MANIPAL INSTITUTE
OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

DEPARTMENT OF INFORMATION & COMMUNICATION TECHNOLOGY

CERTIFICATE

This is to certify that Mr./Ms. ……..…….

Reg. No.: …..…… Section …… Roll No.: …..……..…

has satisfactorily completed the lab exercises prescribed for Rational Unified Process Lab [ICT 363] of Third Year B.Tech. Degree at MIT, Manipal, in the academic year 2018-2019.

Date: .....

Signature of the Faculty

CONTENTS

Course Objectives

- To analyze, verify and validate the software requirements
- To understand the importance of design strategies in software projects
- To know the importance of software testing techniques

Course Outcomes

At the end of this course, students will be able to

- Understand importance of Software Requirements Specification
- Design UML diagram based on analysis
- Review and test the project

Evaluation plan

Total marks: 100 (Regular Lab Evaluation + End semester Lab evaluation)

Split up of 60 marks for Regular Lab Evaluation
Total of 6 regular evaluations which will be carried out in alternate weeks. Each evaluation is for 10 marks of which will have the following split up: Record/Document submission: 3 Marks Viva/Quiz: 5 Marks Execution/Completion of Task: 2 Marks Or Project Document Submission: 4 Marks Project Evaluation: 6 Marks Total Internal Marks: 6 * 10 =60 Marks
End Semester Lab Evaluation: 40 marks
UML Design: 20 Marks Project: 20 Marks Total = 40 Marks

INSTRUCTIONS TO THE STUDENTS

Pre-Lab Session Instructions

1. Students should carry the Lab Manual Book and the required stationery to every lab session
2. Be in time and follow the institution dress code
3. Must Sign in the log register provided
4. Make sure to occupy the allotted seat and answer the attendance
5. Adhere to the rules and maintain the decorum

In-Lab Session Instructions

- Follow the instructions on the allotted exercises
- Show the design to the instructors on completion of experiments
- On receiving approval from the instructor, copy the design in the Lab record
- Prescribed textbooks and class notes can be kept ready for reference if required

General Instructions for the Lab exercises and Project work

- Design should contain all the constraints specified in the question.
- The project should meet the following criteria:
  - Project should be selected with innovative ideas and high complexity
  - System Requirement Specification (SRS) should be properly documented without any ambiguity
  - Requirements specified in the SRS should be incorporated in the project
  - All the constraints specified in SRS should be met
  - Proper mapping should be there between design and implementation
  - Project implementation should be in JAVA
  - Each team should be a group of 3 students
  - Project work must be divisible among students into multiple tasks of the same complexity
  - Project report should be typed on A4 size paper in *Times New Roman*. The margins should be as follows: Left -1.25, Right -1, Top and Bottom -0.75.

    Main heading font -18, Sub heading -14, Text -11
  - Before taking the final print-out, the approval of the concerned lab faculty(s) is mandatory and suggested corrections, if any, must be incorporated
- Plagiarism (copying from others) is strictly prohibited and would invite severe penalty in evaluation.

- The exercises for each week are divided under three sets:
  - Solved exercise
  - Lab exercises - to be completed during lab hours
  - Additional Exercises - to be completed outside the lab or in the lab to enhance the skill
- In case a student misses a lab class, he/ she must ensure that the experiment is completed during the repetition class with the permission of the faculty concerned but credit will be given only to one day's experiment(s).
  - Students missing out lab on genuine reasons like conference, sport or activities assigned by the department or institute will have to take **prior permission** from the HOD to attend **additional lab** (in other batch) and complete it **before** the student goes on leave. The student could be awarded marks for the write up for that day provided he submits it during the **immediate** next lab.
  - Students who fall sick should get permission from the HOD for evaluating the lab records. However attendance will not be given for that lab.
  - Students will be evaluated only by the faculty with whom they are registered even though they carry out additional experiment in other batch.
  - Presence of the student during the lab end semester exams is mandatory even if the student assumes he has scored enough to pass the examination
  - Minimum attendance of 75% is mandatory to write the final exam.
  - If the student loses his book, he/she will have to rewrite all the lab details in the lab record.
  - Questions for lab tests and examination are not necessarily limited to the questions in the manual, but may involve some variations and / or combinations of the questions.
  - A sample note preparation is given as a model for observation.

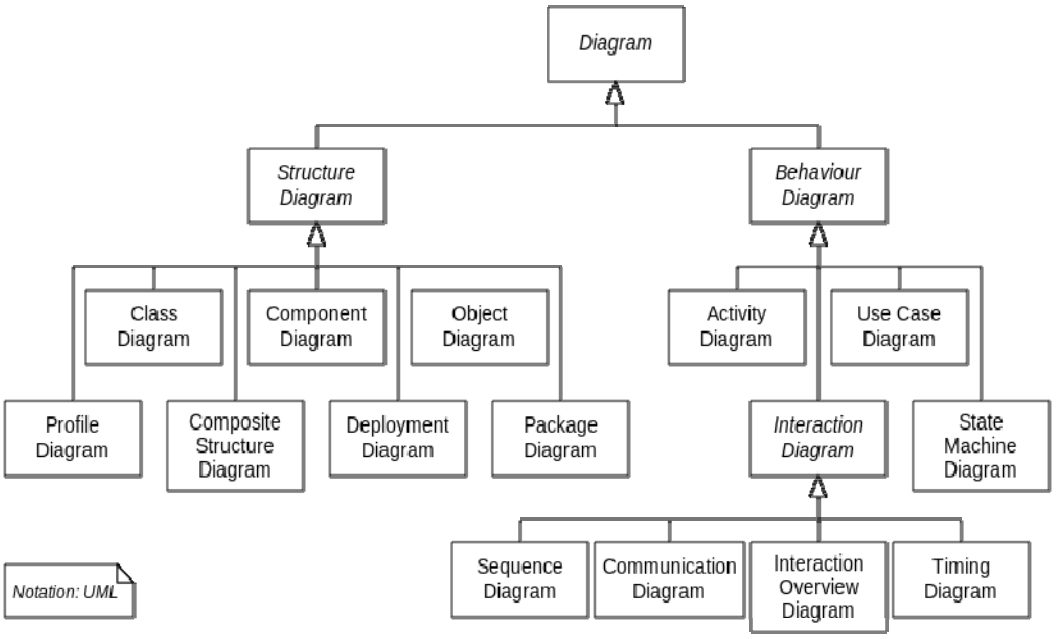
THE STUDENTS SHOULD NOT

- Bring mobile phones or any other electronic gadgets to the lab.
- Go out of the lab without permission.

INTRODUCTION TO OBJECT ORIENTED SYSTEM DEVELOPMENT

Traditional system development methodology and Object oriented methodology are the two orthogonal views of software construction. Traditional system development methodology focuses on the functions of the system. Object oriented methodology focuses on the object (which combines data and functionality). The main advantage of object oriented methodology is building self-contained modules or objects that can be easily replaced, modified and reused.

The Unified Modeling Language (UML) is a standardized general-purpose modelling language in the field of object-oriented software engineering. The Unified Modelling Language includes a set of graphic notation techniques to create visual models of object-oriented software-intensive systems. UML2 has many types of diagrams which are divided into two categories as shown in the figure below



Hierarchical structural of UML2

RATIONAL SOFTWARE ARCHITECT

Rational Software Architect is a tool that enables software architects to model and design the architecture of their applications. The content that can be created within Rational Software Architect includes all kinds of UML2.2 diagrams

Application description

Title: Phone book application.

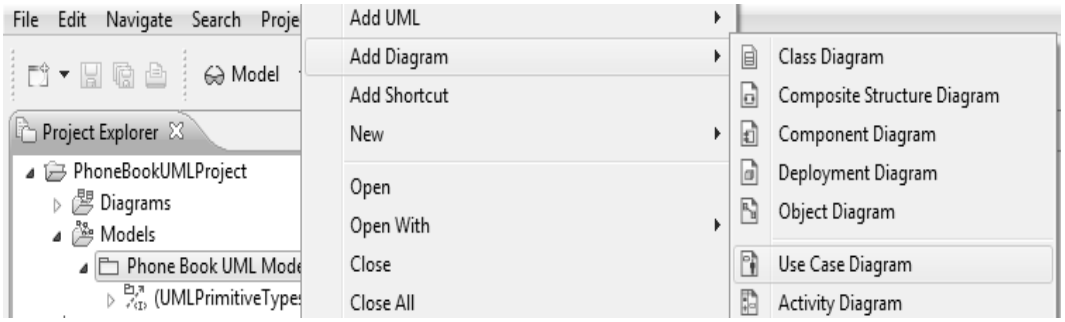
The user will be able to add entries that will be an association of a name to a phone number. And then the user will be able to search for a phone number by entering a name.

Creating the project

1. Go to File > New > Project.
2. Select the UML Project wizard. Click Next
3. Enter a Project Name. Click **Next.**
4. On the Create Model Step, choose category **General** and template **Blank Package** and enter a model name. Click **Finish.**
5. UML Project with two subdirectories, the first subdirectory is Diagrams, which will only show the UML diagrams you create organized by the type of diagram. The second subdirectory, Models, will show the diagrams and all the UML objects that you create within the model.

Creating UML Diagrams – Use Case Diagram

1. Right click the **Phone Book UML Model** and select **Add Diagram > Use Case Diagram**

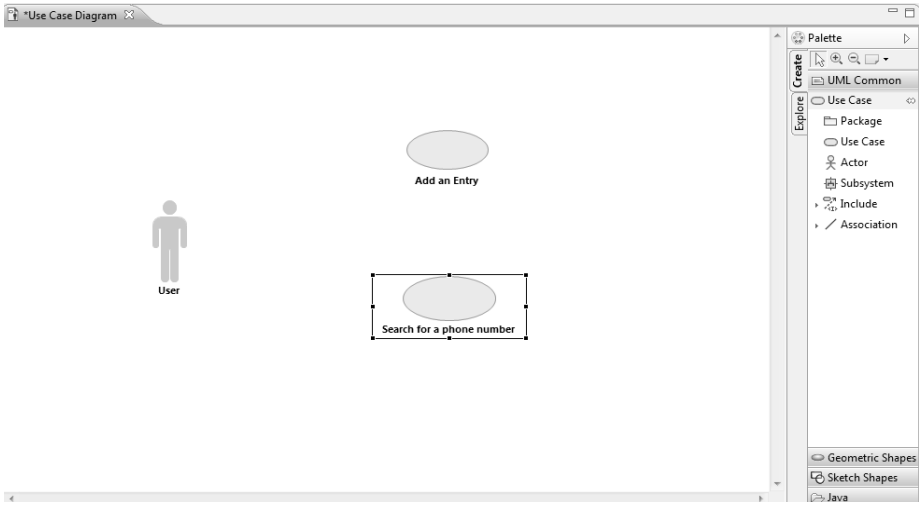


2. Enter a name for your diagram. Now you will see a pane where you will be able to add items from the palette to the diagram.

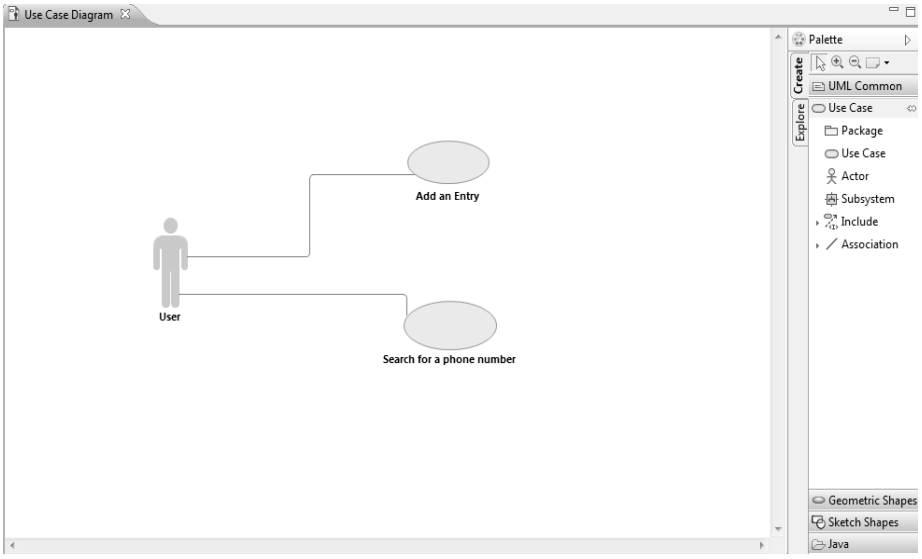
3. Click **Actor** in the palette and then click in the pane in order to add it. Name it "User".

4. Click **Use Case** in the palette and then click in the pane in order to add it. Name it "Add an Entry"

5. Click **Use Case** in the palette and then click in the pane in order to add it. Name it "Search for a phone number"



6. Click **Association** in the palette and then click and drag the association from “User” to “Add an Entry”.
7. Click **Association** in the palette and then click and drag the association from “User” to “Search a phone number”.
8. Now you have the complete use case diagram for the application. Save it.



LAB NO.: 1 **Date:**

REQUIREMENT GATHERING AND IMPLEMENTATION IN JAVA

Objectives:

In this lab, student will be able to

- Collect the system requirements
- Implement the system requirements in Java

Introduction:

A requirement is a statement about an intended product that specifies what it should do or how to do it. For requirements to be effectively implemented and measured, they must be specific, unambiguous and clear. For example, a requirement may be that a specific button must enable printing of the contents of the current screen.

Functional requirements specify the software functionality that the developers must build into the product to enable users to accomplish their tasks, thereby satisfying the business requirements, example legal/regulatory requirements. In simpler words, functional requirements state what the system must do. Non-functional requirements define the system's quality characteristics, example scalability. Requirement gathering can be done by interviews, questionnaires, direct observation, indirect observation, studying documentation, researching similar products etc.

Lab Exercises:

1. Collect the requirements for Order Processing System
2. Implement in Java for the functionalities identified in Q1. [Minimum eight functionalities]

Additional Exercise:

1. Collect the requirements for Crisis Management System
2. Collect the requirements for Online College Registration System

[OBSERVATION SPACE – LAB 1]

LAB NO.: 2 **Date:**

APPLICATION DEVELOPMENT USING JAVA SWING

Objectives:
In this lab students should be able to implement GUI application using Java Swing

Introduction to Java Swing
Swing components provide GUI environment to java. It is the collection of lightweight visual components that provide a replacement for the heavyweight AWT components.

- ✓ A swing GUI consists of two key items i.e. *components* and *containers*.
- ✓ Components are derived from the JComponent class and defined within the package javax.swing.

  The following shows the class names for Swing components: JButton, JCheckBox, JComboBox, JList, JMenu, JMenuBar, JRadioButton, JTable, JTextField, JTogglebutton, JTextArea, JToolBar, JLabel etc.
- ✓ Swing defines two types of containers:
  - Top level containers: JFrame, JApplet, JWindow and JDialog. Each Top level containers defines a set of panes, example content pane
  - Lightweight containers: JPanel
- ✓ The layout managers are used to position the components. Different types of layout managers are: FlowLayout Manager, BorderLayout Manager and GridLayout Manager
- ✓ Events specific to Swing are stored in javax.swing.event

A Simple Swing Application

```
import javax.swing.*;

class SwingDemo {

   SwingDemo()  {
```

```
      // create a new JFrame container

      JFrame jfrm = new JFrame ("A simple swing application");

      // give the frame an initial size

      jfrm.setSize(275, 100);      // Similarly, jfrm.setText("Click")

      // Terminate the program when the user closes the application
      jfrm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      // create a text based label
      JLabel jlab = new JLabel(" swing means powerful GUIs");
      jfrm.add(jlab);      // Add the label to the content pane
      jfrm.setVisible(true);   // Display the frame
}
public static void main (String args[])  {
  // create the frame on the event dispatching thread
  SwingUtilities.invokeLater (new Runnable()  {
    public void run()  {

      new SwingDemo();   }
});
} }
```

Lab Exercises

1. Write a program to create a GUI with any eight functionalities for Order Processing System using Java Swing. Use at least 8 swing components, two layout manager and necessary event handlers to show the functionalities of Order Processing System

Additional Questions:

1. Write a program to create a GUI with any six functionalities using Java Swing for the following:
   i) Crisis Management System
   ii) Online College Registration System

[OBSERVATION SPACE – LAB 2]

LAB NO.: 3 **Date:**

PROJECT SYNOPSIS & USE-CASE

Objectives:

In this lab students should be able to:

- Draw the Use-Case diagram and Write Use-Case specification using Use-Case Template
- Draw the Use-Case in Rational Software Architect

Format for the Project Synopsis

PROJECT TITLE *<Title of the project>*

<Synopsis of the project work and should be written in 3 paragraphs. The first paragraph should introduce the area of the topic and give importance of the work/topic in the present day scenario, hence leading to the objective of the project work. The second paragraph should discuss briefly the important results that were obtained and its significance. The third paragraph should discuss the important conclusion(s) of the project work. If you have used some software tools/packages or hardware/systems, indicate them in the last line. (The abstract should fit in one page only)>

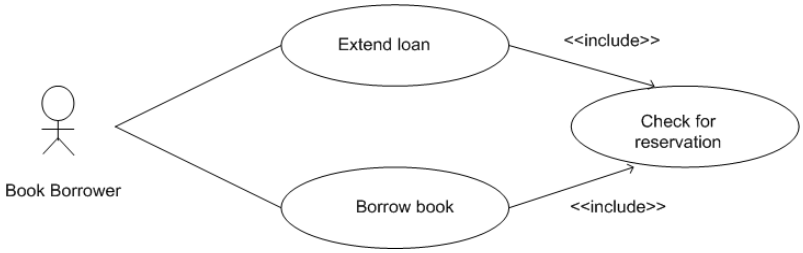
Submitted by

<name>		*<registration no>*
Instructor	:	*<place your instructor's name here>*
Course	:	*<place your course name here>*
Lab Section	:	*<place your lab section here>*
Date	:	*<place the date of submission here>*

Use-Case

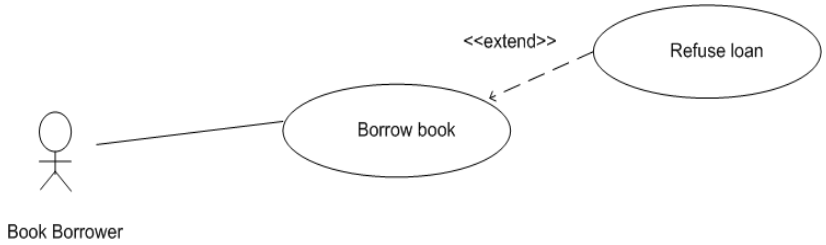
- ✓ Use-cases are descriptions of the functionality of the system form the user's perspective. Eg: Borrow Book and Return Book are the examples in library management system
- ✓ Use-case, Actors and System are the model elements of use case model/diagram
- ✓ Relationship between use-cases <<include>> and <<extends>>

✓ <<*include*>> is used for events that are in the flow of events of the source use-case



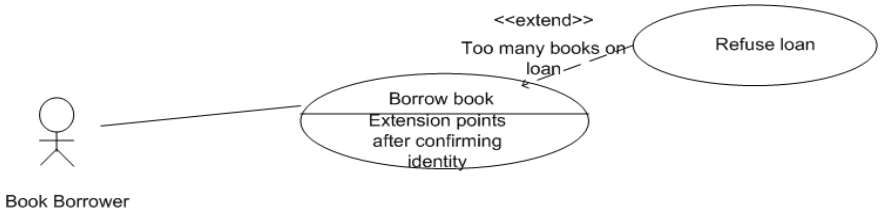
Book Borrower should check whether the *book is reserved* before *borrowing* or *extending* the book

✓ <<*extends*>> describes extra behavior which should sometimes be added depending on run time conditions

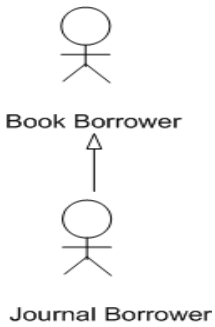


Borrow book is a normal case and not allowed to borrow books is an exceptional case

✓ <<*extends*>> *with Extension point*- An extension point identifies the point in the base use-case where the behavior of an extension use case can be inserted.



- ✓ Actors/Use-cases can be related by generalization



Every *Journal Borrower* is a *Book Borrower*

Example for Use-Case Template

Case Study: *Unauthenticated users can sign-up by providing the credentials***.**

Primary Actor	**Use Cases**
Unauthenticated User	Sign-up

Actors:	**Primary Actor**: Unauthenticated User **Supporting Actor**: Database Manager
Description:	This is used for authentication purpose, done by username and password. First the user have to register by providing unique username and strong password and some other information (like email, payment etc)
Trigger:	Submitting the information
Pre-conditions:	1. Should not be registered user

Post-conditions:	**Success end condition** Username, Password and other related information is valid **Failure end condition** Invalid Username or Password or other provided information **Minimal Guarantee** User can retry for registration
Main Success Scenario:	1. Enter the username 2. Enter the password 3. Enter email-id 4. Enter the contact number 5. Upload the attested certificate 6. Enter the credit card details 7. Enter the amount 8. Successful registration
Alternative Scenarios:	1 a. Username in not available 1. Go to step1 2 a. Contain less than 4 letters 1. Go to step2 2 b. Only character/digits 1. Go to step2 3 a. Invalid email-id 1. Go to step3 4 a. Invalid contact number 1. Go to step 4 5 a. Certificate attested by invalid person 1. Go to step5 5 b. Not attested to all the pages 1. Go to step5 6a. Invalid credit card details 1. Go to step6 7a. Amount may be less or more than specified 1. Go to step7

Variations:	Should accept the input through voice or hand written text
Priority:	High
Frequency of Use:	100 times a day
Business Rules:	Better to give unique identity like AADHAAR
Special Requirements:	Performance 1. It should have high availability (24*7) and faster access 2. Should be reliable, easy User Interface 1. Display all options and messages in English 2. The height of letters displayed on the display console shall not be smaller than 0.5 inches. Security 1. The system shall display the letters of password in a masked format when they are entered by the user. 2. Prevent password guessing 3. Multi key, multifactor authentication, challenge questions, mnemonic password
Assumptions:	The user understands English language
Notes and Issues:	Maximum length of username, password is 25 letters

Lab Exercise:

1. A bank client must be able to deposit an amount to and withdrawal an amount from his/her accounts using the touch screen at the ATM kiosk. Each transaction must be recorded. Date, time, balance amount and transaction type should be present in the recorded transaction. Bank client can have two types of accounts: current account and saving account. Client should be authenticated to access the bank account. OTP is must for money withdrawal and optional for money deposit. Draw the use case diagram and write the use-case specifications for any two main usecases.

2. Write the software requirements (functional and non-functional) and draw the use-case diagram for Online Placement System. Also write the use-case specification for atleast two usecases. Make use of all the relationships between usecases and between actors.

Additional Exercise

1. Write the software requirements (functional and non-functional) and draw the use-case diagram for Time Table Management System. Also write the use-case specification for atleast two use cases. Make use of all the relationships between usecases and between actors.
2. Write the software requirements and draw the use-case diagram for ATM kiosk.

[OBSERVATION SPACE – LAB 3]

LAB NO.: 4 **Date:**

ACTIVITY AND SWIMLANE DIAGRAM

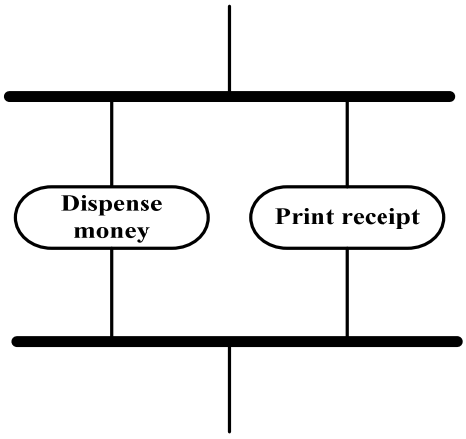
Objectives:

In this lab students should be able to:

- Understand the flow of control among different functionality
- Understand the actors associated with the functionality
- Draw activity and Swimlane in Rational Software Architect

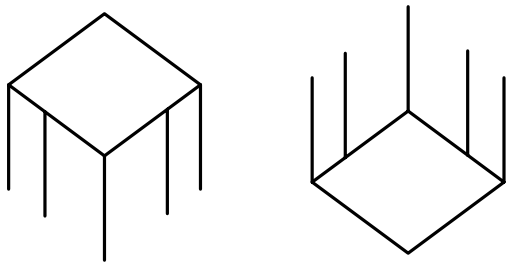
Activity Diagram

- ✓ Activity diagram represent the flow from one activity to another activity.
  - So the control flow is drawn from one operation to another.
- ✓ The flow can be sequential, branched or concurrent.
- ✓ Activity diagrams deals with different types of flow control by using elements like fork, join, decision box and merge etc.
- ✓ Example for *fork* and *join*

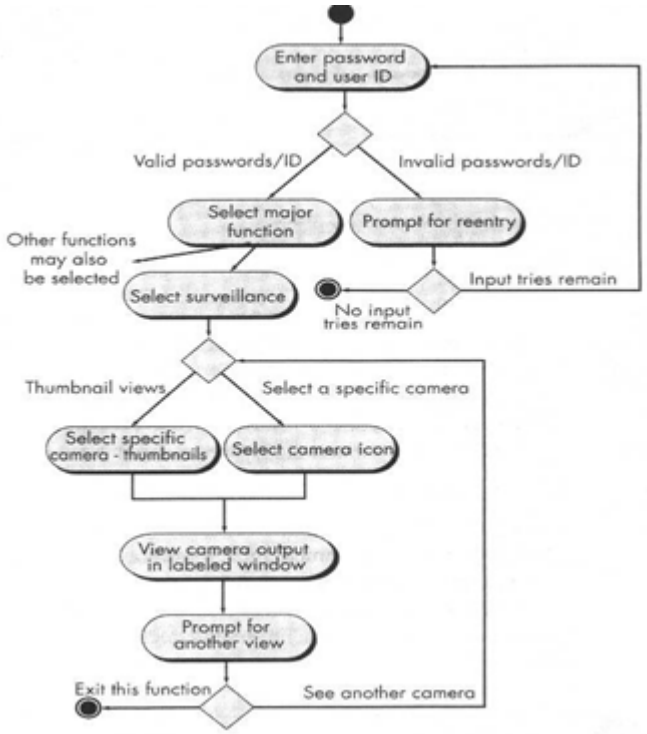


Dispensing money and printing the receipt are the parallel process while user withdraw the money from ATM kiosk.

- ✓ Example for *Decision box* with five outputs and *Merge box* with five inputs



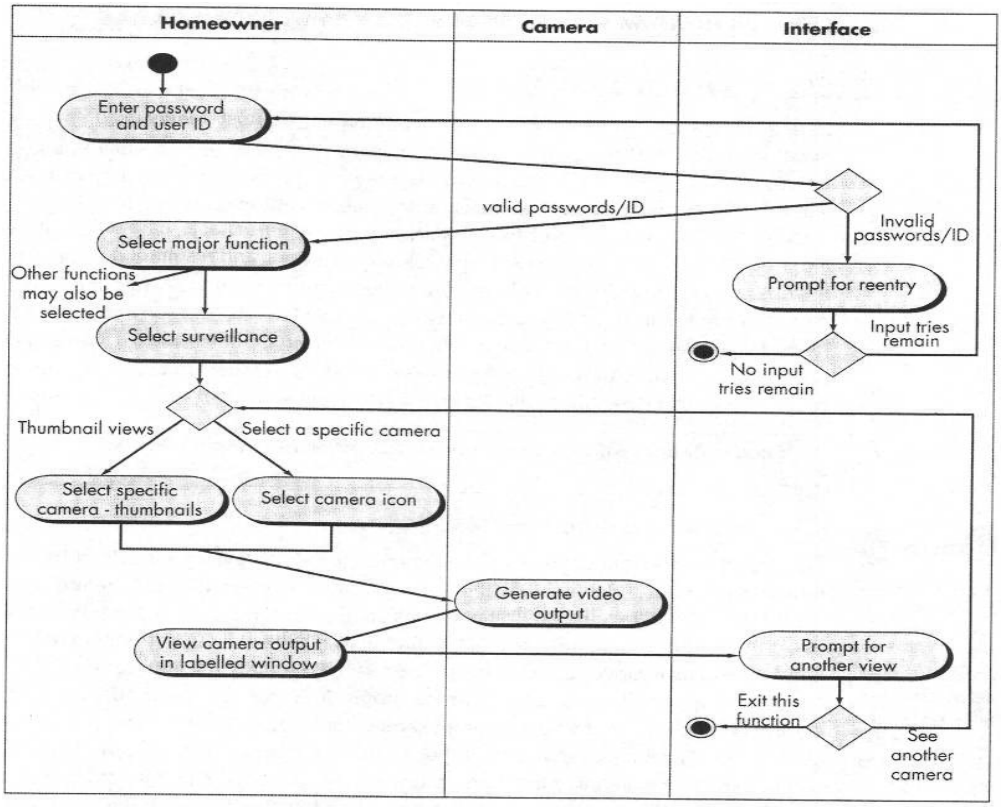
- ✓ Example for Activity diagram for access camera surveillance- display camera views function



Swimlane Diagram

- ✓ A variation of activity diagram.
- ✓ Indicate which actor has responsibility for the action described by an activity rectangle.

- ✓ A swimlane diagram represents the flow of actions, decisions and indicates which actors perform each action
- ✓ Example for Swimlane diagram for access camera surveillance- display camera views function



Lab Exercise:

1. A client needs to know the network address of the server for the communication. Client performs some work, which includes: Making a decision to initiate a connection to a server. If connecting to the server fails, or the server rejects the connection, the client may try again or may give up. On successful connection, the client sends the request to execute the services. GET_OS and GET_RAM are the services which will execute parallel and acknowledge will be sent to the client by the server.
2. Write the system requirements and draw the Activity and Swimlane diagram to function for Employee Performance Management System. Make use of all types of control flows.

Additional Questions:

1. Write the software requirements and draw the Activity and Swimlane diagrams by using all types of control flows for the following:
   i) Conduct any workshop and conference.
   ii) Release a new product into the market.

[OBSERVATION SPACE – LAB 4]

LAB NO.: 5 **Date:**

CLASS DIAGRAM – 01

Objectives:

In this lab students should be able to:

- Identify the class using noun phrase approach
- Draw the class diagram using Rational Software Architect

Noun Phrase Approach

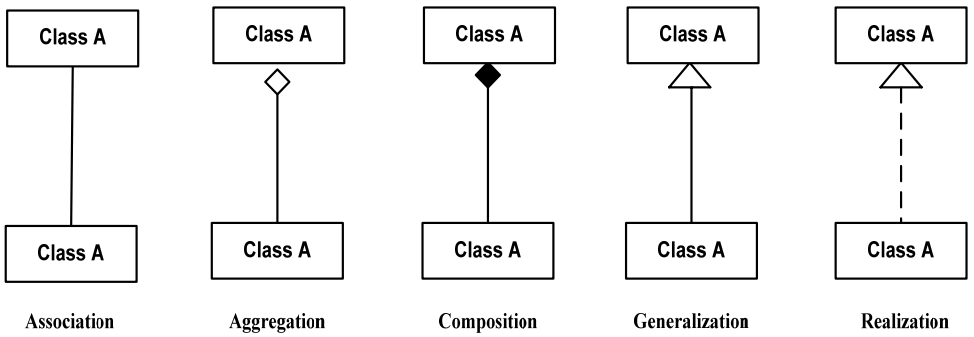
It consists of following phrases as follows

- ✓ Initial list of noun phrases
- ✓ Eliminate irrelevant classes
- ✓ Reviewing redundant classes
- ✓ Review adjective classes
- ✓ Review possible attributes
- ✓ Review the class purpose

Object Relationships, Attributes and Methods

i) Identifying Association Relationship
ii) Identifying a Super-sub Relationship
iii) Identifying the Composition, Aggregation/a-part-of Relationship & Collection member relationship

Example on Identifying Object Relationships



Object Relationship in Code

- ✓ **Association**

```
public class A {
        public void doSomething(B b) { }
        }
```

- ✓ **Aggregation**

```
public class A {
        private B  b1;
        public void setB(B b) { b1 = b; } }
```

- ✓ **Composition**

```
public class A {
        private B b1;
        public A() {
                b1 = new B();
                } }
```

- ✓ **Generalization**

```
public class A {
        ...
        } // class A
public class B extends A {
        ....
        } // class B
```

- ✓ **Realization**

```
public interface A {
        ...
        } // interface A
public class B implements A {
        ...
        } // class B
```

Note:

1) It is essentially a class attached to an association; the association itself is modeled as a class called *Association class*
2) Multiplicity & Cardinality Relationship can also be used in Class diagram

Lab Exercise:

1. Online publishing system allows authors to publish their various articles, which can be done free of cost. Article might be either book or paper. Authors need not be a registered user. Article size must not exceed 50 pages. It must be either in pdf, doc format only and must be in English. At a time only 1 article can be uploaded. Article will be reviewed by editor and will be published if it is up to the expectation else rejected. If it is rejected then authors can view the comments given and try to improve on their article. If article is successfully published then intimation will be sent indicating the sign of success for their submission.

   The members can search for various articles. They can view the papers published by various authors and can also buy books. There are two types of authorized users i.e., Institutional user and Personal user. They need to register by filling details such as Name, e-mail address and Contact information, Attestation certificates, Payment options and other details.

   Cost of registration for Institution is $125 for the period of 125 years. Validity of institutional user is as along as he/she is serving or studying in the institution. Institutional user can search and view for papers and books. They can buy books for $10. They need to sign-in in order to get these benefits. Personal user have period of registration of 3 years at the cost of $20. If their period of validation expires, then after which they can renew the period of registration. All the benefits provided to institutional user are also provided to these. Password recovery option is also provided to members.

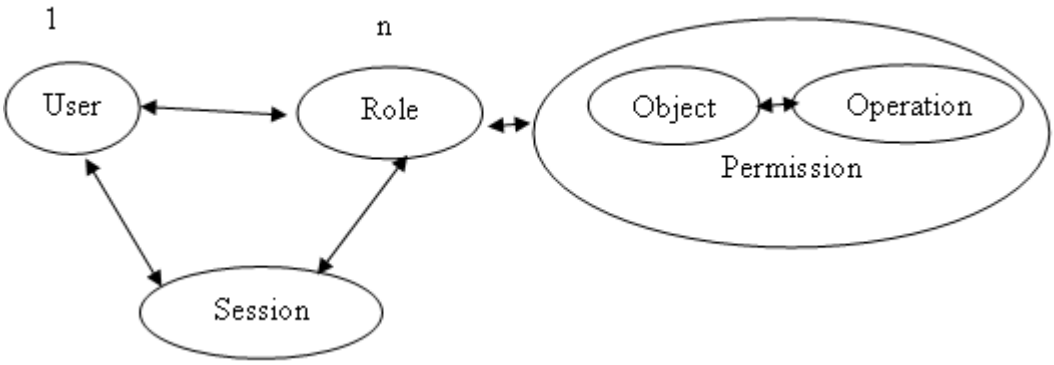
   Guest user can view only abstract part of the paper and can purchase the book for an amount more than the authorized user.

   Identify the class and draw the class diagram.

2. Write the software requirements and draw a class diagram for Order Processing System by identifying the class, relationship, attributes, methods, multiplicity and association name.

Additional Exercise:

1. Write the system requirements and draw a class diagram by identifying the class, relationship, attributes, methods, multiplicity and association name for
   i) Hospital Management System where functionalities are assigned based on roles (Role Based Access Control (RBAC) Model)



[OBSERVATION SPACE – LAB 5]

LAB NO.: 6 **Date:**

SOFTWARE REQUIREMENT SPECIFICATION OF THE PROJECT

Objective

- In this lab, student will be able to write the Software requirement specification for their project

Lab Exercise

1. Prepare Software Requirement Specification for the project

Note: Soft copy of the IEEE formatting requirements will be circulated to the students

---

[OBSERVATION SPACE – LAB 6]

LAB NO.: 7 **Date:**

CLASS DIAGRAM 02

Objectives:

In this lab students should be able to:

- Identify the class using common class pattern
- Identify the class using CRC
- Draw the class diagram using Rational Software Architect

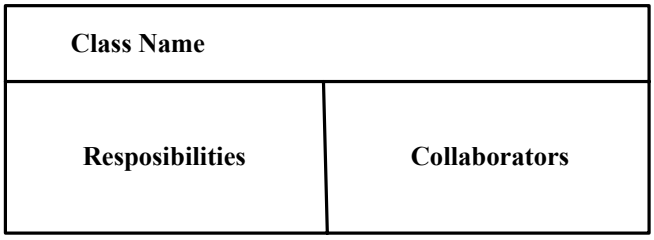
Common Class Pattern

Based on a knowledge base of the common classes proposed by various researchers

- ✓ Concept class
- ✓ Event class
- ✓ Organization class
- ✓ People class
- ✓ Place class
- ✓ Tangible things and device class

Class Responsibility Collaborator (CRC) model

It is a collection of standard index cards that have been divided into three sections, as depicted in figure. A class represents a collection of similar objects, a responsibility is something that a class knows or does, and a collaborator is another class that a class interacts with to fulfill its responsibilities



Lab Exercise:

1. The hospital has several specialized departments like Cardiology, Orthopedics, Pediatrics, ENT etc. OPD is another independent department. A doctor is only associated with one specialized department at a time though he/she can be a member of the OPD (Outside Patients Department) department. Each doctor has a visiting time and day in a week. At reception the patient details are entered and the fees are also taken and the patient is tracked on the basis of the Id generated. In routine a patient can visit the doctors either directly selecting a doctor or by getting admitted to the hospital and then a doctor visits the patients.

   A doctor can prescribe tests for the patient to perform. The patient visits the lab to get done the tests prescribed by his/her doctor. The reports are given to the patient. The payments pertaining to the tests are done at the reception. Referring the reports, the doctor prescribes the patient medicines or further tests or is asked to get admitted.

   A patient is admitted into a ward of a specialized department (if available) as per the doctor's prescription. The number of wards is limited and if there is no vacant ward the admission of the patient is rescheduled.

   As per the prescription of the doctor the patient is operated on a specified date and time as decided by the doctor who is doing the operation. Further at least 4 nurses should be present during the operation to assist the doctor. After the completion of the treatment a patient may get discharged on an advice of a doctor and upon the complete payment of all due charges at the reception. On payment of full dues the reception generates a discharge ticket for the patient.
   Assume suitable attributes and methods
   Draw a class diagram using Common Class Pattern

2. Write the software requirements and draw a class diagram for Social Media services by identifying the class, relationship, attributes, methods, multiplicity and association name. Draw a class diagram using CRC

Additional Exercise:

1. Write the software requirements and draw a class diagram by identifying the class, relationship, attributes, methods, multiplicity and association name
   i) Library Management System using common class pattern
   ii) Employer's Performance Management System using CRC

[OBSERVATION SPACE – LAB 7]

LAB NO.: 8 **Date:**

SEQUENCE DIAGRAM

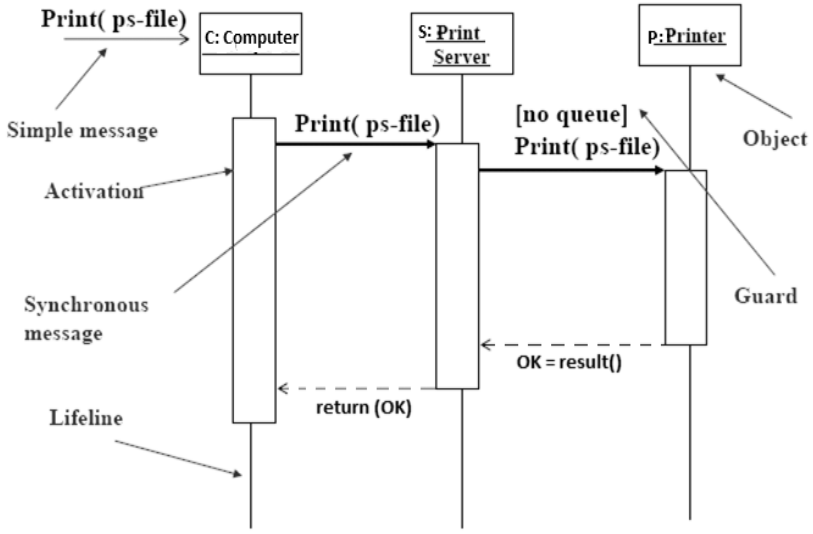
Objectives:

In this lab, student will be able to:

- Understand the interaction between the objects
- Draw the sequence diagram in Rational Software Architect

Introduction:

- ✓ Sequence diagrams illustrate how objects interact with each other.
- ✓ They focus on message sequences, that is, how messages are sent and received between a number of objects.
- ✓ The Branches, Conditions, and Loops may be included.
- ✓ Messages between system objects and actors may be Synchronous or Asynchronous
- ✓ Example for Sequence diagram- *Printer*



Lab Exercises

1. Nurse requests the Medical lab to reserve a date for the patient's diagnostic test (t). Nurse asks the insurance company to approve the test. The order in which these messages are sent/completed is irrelevant. If the Insurance Company approves the test, then the Nurse will schedule the test on the date supplied by the Medical lab. Draw the sequence diagram for the above description
2. Write the system requirements and draw a sequence diagram by showing both Synchronous or Asynchronous messages for Air/Bus/Train ticket Management System

Additional Questions:

2. Write the software requirements and draw a sequence diagram by showing both synchronous and asynchronous messages for the following
   i) Employment recruitment portal e.g. Naukri, Monster etc.
   ii) Student Scholarship Management System based on factors such as performance, caste, linguistic etc.

[OBSERVATION SPACE – LAB 8]

LAB NO.: 9 **Date:**

STATE MACHINE DIAGRAM

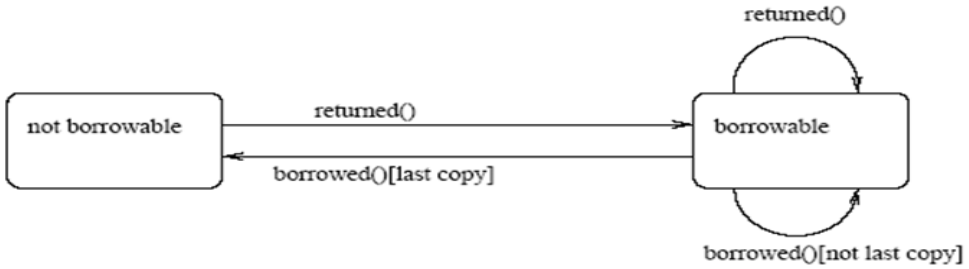
Objectives:

In this lab, student will be able to:

- Understand different states of an object
- Draw the state machine diagram using Rational Software Architect

Introduction

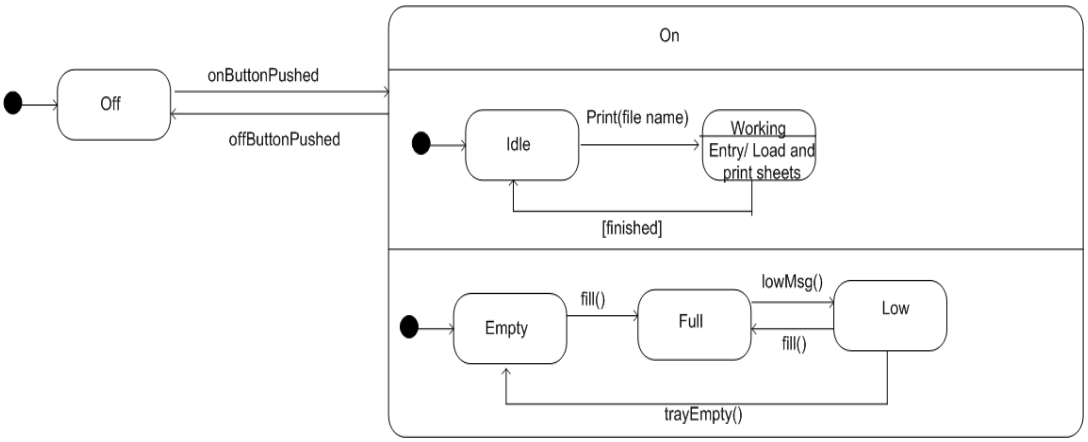
- ✓ Two interaction diagrams with objects of the same class receiving the same message may respond differently. This is because an object's behavior is affected by the values of its attributes
- ✓ Example for State machine diagram



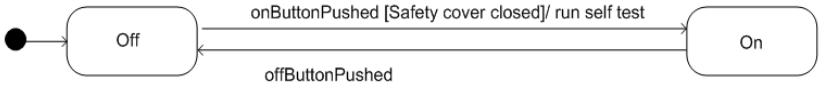
Class *Book* in Library Management System

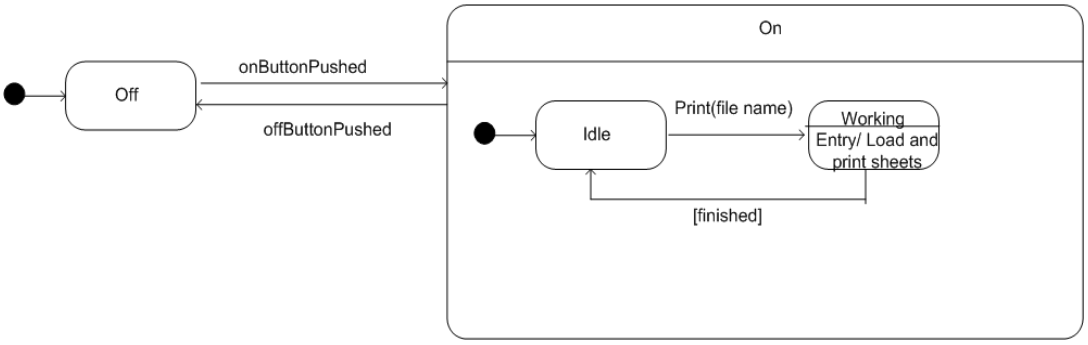
Advanced State Machine Modeling

✓ Example for Concurrent state machine diagram- *Printer*



✓ Example for Nested state machine diagram- *Telephone*





Lab Exercises

The client and server are the two modules, where server will produce the services requested by the client. Initially the server will be in the listen mode. Being in the listen mode the server may get a client's IP address. The server validates the IP address sent by the client. The server will be in the accepted stage and wait for the

filename which wants to be downloaded. On receiving a filename, server starts downloading. The size of file is 50Mbs. Due to the technical problem the server may be down after successful download of 30Mbs so that downloading of a file will be stopped else download will continue till the end of file. On server recovery, due to the installed download manager software the downloading of a file will start from the broken or paused downloads and continues till the end of file. Draw the state machine diagram for the above description

1. Write the software requirements and draw either nested or concurrent state machine diagram for vending machine by specifying event, action and guard condition

Additional Exercises

1. Write the software requirements and draw either nested or concurrent state machine diagram for the following by specifying event, action and guard condition
   i) Micro oven
   ii) Washing Machine

[OBSERVATION SPACE – LAB 9]

LAB NO.: 10 **Date:**

ANALYSIS AND DESIGN OF THE PROJECT

Objectives:

- In this lab, student will be able to draw usecase, activity, swimlane, class, sequence and state machine diagrams for their project.

Lab Exercise

1. Draw the usecase, activity, swimlane, class, sequence and state machine diagrams for the project.

[OBSERVATION SPACE – LAB 10]

LAB NO.: 11 **Date:**

IMPLEMENTATION OF THE PROJECT USING JAVA

Objectives:

- In this lab, student will be able to implement all the requirements specified in SRS

Lab Exercise

1. Implement the project using Java and Java Swing

[OBSERVATION SPACE – LAB 11]

LAB NO.: 12 **Date:**

TESTING OF THE PROJECT

Objective

In this lab students should be able to:

- Perform black box testing
- Perform unit test, system test, boundary value analysis and equivalence partitioning
- To calculate Cyclomatic complexity of a given functionality based on regions, edges and predicate nodes

Introduction

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test.

- ✓ Black-box testing treats the software as a "black box", examining functionality without any knowledge of internal implementation, without seeing the source code.
- ✓ White-box testing tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user, by seeing the source code.
- ✓ Unit testing, also known as component testing, refers to tests that verify the functionality of a specific section of code, usually at the function level.
- ✓ System testing, or end-to-end testing, tests a completely integrated system to verify that it meets its requirements
- ✓ Equivalence partitioning is the process of taking all possible test cases and placing them into classes. One test value is picked from each class while testing.
- ✓ Boundary value analysis is a next part of Equivalence partitioning for designing test cases where test cases are selected at the edges of the equivalence classes.
- ✓ Cyclomatic complexity is a metric that provides a quantitative measure of the logical complexity of a program. It is computed by 3 ways based on number of regions, edges and predicate nodes

Lab Exercise

1. Preform black box, unit test, system test, boundary value analysis and equivalence partition of the project. Refer test case template from the link http://www.softwaretestinghelp.com/test-case-template-examples/
2. Find Cyclomatic complexity of the project for any two code snippets

---

[OBSERVATION SPACE – LAB 12]

REFERENCES

1. Roger S. Pressman, “Software Engineering A practitioner’s approach”- McGraw Hill, 6th edition, 2005.
2. James Rumbaug, “Object Oriented Modeling and design”, Prentice-Hall of India Pvt. Ltd., 8th Reprint, 2000.
3. Grady Booch, Jamesh Rumbaugh, Ivar Jacobson, “The Unified Modeling Language User Guide”, Person Education, 2nd edition, 2005.
4. Ian Somerville, “Software Engineering”, Addison Wesley, 7th edition, 2006.