

LAB 12: IMPLEMENTATION, TESTING, DEPLOYMENT & STD

Learning outcome:

- Develop a program using object-oriented programming.
- Develop STD for testing phase.
- Develop a deployment diagram by using CASE technology.

12.1 INTRODUCTION

Implementation is the development of all parts of the system. Implementation deals with programming. Programming is often seen as the focal point for system development. In other words, systems development is using a selected programming language to write a program. It is the realization of the previous phase of analysis and design (Dennis et al., 2015).

There are four general testing stages which are unit testing, integration testing, system testing and user acceptance testing. However, the emphasis in this lab module is focused on system testing. The system testing examines how well the system meets both the functional and non-functional requirements, such as usability, documentation, performance and security (Dennis et al., 2015). A test case must be developed to execute the system testing. A test case is a specification of the inputs, execution conditions, test procedure and expected results that define a single test to be executed to achieve a particular software testing objective, for example, exercising a particular program path or verifying compliance with a specific requirement (IEEE Standards Association, 2010).

Deployment diagrams are used to represent the relationships between the components of hardware used in an information system's physical infrastructure design. For example, a deployment diagram can be used to show the communication relationships between the different nodes in the network when designing a distributed information system for a wide area network. They can also be used to model the software components and how they are distributed over the physical infrastructure of an information system (Dennis et al., 2015).

12.2 STEP BY STEP / EXAMPLE

▪ **Implementation**

Some people called as construction. Since the paradigm of software development methodology in this lab module is the object-oriented approach with UML, the implementation should therefore also use object-oriented programming. Several object-oriented programming languages that can be used such as Java, C++, Python, PHP, Ruby, Perl, Object Pascal, Objective-C, Dart, Swift, Scala, Kotlin and Common Lisp. However, Java is recommended to be used.

▪ **Development of test cases**

A test case has components that describe input, action and an expected response to determine if an application's feature is working correctly. A test case is a set of instruction about how to validate a particular test objective. It will tell us if the expected behaviour of the system is satisfied or not.

Generally, five required elements should have in a test case:

1. ID – unique identifier of a test case
2. Objective / steps / test data – what you need to do
3. Expected result – what are you supposed to get from the application/system
4. Actual result – what the actual output of the application/system provides after testing
5. Status – Pass/fail

- **Example of test case**

Test case of login module:



Figure 12-1 : Example of Login Module

Figure 12-1 shows the example of the login module GUI as stated in the SDD. Based on this login module, the test cases can be constructed as shown in Table 12-1.

Table 12-1 : Test Case for Login Module

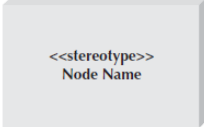
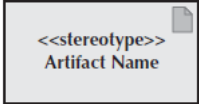
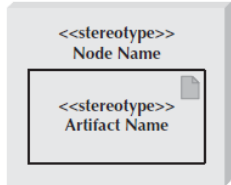
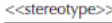
Sr. No.	Test Case ID	Objective	Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
1	Login_1.1	Check the fields available	1. Goto link : http://www.flipkart.com/ 2. On homepage click login link	No data	The login page should contain email and password field, sign up, forgot your password	As expected	Pass
2	Login_1.2	Check for buttons	1. Goto link: http://www.flipkart.com/ 2. On Homepage click login link 3. Check for login button	No data	The login page should contain login /sign in button	As expected	Pass
3	Login_1.3	Check for empty validation	1. Goto link: http://www.flipkart.com/ 2. On Homepage click login link 3. Click login button without filling email or pass	No data	Validation message should be shown for empty email and password	As expected	Pass
4	Login_1.4	Check for valid email and password	1. Goto link : http://www.flipkart.com/ 2. On Homepage click login link 3. Login with valid credentials	Valid email: kiran.hole@technologies.co.in Password: JustTest	User should be redirected to the User's account homepage	As expected	Pass
5	Login_1.5	Check validation for invalid email	1. Goto link: http://www.flipkart.com/ 2. On Homepage click login link 3. Enter invalid email/password	Invalid Email: test@co	Validation message should be shown as "email/password combination is wrong"	As expected	Pass

▪ Syntax for Deployment Diagram

Table 12-2 shows the syntax notation that should be used in order to construct the deployment diagram.

Table 12-2: Syntax and Notation for Deployment Diagram

(Source : Dennis et al., 2015)

No	Notation Name	Notation	Description
1	Node		<ul style="list-style-type: none"> Is a computational resource, e.g. a client computer, server, separate network or individual network device. Is labeled by its name. May contain a stereotype to specifically label the type of node being represented, e.g. device, client workstation, application server, mobile device, etc.
2	Artifact		<ul style="list-style-type: none"> Is a specification of a piece of software or database, e.g. database or a table or view of a database, a software component or layer Is labeled by its name. May contain a stereotype to specifically label the type of artifact, e.g. source file, database table, executable file, etc.
3	Node with deployed artifact		<ul style="list-style-type: none"> Portrays an artifact being placed on a physical node.
4	Communication path		<ul style="list-style-type: none"> Represents an association two nodes. Allows nodes to exchange messages. May contain a stereotype to specifically label the type of communication path being represented (e.g. LAN, Internet, serial, parallel)

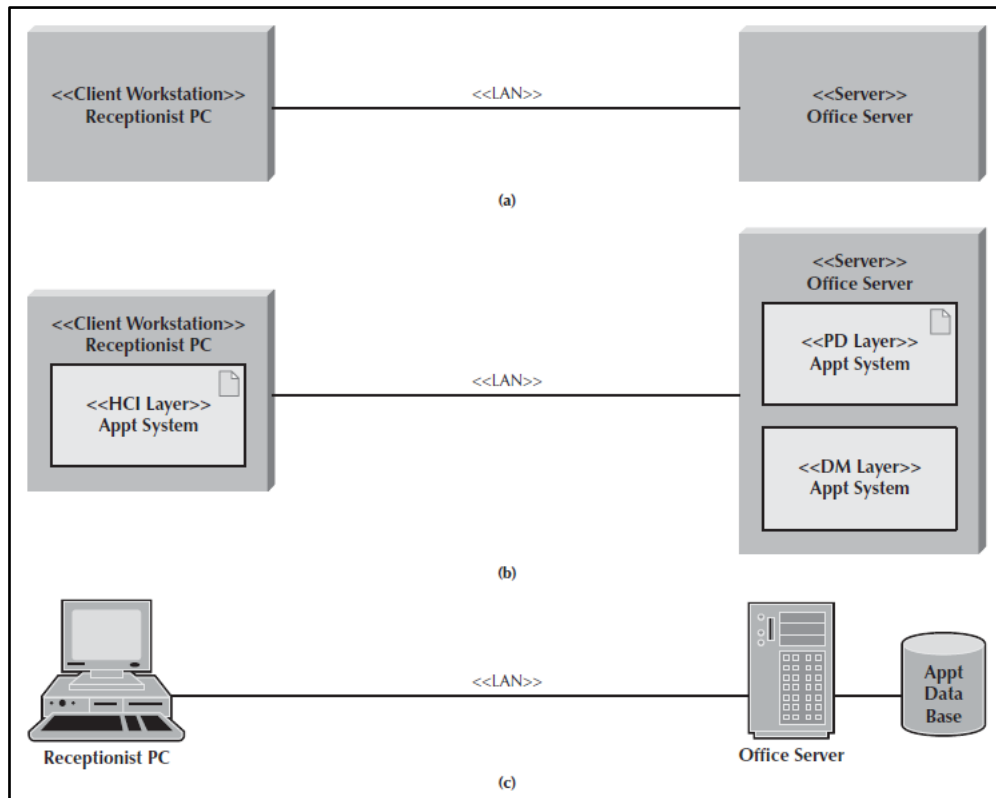


Figure 12-3 : Example of Three Version of Deployment Diagram

(source : Dennis et al., 2015)

Figure 12-3 shows three different versions of a deployment diagram. Version 'a' uses only a basic standard notation. Version 'b' introduces the idea of deploying an artefact onto a node. In this case, the artefacts represent the different layers of the appointment system. Version 'c' uses an extended notation to represent the same architecture.

Figure 12-4 shows the example of a deployment diagram for a low-level network model. The installation implemented on user site can be visualized of by modeling the deployment diagram. It is like a map for the stakeholder and easy for them to do the maintenance purpose in future.

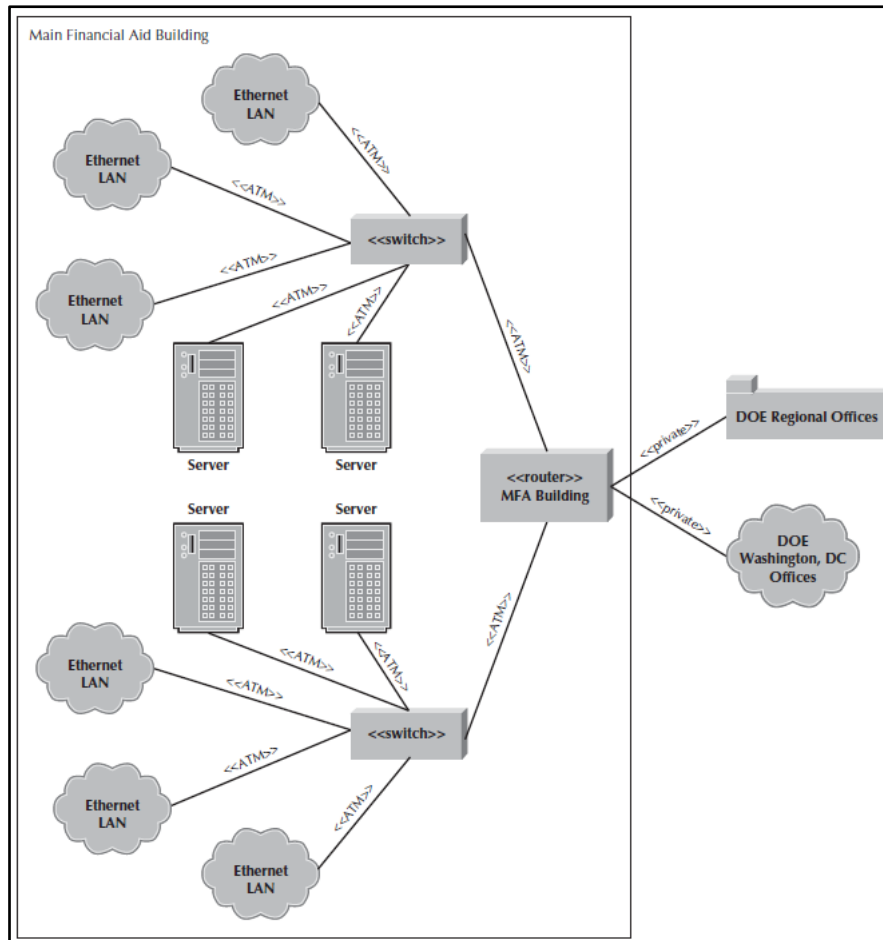


Figure 12-4 : Deployment Diagram Representation of a Low-Level Network Model
(Source : Dennis et al., 2015)

▪ **STD Framework**

Figure 12-1 shows the table of content that the students need to follow to produce the STD.

Cover
Document Approval
Table of Contents
List of Figure
List of Table
1.0 Introduction
1.1 Identification
1.2 System Overview
1.3 Document Overview
2.0 Referenced Document
2.1 Government Document
2.2 Non Government Document
3.0 Test Cases
4.0 Notes
5.0 Exhibit
<i>Note: Diagrams involved is deployment diagram.</i>

Figure 12-1 : Main Contents in STD

▪ **What Should Students Do In Lab 12?: Students' Tasks Checklist**

In lab 12, the students will work in groups and perform the following tasks:

- i. Develop a deployment diagram to visualize the physical infrastructures of your application.
- ii. Develop a program by using an object-oriented programming language.
- iii. Develop STD for system testing based on the functional requirement that you have developed.

12.3 SELF REVIEW QUESTION

1. What is the purpose of deployment diagram?

2. What is the purpose of test case?

3. State two examples of programming language that fully supported an object-oriented paradigm.

4. What is the main task performed in implementation modeling?

5. How to model the physical implementation in the system?

6. How to model a self-contained encapsulation piece of software that can be “plugged” into a system?

7. How to model the network infrastructure of the system?

- **Test Case**

Construct a test case based on the functional requirement that you have developed.

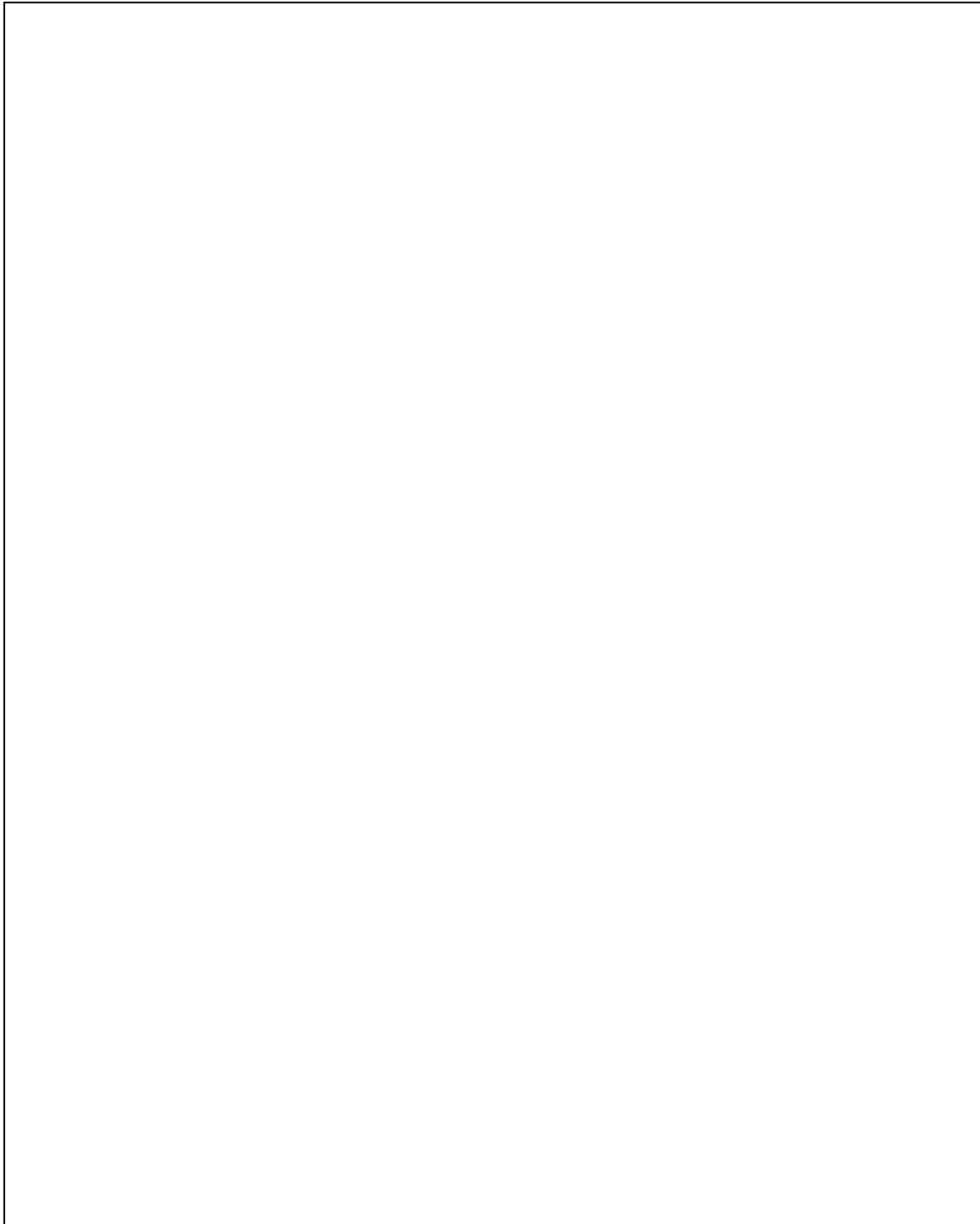


Figure 12-2 : Deployment Diagram

- **Deployment diagram**

Develop a deployment diagram to visualize the physical infrastructures.

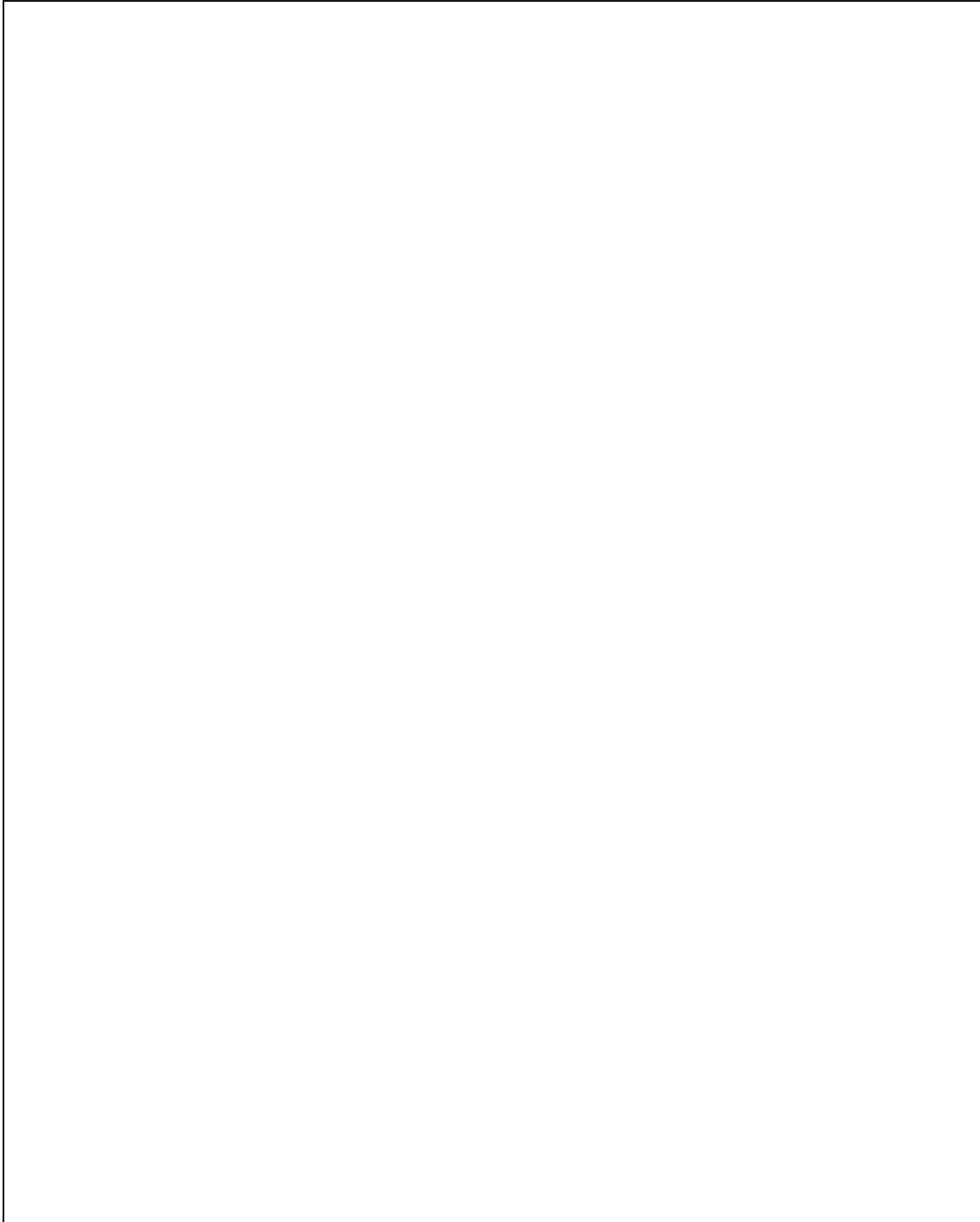


Figure 12-3 : Deployment diagram