# 02 Software Testing Life Cycle

PowerPoint by Wanida Khamprapai

The software development lifecycle (SDLC) is the cost-effective and time-efficient process that development teams use to design and build high-quality software.

```
se if (i==2)
 var atpos=inputs[i].indexOf("@");
  var dotpos=inputs[i].lastIndexOf(".");
  if (atpos<1 || dotpos<atpos+2 ||
   document.getElementById('errEmail').
   else
   document.getElementById(div).innerHTML
```

# **Software** development lifecycle

The goal of SDLC is to minimize project risks through forward planning so that software meets customer expectations during production and beyond. This methodology outlines a series of steps that divide the software development process into tasks you can assign, complete, and measure.

Information from https://aws.amazon.com/what-is/sdlc/

# The Importance of SDLC

Software development can be challenging to manage due to changing requirements, technology upgrades, and cross-functional collaboration.

**Some benefits of SDLC:**

- Increased visibility of the development process for all stakeholders involved
- Efficient estimation, planning, and scheduling
- Improved risk management and cost estimation
- Systematic software delivery and better customer satisfaction

Information from https://aws.amazon.com/what-is/sdlc/

# Software development lifecycle

Software lifecycle models describe phases of the software cycle and the order in which those phases are executed. However, every model uses the following phases.

- Planning
- Design
- Development/Coding
- Testing
- Deployment
- Maintenance

## SDLC models

SDLC model conceptually presents SDLC in an organized fashion to help organizations implement it. Different models arrange the SDLC phases in varying chronological order to optimize the development cycle.

- Waterfall
- Iterative
- Spiral
- Agile

# Concept and Key Terminologies

## Software

### IEEE std 610.12-1990

Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.

### ISO/IEC 2382:2015

All or part of the programs, procedures, rules, and associated documentation of an information processing system

### Sommerville, 2011

Computer programs and associated documentation. Software products may be developed for a  particular customer or may be developed for a general market.

# Concept and Key Terminologies

## Work products

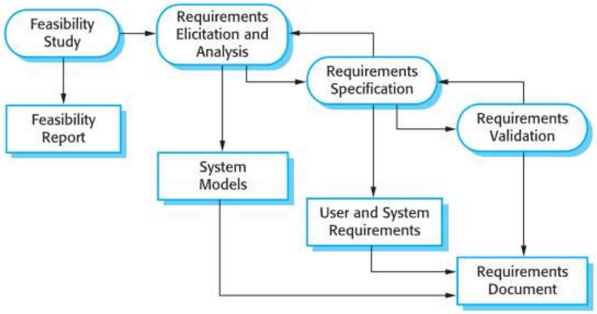
At the end of each development phase, some documents are generated such as the requirement specification document at the end of the requirements phase.

## Function

- A defined objective or characteristic action of a system or component. For example, a system may have inventory control as its primary function.
- A software module that performs a specific action, is invoked by the appearance of its name in an expression, may receive input values, and returns a single value.

## Transaction

- In software engineering, a data element, control element, signal, event, or change of state that causes, triggers, or initiates an action or sequence of actions.
- A transaction is a set of actions related to each other in some way. Transaction is deemed complete when all actions have executed successfully.



Feasibility Study

Requirements Elicitation and Analysis

Requirements Specification

Requirements Validation

Feasibility Report

System Models

User and System Requirements

Requirements Document

The requirements process (Sommerville, 2011)

# Concept and Key Terminologies

## Testing

An activity in which a system or component is executed under specified conditions, the results are observed or recorded, and an evaluation is made of some aspect of the system or component.

## Defect/bug

- (IEEE std 610.12-1990) A flaw in a component or system that can cause the component or system to fail to perform its required function or it wrongly performs the required function, or it produces the wrong result is called a defect.
- (ISTQB) An imperfection or deficiency in a work product where it does not meet its requirements or specifications.

A **mistake/error** made by a human being (any of the team members) produces a **defect/bug** in the design/ code/ document and results in  failure when such a system is executed.

# Concept and Key Terminologies

## Testing Types

Testing types depend on requirement types and there are two major types of requirements.

### Functional testing

Testing performed to evaluate if a component or system satisfies functional requirements. The Objective of functional testing is to verify that each functionality and feature of the software application/system operates in accordance with the functional requirement specifications. To check that it delivers what is expected. It has only one of the two possible results –met or not met.

### Non-functional testing

Testing performed to evaluate that a component or system complies with non-functional requirements. This comprises qualitative aspects of the system such as security (not allowing unauthorized persons to access the system or data), performance (respond very quickly to customer requests), usability (easy to use with good look & feel)

# Concept and Key Terminologies

### Positive testing (Tests-to-pass)

Check that the software behaves normally as expected with normal and legal input data.

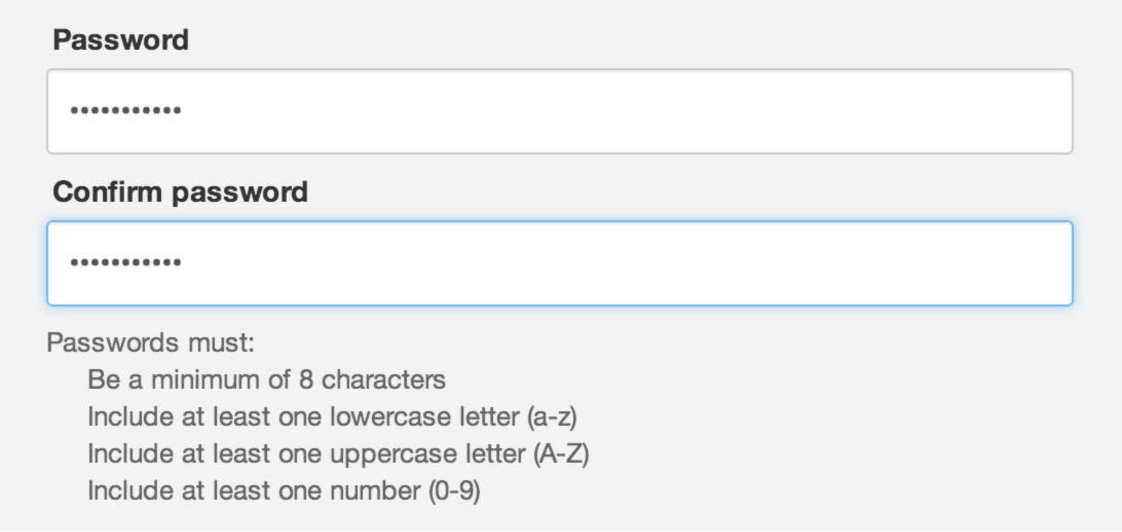
### Negative testing (Tests-to-fail)

Check that the system does not do things that it is not supposed to do when invalid or illegal data is provided. Also check that the system responds with required error messages for such abnormal or illegal data inputs.

| Aspect | Positive Testing | Negative Testing |
|---|---|---|
| Objective | To prove that application functionality is as expected | To try and break the system. |
| To check | that the product does what it is supposed to do | that product does not fail or does not do what it is not supposed to do when invalid/illegal input is given, or unexpected action is taken |
| Test cases | Mostly derived from requirement specifications. The conditions are known. | Not all tests are derived from requirement specifications. These are either derived based on techniques, or some unknown conditions for the product. |
| Coverage | If all requirements and test conditions are covered, we can say that we have achieved 100% coverage. | There is no limit to negative testing. We can never say that we have achieved 100% coverage for negative testing. |

# Concept and Key Terminologies

**Example:** positive testing and negative testing



**Password**

•••••••••••

**Confirm password**

•••••••••••

Passwords must:

- Be a minimum of 8 characters
- Include at least one lowercase letter (a-z)
- Include at least one uppercase letter (A-Z)
- Include at least one number (0-9)

# Concept and Key Terminologies

## Test condition

In software testing, the specification that a tester must follow for testing a software application. Test conditions are derived from a requirement, a technical specification, the code, or a business process.

## Test case

- (IEEE std 610.12-1990) A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement.
- (ISTQB) A set of preconditions, inputs, actions (where applicable), expected results and postconditions, developed based on test conditions.

## Test design

A process of creating/developing effective and efficient test cases.

## Test execution

The activity that runs a test on a component or system producing actual results.

# Concept and Key Terminologies

## Test level

A group of test activities that are organized and managed together. A test level is linked to the responsibilities of a project.

## Testing technique

The method used to perform the actual test execution, either manual or automated. There are two testing techniques applied as give below:

- **Static testing**: Testing of a component or system at specification or implementation level without execution of that software.
- **Dynamic testing**: Testing that involves the execution of the software of a component or system.

# Concept and Key Terminologies

## Test design techniques

Guidelines through which we can find most potential errors with a limited number of test cases. They help us to optimize the testing – Not too much and not too less testing.

### White-box testing

Testing based on an analysis of the internal structure of the component or system.

### Black-box testing

Testing, either functional or non-functional, without reference to the internal structure of the component or system.

### Gray-box testing

Testing that combine white-box testing and black-box testing.

# Concept and Key Terminologies

## Manual testing

A testing process that is carried out manually in order to find defects without the usage of tools or automation scripting.

## Test Automation

The use of software to perform or support test activities. The practice of automatically reviewing and validating a software product

# Concept and Key Terminologies

## Verification

Text

- (IEEE std 610.12-1990) The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.
- (ISTQB) Confirmation by examination and through provision of objective evidence that specified requirements have been fulfilled.
- The process of determining whether output of on phase of development conforms to its previous phase.
- Are we building the product, right?

## Validation

Result Correct

- (IEEE std 610.12-1990) The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements.
- (ISTQB) Confirmation by examination that a work product matches a stakeholder's needs.
- The process of determining whether a fully developed system conforms to its requirement specification document.
- Have we built the right product?

# Software testing lifecycle

**Software testing life cycle (STLC)** is a set of specific tasks completed during the testing process to ensure that software quality goals are met. The STLC process has different objectives and outcomes  at each stage, all with the goal of catching errors and optimizing test coverage. These are popular following phase in STLC.

- Requirement analysis
- Test planning
- Test case design and development
- Test environment setup
- Test execution
- Test cycle closure

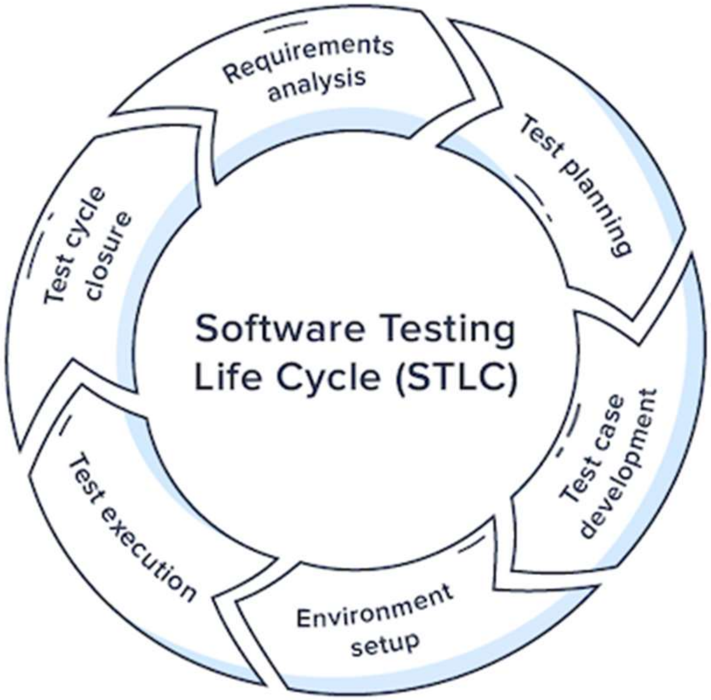


Image from https://biplus.com.vn/software-testing-life-cycle/

# Software testing lifecycle

## Requirement analysis SRD = System Requirement Document

When the SRD is ready and shared with the stakeholders, the testing team starts high level analysis concerning the software/program under Test.

| | Requirement phase testing activities | Requirement phase testing outputs |
|---|---|---|
| เรียงลำดับการทดสอบ | Determine the different sorts of tests to be run. | Report on the viability of RTM automation. (If appropriate) |
| Order the piority | Obtain information on the focus and priorities of the test. | |
| Test to check result correct with requirement | Create a requirements traceability matrix (RTM). | |
| Define user env | Determine the specifics of the test environment where the testing is to be done. | |
| Analysis Automate (IF required) | Analyze the viability of automation (if required). | |

***Note:*** Requirement Traceability Matrix (RTM) is a document that maps and traces user requirement with test cases.

Information from https://biplus.com.vn/software-testing-life-cycle/

# Software testing lifecycle

## Test planning

Test Team plans the strategy and approach. The test plan specifies the scope, objectives, types of functional and non-functional tests (both automated and human), and information about the test environments, among other aspects of the quality assurance (QA) work to be done.

| Test Planning Activities | Test Planning Output |
|---|---|
| Make a test plan or strategy document for a variety of tests | Document outlining the test strategy |
| Choose a testing tool | Estimated effort report |
| Estimate test effort | |
| Analyze role, responsibility and resource planning. | |
| Train necessary | |

Information from https://biplus.com.vn/software-testing-life-cycle/

# Software testing lifecycle

## Test case development

Develop the test cases based on scope and criteria. Test Case Development Phase entails the design, verification, and rework of test cases and test scripts.

| Test Case Development Activities | Test Case Development Output |
|---|---|
| Test Case Development Activities | Test cases/scripts |
| Create test cases and automation scripts (if applicable) | Data from tests |
| Review and baseline test cases and scripts | |
| Create test data | |

Information from https://biplus.com.vn/software-testing-life-cycle/

# Software testing lifecycle

## Test environment setup

When integrated environment is ready to validate the product. Test environment setup determines the software and hardware conditions under which a work product is tested. If the development team provides the test environment, the test team may be excluded from this activity. The test team is required to do a readiness check (smoke testing) of the specified environment.

| Test Environment Setup Activities | Test Environment Setup Output |
|---|---|
| Understand the required architecture, set up the environment, and generate a list of hardware and software requirements for the Test Environment. | Smoke Test Results |
| Configure the test environment and data | |
| Run a smoke test on the structure. | |

Test on the env that we had set

Information from https://biplus.com.vn/software-testing-life-cycle/

# Software testing lifecycle

## Test execution

Real-time validation of product and finding bugs. Executing test scripts, maintaining test scripts, and reporting bugs make up the process. When bugs are discovered, the development team is notified so that the problem can be fixed, and new testing can be done.

| Test Execution Activities | Test Execution Output |
|---|---|
| Carry out tests as planned. | RTM completed and in execution status |
| Record test results and flaws for cases that failed. | Updated test scenarios with findings |
| Link test cases to problems in RTM | Reports of errors |
| Test the corrected issues again | |
| Follow up on the issues until they are resolved. | |

Information from https://biplus.com.vn/software-testing-life-cycle/

# Software testing lifecycle

## Test cycle closure

Once testing is completed, matrix, reports, results are documented. The test cycle closure phase marks the end of the test execution and includes several tasks like test completion reporting, gathering test completion matrices, and gathering test results.

ปิดการทดสอบ

Check order by planned

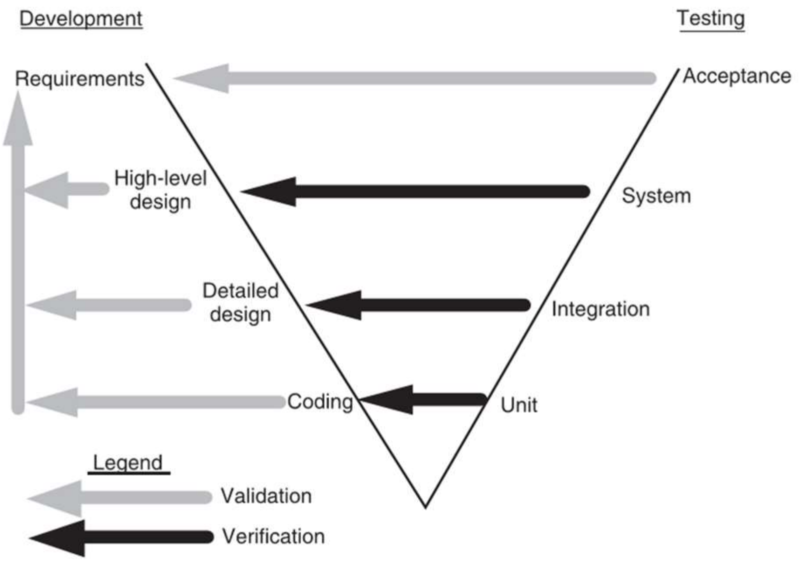
| Test Cycle Closure Activities | Test Cycle Closure Output |
|---|---|
| Analyze the cycle completion requirements in terms of time, test coverage, cost, software, critical business objectives, and quality. | The test closure report |
| Using the criteria as a guide, develop test metrics. | Tests metrics |
| Keep a record of the lessons you learned from the endeavor. | |
| Create a test closure report. | |
| Report to the client on the work product’s quality using both qualitative and quantitative data. | |

Information from https://biplus.com.vn/software-testing-life-cycle/

# Test levels

SDLC

Testing Level



Development and testing phases in the V model (Naik, 1959)

## Unit testing

Testing fn by fn. Seperate every part

Unit testing confirms that the smallest entity (e.g., a program module) can operate properly when separated from the other scripts or units.

## Integration Testing

Test data link

Modules get logically integrated and tested as one complete system test. It aims to expose any defects, bugs, or errors in the interaction between these software modules, while emphasizing on the data communication between various modules.

## System Testing

Test the whole system

The method by which a quality assurance (QA) team assesses how different parts of a program interact to form a complete, integrated system or program.

## Acceptance Testing

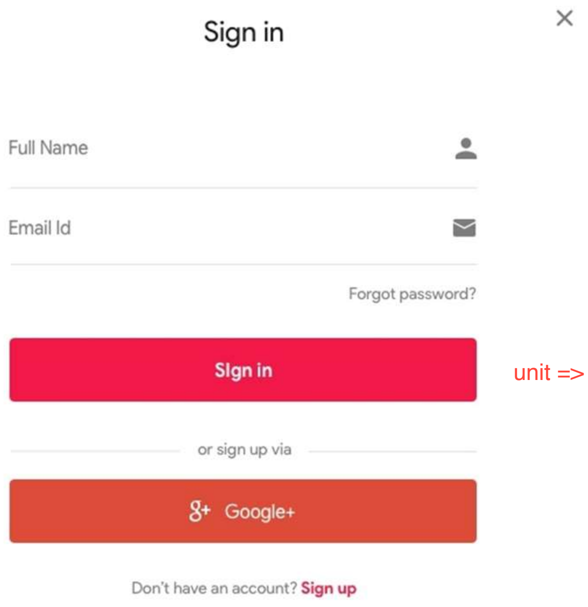
Test by customer

A quality control (QC) procedure that establishes how well a program satisfies end users' expectations.

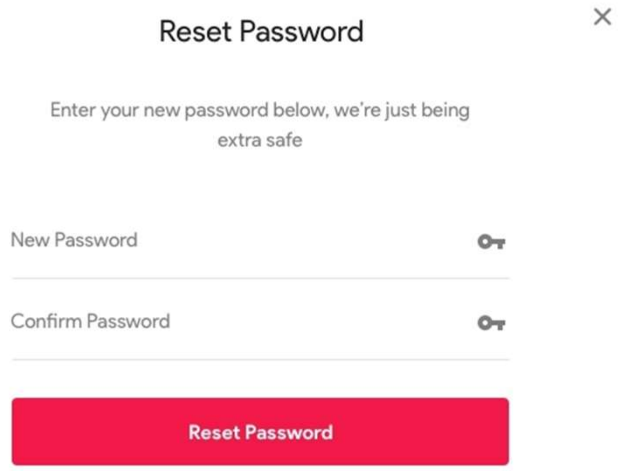
Ref.: Naik, K. 1959. Software testing and quality assurance. John Wiley & Sons.

# **Test** levels

**Example**: Test levels



unit => can signin or not



Enter your new password below, we're just being extra safe

Integration => Password is usable after rest?

# Questions & Answers