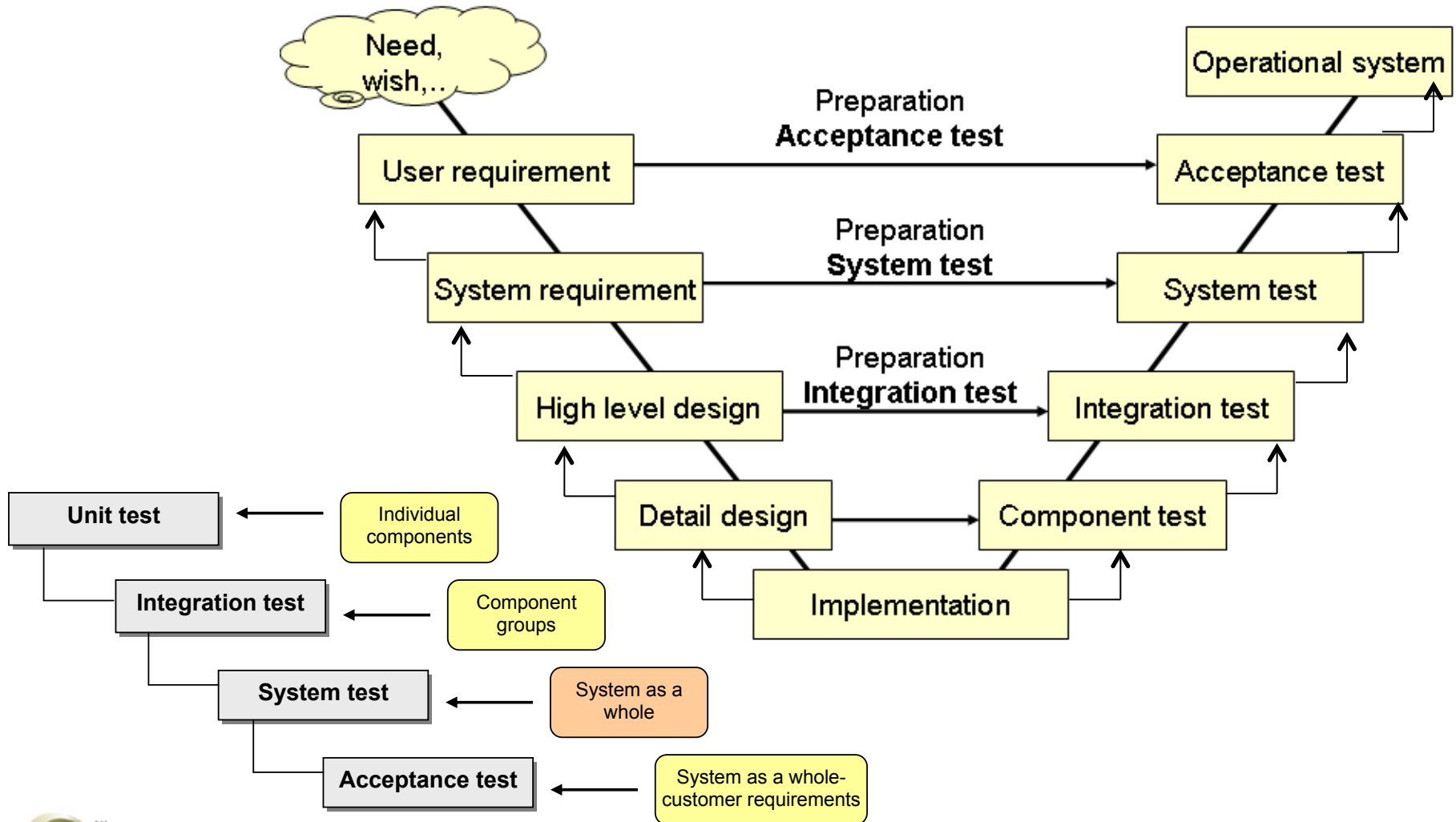
---

## □ Waterfall model



# Sequential development model (2)

## □ V Model



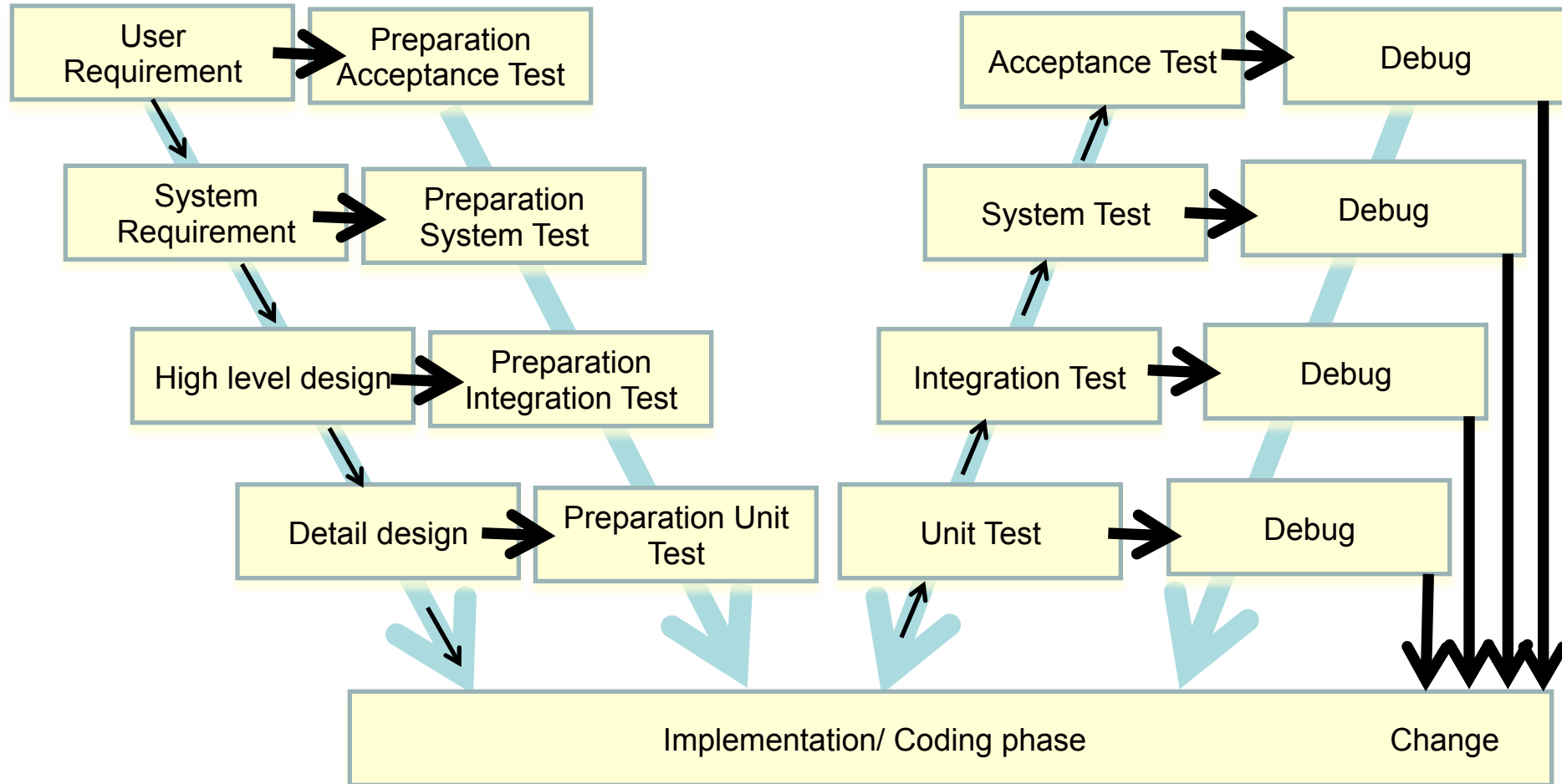
## Fundamental testing process (Cont.)

---

- In every development life cycle, testing focus on **verification** testing and **validation** testing
  - **Verification:** is concerned the outcome of particular work product, component or system is meet the requirement set.  
→ *Are we building the system right?*
  - **Validation:** in concerned with evaluating a work product, component or system is meet the user needs and requirement.  
→ *Are we build the right system?*

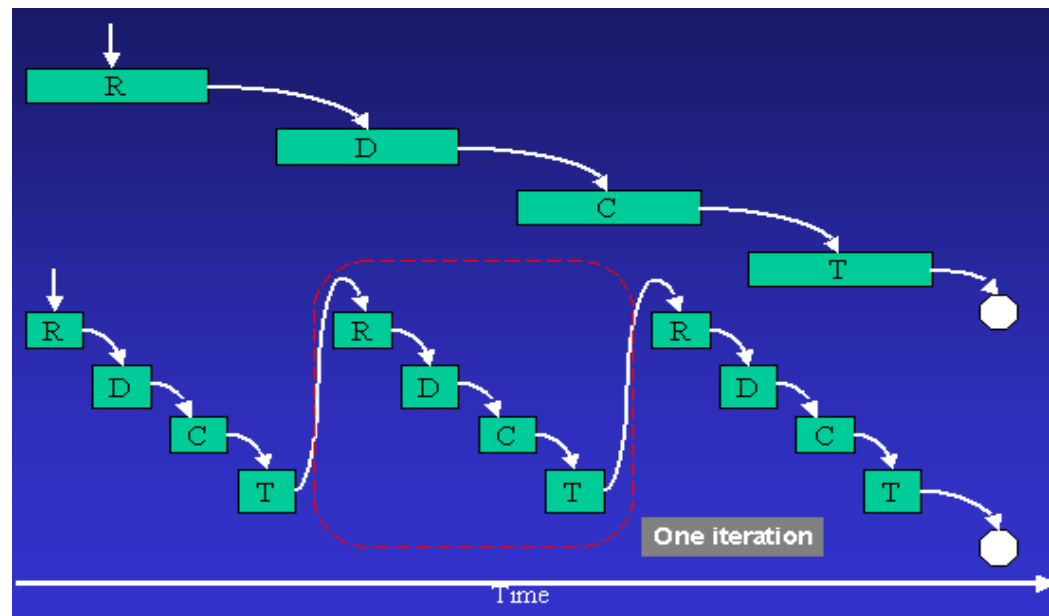
# Sequential development model (3)

## □ W-Model



# Iterative development Model

- The process of establishing requirement, designing, building and testing a system, done as a series of smaller development.
- Some models:
  - Rational Unified Process (RUP)
  - Spiral model
  - Rapid Application Development (RAD)

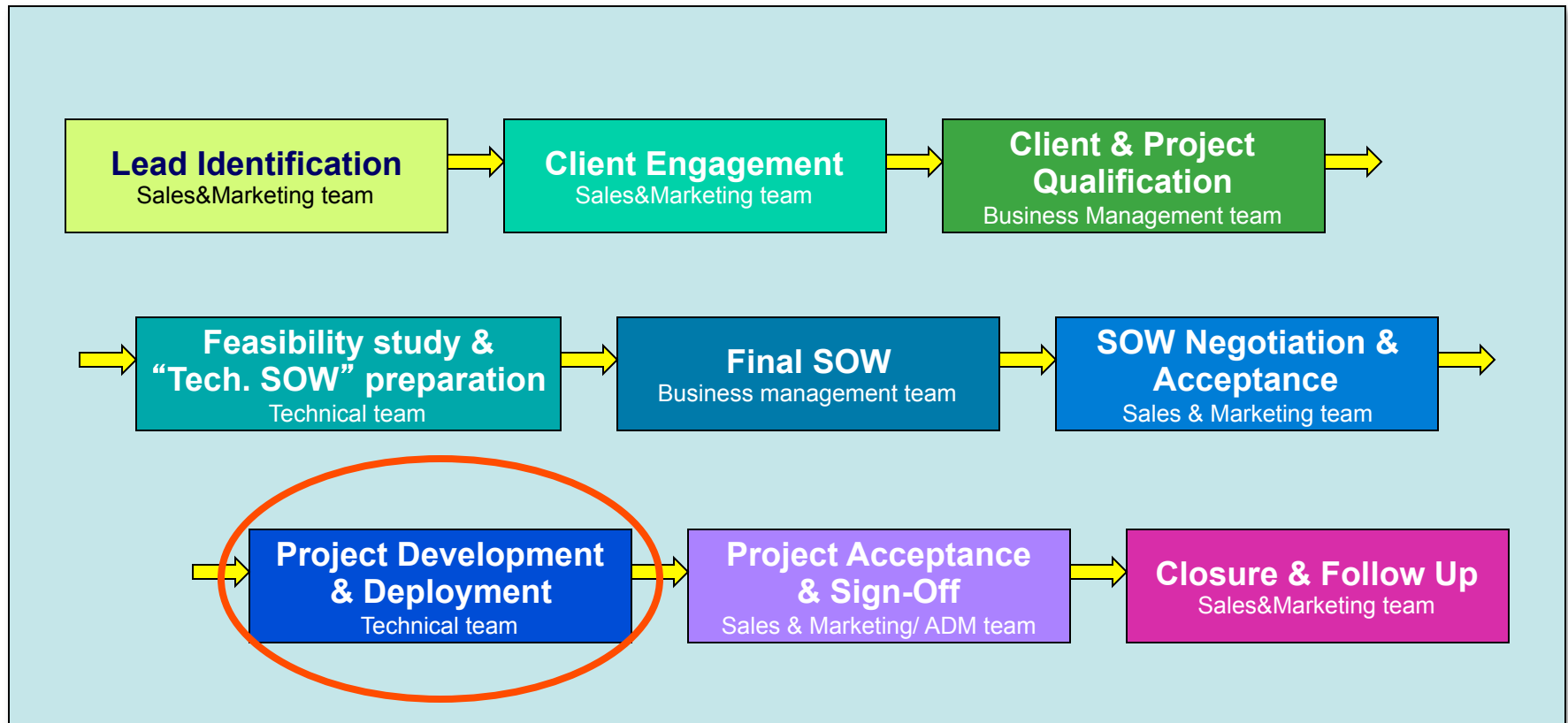
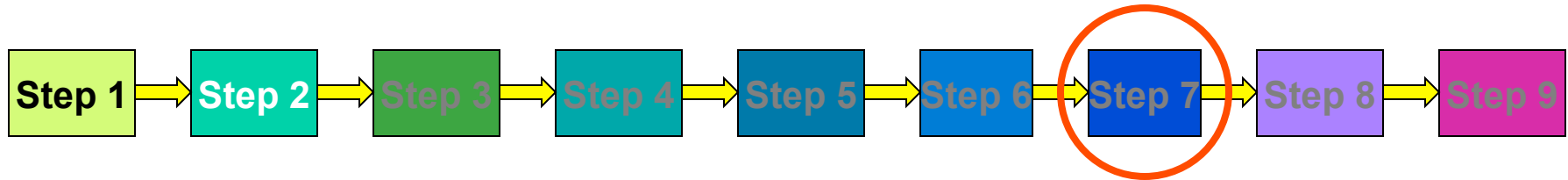


# Incremental development Model

---

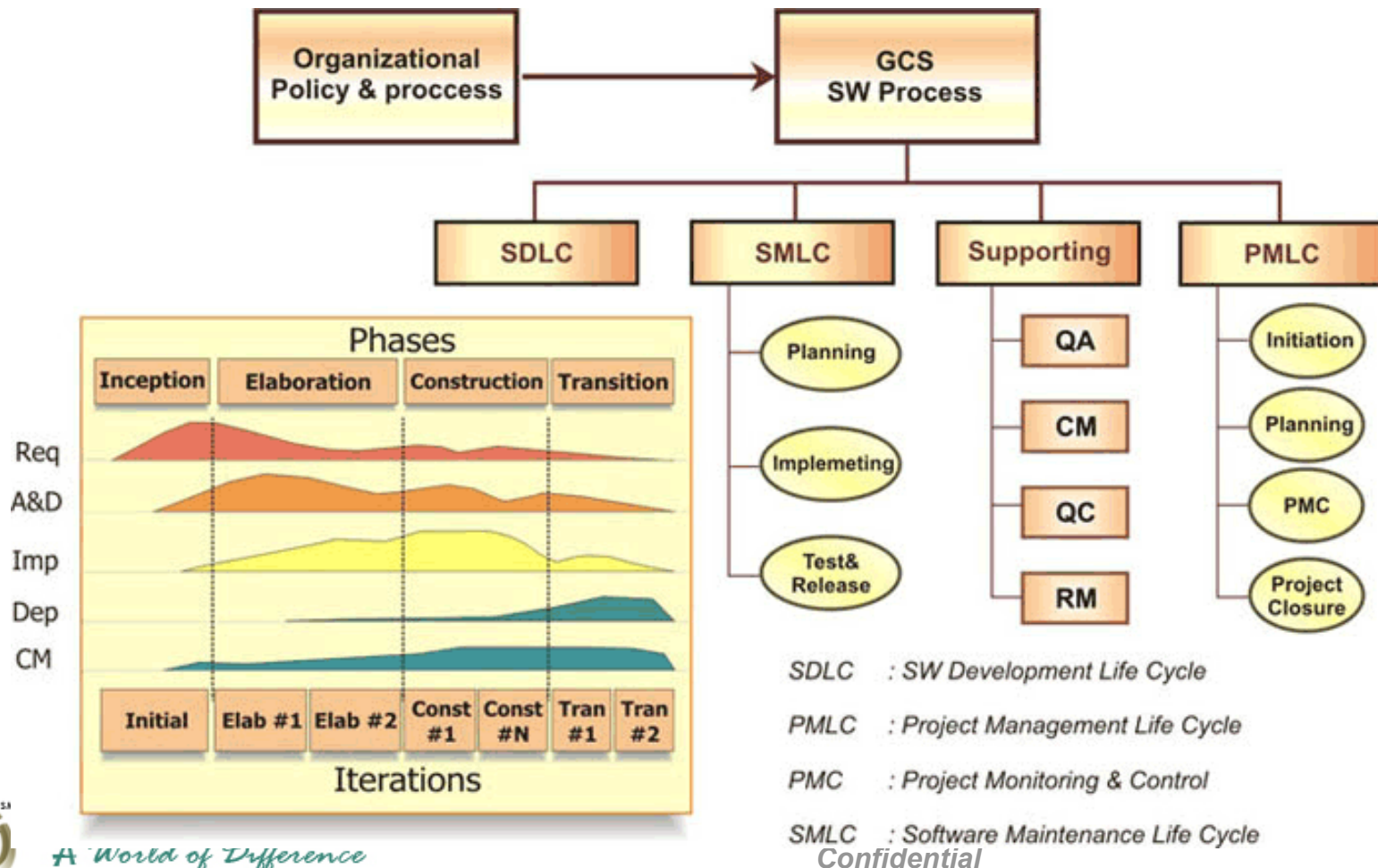
- Some models:
  - Evolutionary
  - Agile methods:
    - XP
    - Scrum

# GCS SDP in steps of the whole process



# SDP - Process structure

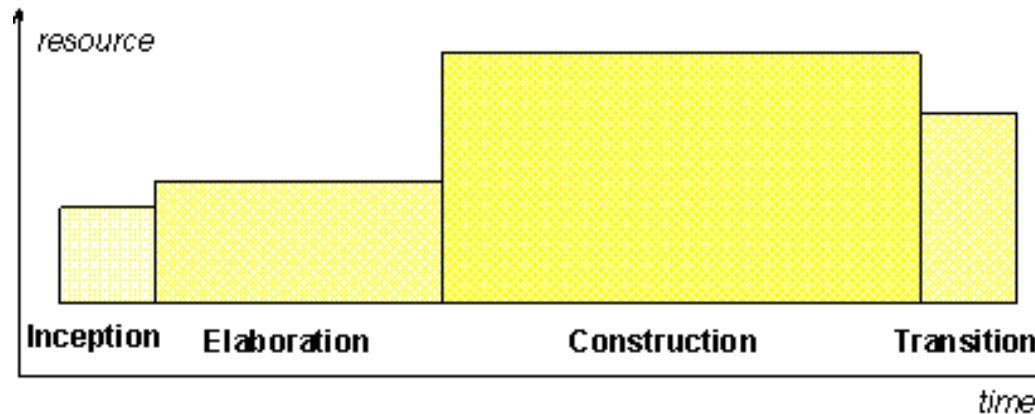
Product Quality belongs not to an **alone** process, BUT the combinations of and the result from multiple processes of a solid system





# Four phases of Development Life Cycle

Phases are to plan and track project objectives through stages of project life cycle.



<b>Inception</b>	<b>Project initial activities. Input is SOW</b>
<b>Elaboration</b>	<b>Design &amp; architecture, Project plans, milestones</b>
<b>Construction</b>	<b>Coding &amp; Testing, Design details</b>
<b>Transition</b>	<b>Stabilize and release the product</b>

# Supporting processes

---

<b>Training</b>	<b>Build and improve staff capabilities</b>
<b>Risk Management</b>	<b>Identify, track and mitigate risks</b>
<b>Configuration Management</b>	<b>Control project's work products</b>
<b>Peer Review</b>	<b>Remove defects from the work products</b>
<b>Measurement &amp; Analysis</b>	<b>Quantitative project management</b>
<b>Decision Analysis</b>	<b>Support for making right decisions</b>
<b>Quality Assurance</b>	<b>Monitor process compliance.</b>

# Summary of key activities in a project

---

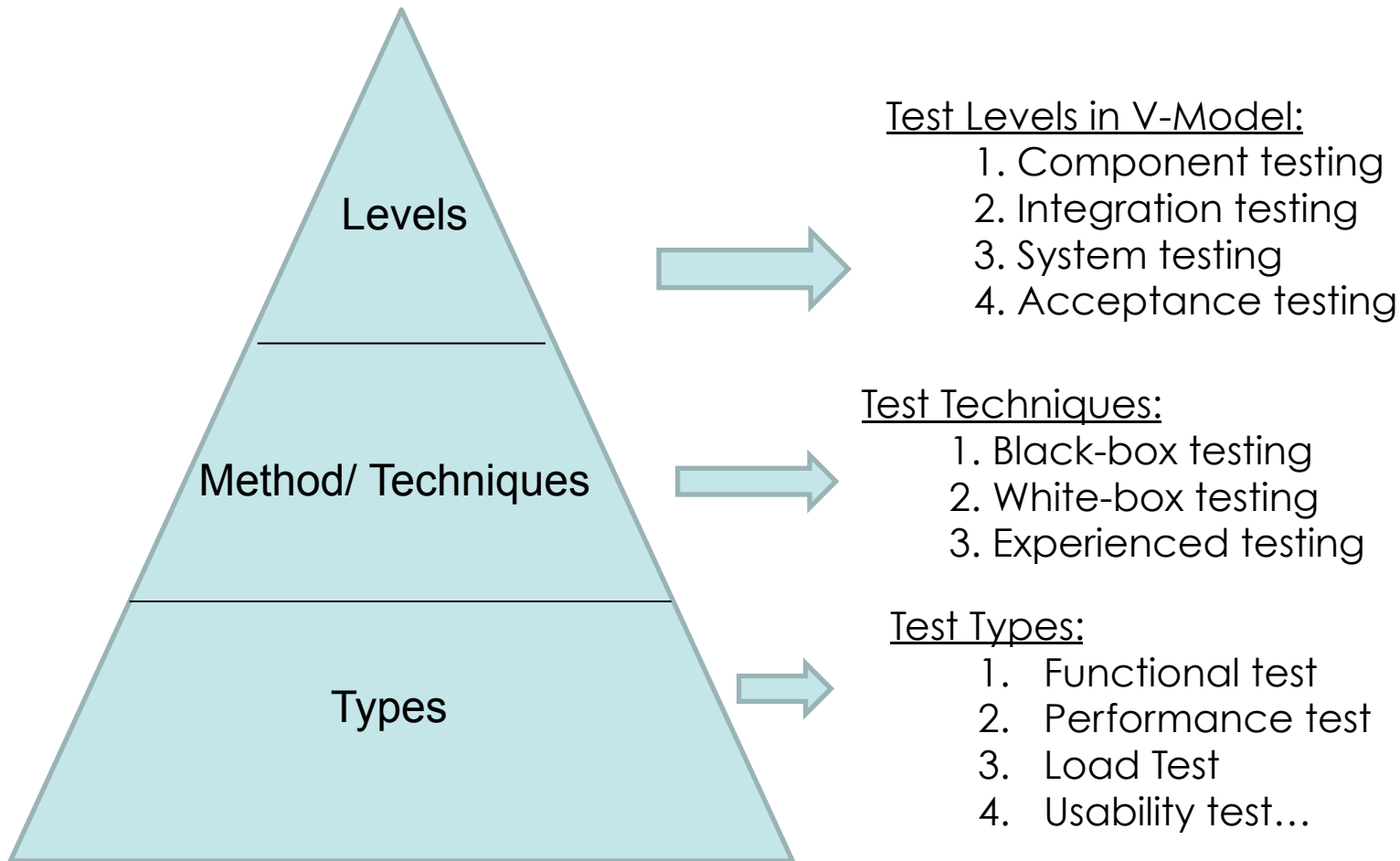
- ❑ Estimation
- ❑ Statement of Work/Proposal
- ❑ Product acceptance criteria
- ❑ Risks list
- ❑ Project repository
- ❑ Development cases
- ❑ Software development plan
- ❑ Project schedule
- ❑ Software requirement specifications
- ❑ Software architecture document
- ❑ Design models
- ❑ Change requests / change tracking list
- ❑ Review records
- ❑ Source code controlling
- ❑ Unit test cases
- ❑ QA plan/Test plan/Test cases/Bug tracking list/Test reports
- ❑ Review and audit schedule
- ❑ NC tracking list
- ❑ Build plan/build notes/release notes
- ❑ Project reports
- ❑ Project closure report

# Test Level, Test Types, Testing techniques

## Test Concepts:

- **Test Objective**
- **Quality Attribute**
- **Test Level**
- **Test Type**
- **Test Method/ Technique**
- **Test Strategy**

# Test Levels



# Test Levels

---

- ❑ A group of test activities that are organized and managed together
- ❑ A test level is linked to the responsibilities in a project
- ❑ A common type of V model uses four test levels
  - Component testing
  - Integration testing
  - System testing
  - Acceptance testing

# Test Level – System test

---

- ❑ Test Objective: Verifies system as a whole meets the specified requirements
- ❑ Test objects: The system as a whole
- ❑ Test Environment: Hardware and software
- ❑ Test environment should be separated (not to damage customer's operational system if any failure OR conflict with developers' environment AND to have full control of test environment for easy configuration and determination root cause).
- ❑ Difficulties:
  - Unclear system requirements
  - Missed decisions from earlier phases will cause trouble in determining failure and root cause. Bugs now cost too much because it's late to change design.

## Test Level – Acceptance test

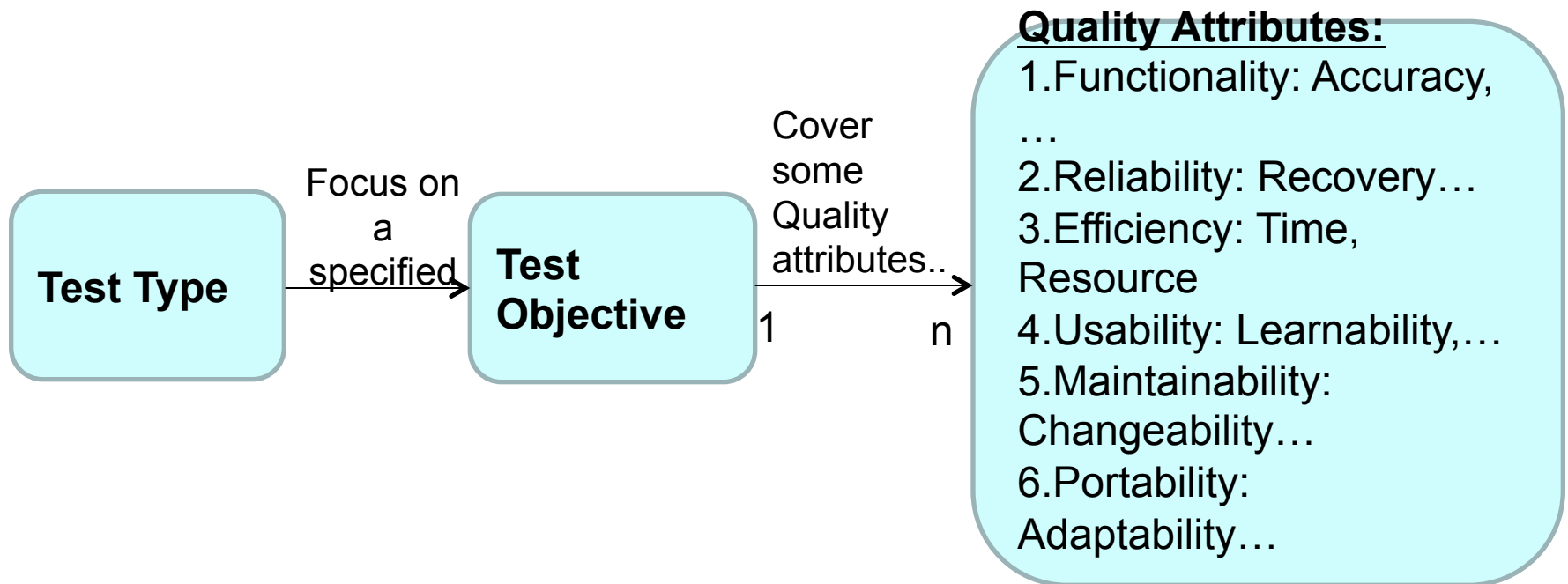
---

- ❑ Test Objective: Customer checks if the system meets their requirements as specified in the contract and from the user's point of view. To provide confidence in the system.
- ❑ Test Object: A component, new functional enhancement or a whole system
- ❑ Testing according to the contract:
  - Operational Testing: Testing of backup/restore cycles, disaster recovery, user management, maintenance tasks and security vulnerabilities.
  - Compliance Testing
  - Field testing: Testing implemented by preselected customers/users who represent the market for software under test.
    - Alpha Testing: carried out at the developer's site
    - Beta Testing: carried out at the customer's site
- ❑ Testing for user acceptance: Customer and user can be in different groups (of organizations or individuals) with their own interests and requirements of the system.
- ❑ Difficulty: Bugs found in this test cost too much (too late to fix). Solution: Present prototypes to the customer early.



# Test Type

- ❑ Test Type – Target of Testing
- ❑ A group of test activities aimed at testing a component or system focused on specific test objective.



# Test Types – Target of Testing

---

- The following types of testing can be:
  - Functional testing
  - Non-functional testing
  - Structure testing
  - Confirmation & regression testing

# Test Types – Target of Testing

---

## ❑ **Functional Testing:** *“What the system does?”*

Functions are described in:

- Requirement specification
- Use cases
- Functional requirement
- Maybe undocumented

## ❑ **Non-functional test:** *“How well the system works?”*

- Performance testing
- Load testing
- Stress testing
- Usability Testing
- Interoperability Testing
- Reliability Testing
- Scalability, Stability

- Portability Testing

### Test Concepts:

- Performance testing
- Load testing
- Stress testing
- Usability Testing
- Interoperability Testing
- Reliability Testing
- Portability Testing



# Test Types – Target of Testing

---

## ❑ Structure testing

- Performed at all test levels
- Best used after specification-based techniques, in order to help measure the thoroughness of testing through assessment of coverage of a type of structure
- Test Coverage: expressed as a percentage of the items being covered.
  - If coverage is not 100%, then more tests may be designed to test those items that were missed, and therefore increase coverage.

# Test Types – Target of Testing

---

- ❑ **Testing related to changes** (confirmation test or retest)
  - Confirmation Test
    - When a defect is detected and fixed
    - To confirm that the original defect has been successfully removed.
  - Regression Test
    - The repeated testing of an already tested program, after modification, to discover any defects introduced or uncovered as a result of the changes
    - These defects may be either
      - In software being tested
      - In another related or unrelated software component
    - The extent of regression testing is based on the risk of not finding defects in software that was working previously.

# Maintenance Testing

---

- ❑ Testing **the changes** to an operational system or **the impact** of a changed environment to an operational system.
- ❑ Maintenance testing is done on an existing operational system and is triggered by modifications, migration or retirement of the software or system.
- ❑ The Changes/Modification includes:
  - Planned enhancement changes (e.g release-based)
  - Corrective and emergency changes
  - Changes of environment
    - Planned operation system
    - Database upgrades
    - Patches to newly exposed
    - Discovered vulnerability of the operation system
- ❑ **Note: Maintenance testing is different from Maintainability Testing**



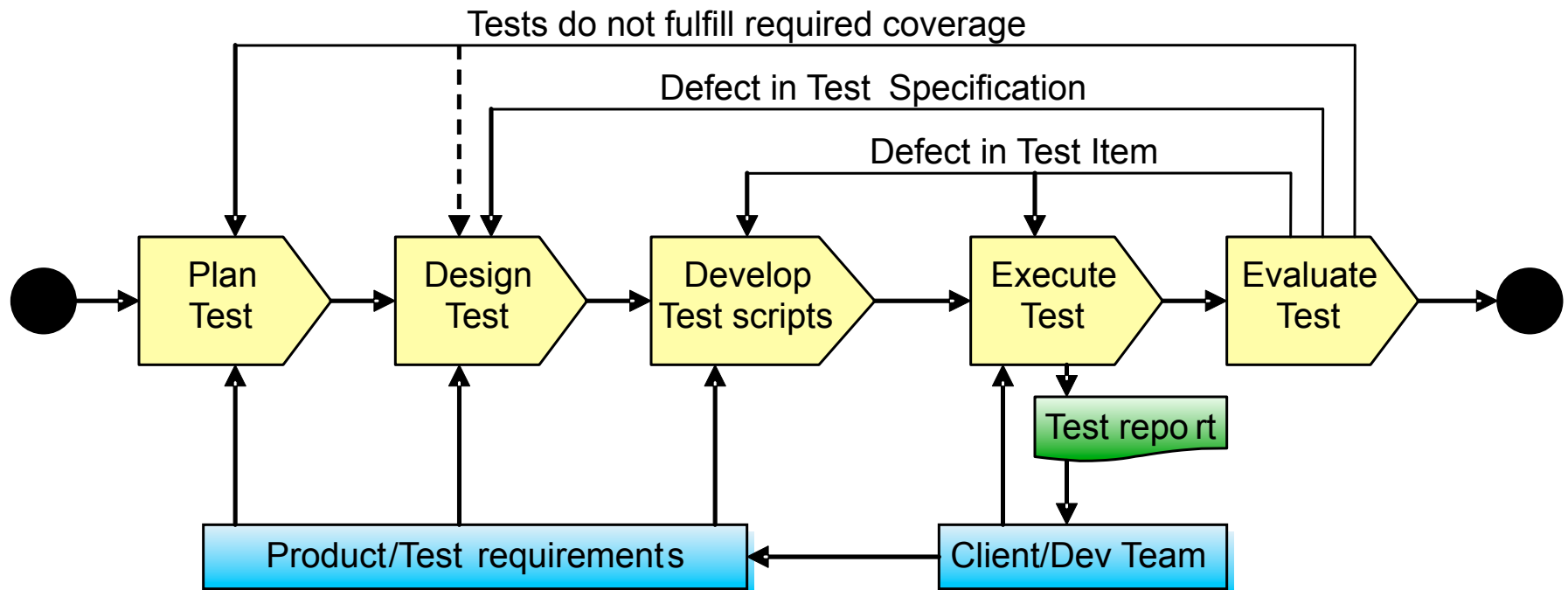
# Testing Services at Global CyberSoft

---

## GCS Testing Process

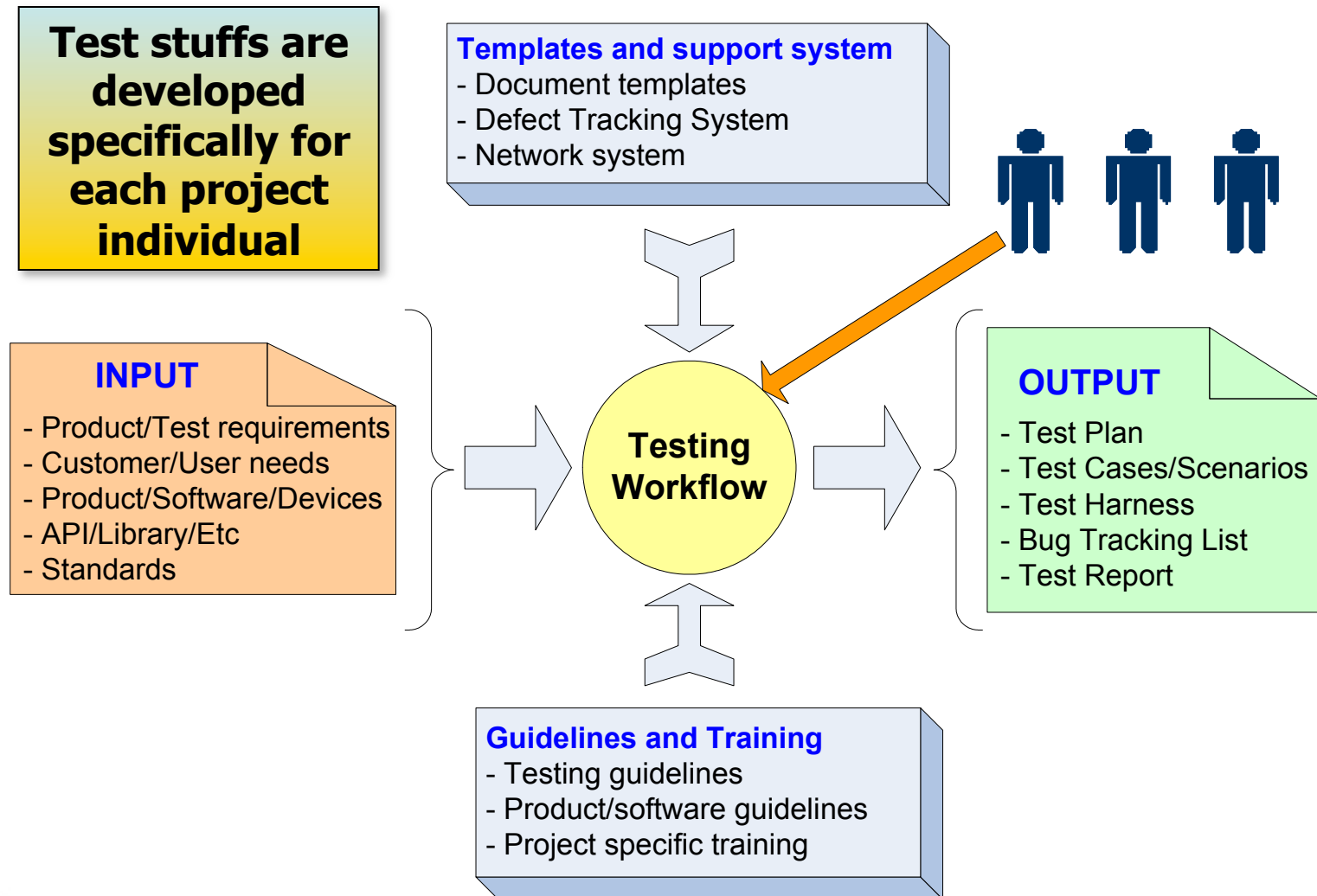
# Testing process workflow

**Testing process is not immutable, but is customized from project to project based on project specific factors and needs**





# Testing process components

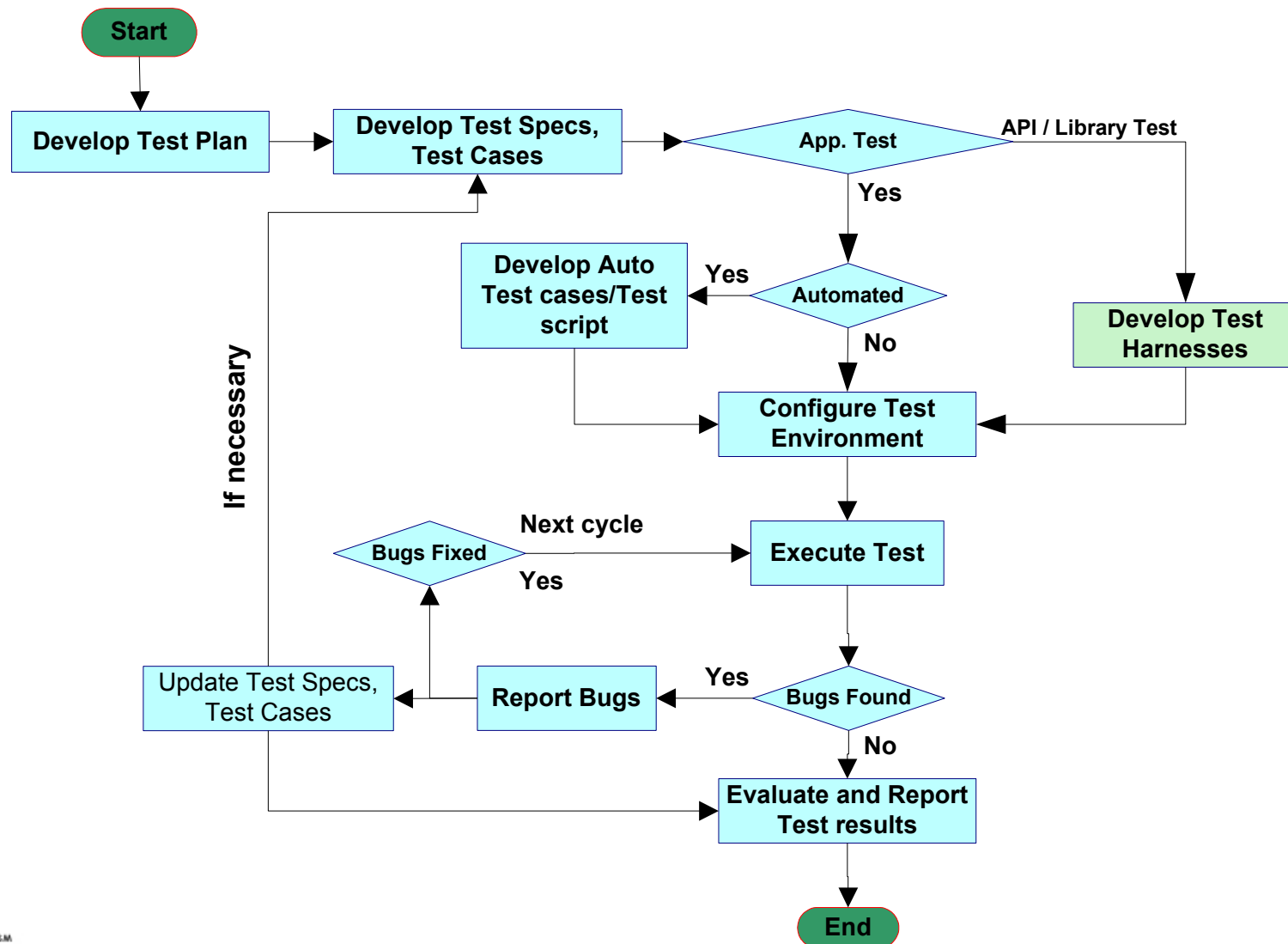


# Testing output documents

**Depending on project and customer needs, list of output documents varies from project to project**

<b>Test Plan</b>	What to test, Testing strategies, Testing environment, Resources and Schedule
<b>Test Cases/Scenarios</b>	Detailed steps on how to verify a required case, Maintain traceability between requirements and tests
<b>Test Harness</b>	A set of files used to test a Unit/API/Library, Included test driver code, test data, source code and/or libraries
<b>Bug Tracking List</b>	To track status of all bugs, bug classifications, analysis results of bugs, and bug trends
<b>Test Evaluation Report</b>	To track status of test cases, status if testing work, measurement and metrics related to executing tests

# Testing activities workflow



# Testing activities

Activity	Resp	Reviewed By	Approved By	Artifact
Develop Test Plan	Test Lead	PM, Test Lead, Tester	PM	Test Plan
Develop Test Cases	Tester	PM, Test Lead	PM	Test Cases Review record
Develop Test Harnesses if required	Tester	Test Lead	PM	Test Harnesses stuffs
Determine Auto Test Cases	Test Lead	PM	PM	(Auto) Test Cases
Develop Automated Test	Test Lead	PM	PM	Test scripts
Configure Test Environment using criteria	Tester	Test Lead	Test Lead	Test environment
Execute Test	Tester	Test Lead	Test Lead	Test records
Select regression test cases using criteria	Test Lead	PM	PM	Test Cases
Report Bugs	Tester	Test Lead	Test Lead	Bugs tracking list
Update Test Cases	Tester	Test Lead	Test Lead	Test Cases
Evaluate and Report Test Results	Test Lead	PM, Test Lead	Test Lead	Test Report, Bugs tracking list

# Testing Services at Global CyberSoft

---

## GCS Testing Templates

- ❑ Test plan
- ❑ Test case master list
- ❑ Detail test case
- ❑ Test report

# Test plan (1)

There are 7 sections in the current template

## 1. Introduction

## 2. Target Test Areas

## 3. Test Specifications

## 4. Milestones


## 5. Test Cycle Entry and Exit Criteria


## 6. Environmental Needs

## 7. Risks

TEST PLAN FOR <PROJECT/PRODUCT> RELEASE 1.0	
1	Introduction
1.1	Purpose
1.2	Scope
1.3	Document Terminology and Acronyms
1.4	References
2	Target Test Areas
3	Test Specifications
3.1	Features and Functions to Test
3.2	Features and Functions not to test
4	Responsibilities
5	Milestones
6	Test Cycle Entry and Exit Criteria
6.1	Entry Criteria
6.2	Exit and Resume Criteria
6.2.1	Test Completed
6.2.2	Test Resume
6.3	Abnormal Termination
7	Environmental Needs
7.1	Hardware and Software
7.2	Productivity and Support Tools
7.3	Test Environment Configuration
7.4	Test Environment Validation
8	Risks
9	Staffing and Training Needs



- | <div>  <h1>TEST SPEC MASTER LIST AND EXECUTION REPORT</h1> </div> |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
|--|------------------|--------------|------------------|-----------------|-----------|---------------|--------------------------|------|--------|----------------|------|--------|-----------------|--------|--------|------|
| Test Spec  | Case Description | Test Case ID | Priority (H/M/L) | Produce Version | Test Type | Result (P, F) | Estimated Execution Time |      |        | Execution Time |      |        | <BUILD NAME>    |        |        | Note |
|  |                  |              |                  |                 |           |               | Day                      | Hour | Minute | Day            | Hour | Minute | Executed (P, D) | Bug ID | Tester |      |
|  |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
|  |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
|  |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
|  |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
|  |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
| Total:   |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
|  |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
|  |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
|  |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
|  |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
|  |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |
| Total:   |                  |              |                  |                 |           |               |                          |      |        |                |      |        |                 |        |        |      |




Function 01

Req Brief Descriptions


[Return to](#)

Test Case ID	Action	Expected Result
FT-LOGIN-01	Test case <u>login</u> 01	

 <b>Function 01</b> Req Brief Descriptions		<a href="#">Return to Summary Sheet</a>
Test Case ID	Action	Expected Result
FT-LOGIN-01	Test case <a href="#">login</a> 01	
	*	
	1	
	2	
	3	
	4	
	5	
	6	
	7	
	8	
FT-LOGIN-02	Test case <a href="#">login</a> n	
	*	
	1	
	2	
	3	
	4	
	5	
	6	
	7	
	8	

# Report Templates

- ❑ Test Execute report
- ❑ TL-SDL-302 - Bugs Tracking List or Using Issue tracking tool



Global CyberSoft

TEST EXECUTION REPORT

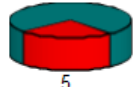
Information

Project name	
PM	
Reported By	
Date	

Test Result Summary

#Test Cases	58	
#Executed	20	34.5%
#Passed	15	75.0%
#Failed	5	25.0%

Summary



Test Case ID	Test Cases Description	Priority	Result	Tester	Build 060619-05		Build 060614-00		Build 060614-00		Note
					Status	Date	Status	Date	Status	Date	
Function 01											
FT-LOGIN-01	Test case login 01										
FT-LOGIN-02	Test case login 02										
FT-LOGIN-03	Test case login 03		P		P		F		F		
FT-LOGIN-04	Test case login 04		P						P		
FT-LOGIN-05	Test case login 05		P		I		P				
FT-LOGIN-06	Test case login 06										
FT-LOGIN-07	Test case login 07										
FT-LOGIN-08	Test case login 08										
FT-LOGIN-09	Test case login 09										
Function 02											
FT-LOGIN-01	Test case login 01										
FT-LOGIN-02	Test case login 02										
FT-LOGIN-08	Test case login 08										
FT-LOGIN-09	Test case login 09										

GENERAL SUMMARY

Total	0	
Status	#	%
Open	0	0.0%
Re-Open	0	0.0%
Accepted	0	0.0%
Not-a-bug	0	0.0%
Deferred	0	0.0%
Duplicated	0	0.0%
Fixed	0	0.0%
Passed	0	0.0%
Failed	0	0.0%
Closed	0	0.0%
Severity	#	%
Critical	0	0.0%
Major	0	0.0%
Minor	0	0.0%
Enhancement	0	0.0%



# Questions

---



# Appendix

---

- ❑ **Error:** a human action that produces an incorrect result
  - ❑ **Fault:** a manifestation of an error in software
    - also known as a defect or bug
    - if executed, a fault may cause a failure
  - ❑ **Failure:** deviation of the software from its expected delivery or service
  - ❑ **Defect masking** is that a fault is hidden by one or more other faults in different parts of the application
  - ❑ ...
- 
- ❑ Refer to: ISTQB Glossary

# Reference

---

- ❑ Refer to:
  - ISTQB Foundations of software testing – Rex Black
  - ISTQB Glossary
  - “GCS-T101 - GCS Testing Concepts & Process” – Toan Ngo

# Appendix: Course detail form

<b>Author</b>	Son Pham	<b>Duration</b>	4 hours
<b>Category</b>	GCS Testing Foundation	<b>Type</b>	Theory

<b>Examination</b>	N/A
<b>Intended Audience</b>	Any QC
<b>Pre-requisites</b>	N/A
<b>Completion criteria for the course</b>	Attendee must join at least 90% course length
<b>Criteria for granting training waivers</b>	N/A

# Thank you

---

## THANK YOU

Inquires regarding the above may be directed to:  
**Someone, Title**, email [@globalcybersoft.com](mailto:@globalcybersoft.com)