

# MN507 – Software Engineering

**Laboratory 7**: Objective Oriented and Service Oriented Design

**Submission Due**: End of laboratory class, submit the file on Moodle at least 10 minutes before the end of laboratory class.

**Total Marks** = 5 marks for 10 weeks

Marks will be given only to students who attend and participate during 2-hour laboratory class. Submission on Moodle is mandatory as an evidence of participation.

**Description of the laboratory exercise**: Form a group to work during the semester:

* 3 members each group: write your names and student ID in every submission.
* Select a leader for the first 3- 4 weeks (leadership will rotate every 3-4 weeks).
* Most of the questions throughout these lab sessions, you can discuss with your group members, but you have to submit your own answer as an individual submission on Moodle. However, some of the lab work/assignments you may need to submit only one submission as a group. Therefore, carefully read the given instruction in every lab sheet/assignment.

# Learning Outcome:

Use appropriate modelling techniques to specify system requirements. Produce and communicate to others the functional specifications of a system.

*MN507 Software Engineering Overview Laboratory 7* Page **2** of **2**

# Knowledge Tasks:

1. Describe attributes and methods of a class in object oriented programming.

**Answer:**

In object oriented programming two attributes, storage class and type are used as 'identifiers'. Together these attributes allow data to be checked for correct functioning. Attributes are an aspect of object oriented programming that are given a great deal of emphasis in c++ and some other object oriented languages.

A method in object-oriented programming (OOP) is a procedure associated with a message and an object.

Data is represented as properties of the object and behaviors are represented as methods of the object.

1. Describe the differences between object and class in object-oriented programming.

**Answer:**

Classes and objects from the essential part of Object-oriented programming, where a class can be considered as a construct which encapsulates a group of variables and methods; whereas, an object acts as member or instance of that class

1. Describe encapsulation, inheritance, and polymorphism in object-oriented programming.

**Answer:**

**Encapsulation** in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit. In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class. Therefore, it is also known as data hiding.

**Inheritance** can be defined as the process where one class acquires the properties (methods and fields) of another. With the use of inheritance the information is made manageable in a hierarchical order.

**Polymorphism** is the ability of an object to take on many forms. The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.

Any Java object that can pass more than one IS-A test is considered to be polymorphic. In Java, all Java objects are polymorphic since any object will pass the IS-A test for their own type and for the class Object.

1. Discuss the use of throw-away prototype and evolutionary prototype with simple examples.

**Answer:**

In evolutionary prototyping, the concept of the system will be developed as the project progresses. First of all, we will develop the most visual aspect of the system. You present a portion of the system to the customer and continue to develop prototypes based on the feedback received. At some point, you and the customer agree that the prototype is “good enough” and release the prototype as the final product.

It is may be best suited for business systems where developers can interact frequently and informally with end users. However, it is suitable for commercial, shrink wrap, and system projects as long as end-user involvement is obtained. Communication with users of these projects is generally more structured and formal.

If evolved prototyping does not provide more control than is necessary or if you already know what the system should do, you can use evolutionary delivery or gradual delivery instead.

# Practical Tasks:

**You are to draw sequence diagram for the selected software system.**

1. Develop a sequence diagram showing the interactions involved when a student registers for a course in a university. Courses may have limited enrolment, so the registration process must include checks that places are available. Assume that the student accesses an electronic course catalogue to find out about available courses.
2. You have already drawn the **Use Case Diagram and Use Case Scenarios** for the following.

**Scenario:** Consider the following description of a ATM (automated teller machine) system: An ATM or the automatic banking machine (ABM) is a banking subsystem ([subject](http://www.uml-diagrams.org/use-case-subject.html)) that provides bank customers with access to financial transactions in a public space without the need for a cashier, clerk, or bank teller.

Customer uses bank ATM to Check Balances of his/her bank accounts, Deposit Funds, Withdraw Cash and/or Transfer Funds.

ATM Technician provides Maintenance and Repairs. All these activities also involve with the Bank, whether it is related to custome0r transactions or to the ATM servicing.

* 1. Identify different **classes** involve in during the execution of “**Withdraw Money”**

use case.

**Answer:**

Customer

Bank

Account

Current Account

Savings Account

Atm Transactions

Atm

# Draw the sequence diagram for “Withdraw Money” main use case scenario.

**Answer:**

# Image result for sequence diagram for âWithdraw Moneyâ

* 1. Show all the messages passing between different objects

**Answer:**

Card enter

Request pin

Enter pin

Verify

Valid

Withdrawal amount

Request amount

Check amount

Success/failure

Take cash

Continuation

Receipt

# It is an individual task. Each student is required

1. **Upload your work into your Github**
2. **Submit the Github screenshot in the Moodle**
3. **Submit your work in the Moodle as well.**

