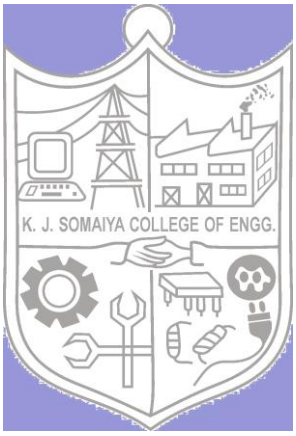


Experiment No.5

Title: Modelling Behaviour – Use Case Diagram and Interaction Diagram using UML

Batch: B3 **Roll No.: 1714103** **Experiment No.: 5**

Aim: To model Behaviour – Use Case Diagram and Interaction Diagram using UML

Resources needed: IBM Rational Rose/Open Source UML Tool

Theory

UML specification defines two major kinds of UML diagram: **structure diagrams and behavior diagrams.**

Structure diagrams show the static structure of the system and it's parts on different abstraction and implementation levels and how they are related to each other. The elements in a structure diagram represent the meaningful concepts of a system, and may include abstract, real world and implementation concepts.

Behavior diagrams show the dynamic behavior of the objects in a system, which can be described as a series of changes to the system over time.

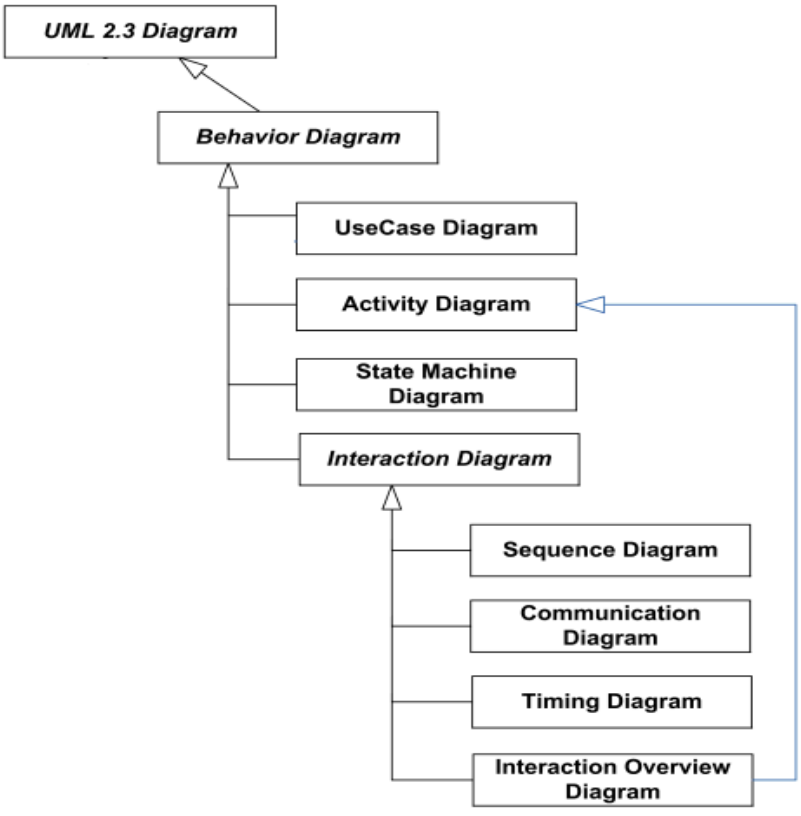
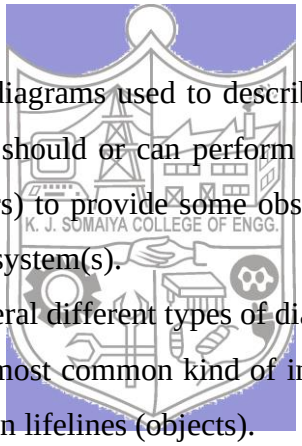


Figure 4.1 UML Behaviour Diagram

Behavior diagrams show the dynamic behavior of the objects in a system, which can be described as a series of changes to the system over time.



Use case diagrams are behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors) to provide some observable and valuable results to the actors or other stakeholders of the system(s).

Interaction diagrams include several different types of diagrams:

- **Sequence diagram** is the most common kind of interaction diagrams, which focuses on the message interchange between lifelines (objects).
- **Interaction overview diagram** defines interactions through a variant of activity diagrams in a way that promotes overview of the control flow. Interaction overview diagrams focus on the overview of the flow of control where the nodes are interactions or interaction uses. The lifelines and the messages do not appear at this overview level.
- **Communication diagram (previously known as Collaboration Diagram)** is a kind of interaction diagram, which focuses on the interaction between lifelines where the architecture of the internal structure and how this corresponds with the message passing is central. The sequencing of messages is given through a sequence numbering scheme.
- **Timing diagrams** are used to show interactions when a primary purpose of the diagram is to reason about time. Timing diagrams focus on conditions changing within and among Lifelines along a linear time axis.

Use case diagrams

A use case diagram is a behavioural diagram, which aims to present a graphical overview of the functionalities provided by the system. It consists of a set of actions (use cases) that the concerned system can perform, one or more actors, and dependencies among them.

Actor

An actor can be defined as an object or set of objects, external to the system, but interacts directly with the system to get some meaningful work done. Actors could be human, devices, or even other systems.

For example, consider the case where a customer withdraws cash from an ATM. Here, customer is a human actor. However, before any transaction can proceed, the ATM must authenticate the customer by verifying in his PIN. Here, the "ATM authentication service" is a non-human actor.

Actors can be classified as below :

Primary actor: Primary actors fulfill their goal by using some service from the system. For example, a customer uses an ATM to withdraw cash when he needs it. So, customer is the primary actor here.

Supporting actor: They provide some service to the system. "ATM authentication service" is such an example.

In a use case diagram primary actors are usually drawn on the top left side of the diagram.

Use Case

A use case can be defined as a functionality that a system can provide by interacting with the actor(s).

Continuing with the example of the ATM, withdraw cash is a functionality that the ATM provides. Therefore, this is a use case. Other possible use cases include check balance, change PIN, and so on.

Use cases include both successful and unsuccessful scenarios of user interactions with the system. For example, authentication of a customer by the ATM would fail if he enters wrong PIN. In such case, an error message is displayed on the screen of the ATM.

Subject

Subject can be defined as the system under consideration to which the use cases apply.

Graphical Representation

An actor is represented by a stick figure and name of the actor is written below it. A use case is depicted by an ellipse and label of the use case is written inside it. The subject could be shown by drawing a rectangle. Label for the system could be put inside it either at the top or near the bottom. Use cases are enclosed inside the rectangle and actors are drawn outside the rectangle, as shown in figure 4.2.

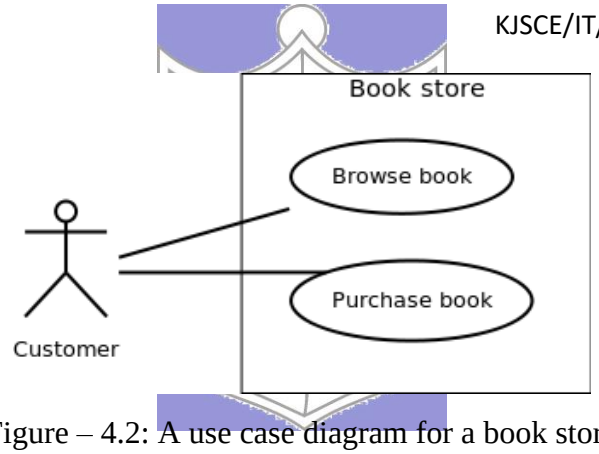


Figure – 4.2: A use case diagram for a book store

Association between Actors and Use Cases

A use case describes a specific functionality that the system provides to its users. The functionality is triggered by actor. Actors are connected to use cases through binary associations. The association indicates that the actor and use case communicates through message passing.

An actor must be associated with at least one use case. Similarly, a given use case must be associated with at least one actor. However, when a use case is associated with multiple actors, it might not be clear who triggers the functionality. No association among the actors are shown.

Use Case Relationships

Three types of relationships exist among use cases:

Include relationship, Extend relationship, and Use case generalization

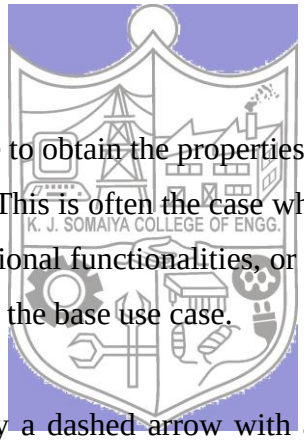
Include Relationship

Include relationships are used to depict similar behaviour that are shared by multiple use cases without replicating the common behaviour in each of those use cases.

For example, consider an email application. A user can send a new mail, reply to an email he has received, or forward an email. However, in each of these three cases, the user must be logged in to perform those actions. Thus, we could have a login use case, which is included by compose mail, reply, and forward email use cases.

Notation

Include relationship is depicted by a dashed arrow with a <<include>> stereotype from the including use case to the included use case.



Extend Relationship

A use case extends a base use case to obtain the properties and behaviour of the base use case as well as add some new features. This is often the case when the extended use case could not be modified to accommodate additional functionalities, or there are multiple use cases having behaviour and properties similar to the base use case.

Notation

Extend relationship is depicted by a dashed arrow with a <<extend>> stereotype from the extending use case to the extended use case.

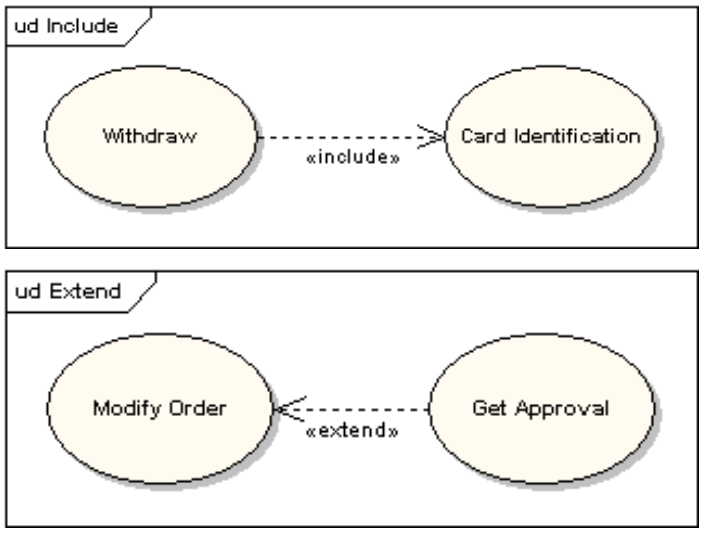


Figure 4.3 Include and Extend relationship

Generalization Relationship

Generalization relationship exists between a base use case and a derived use case when the derived use case specialize some functionalities it has inherited from the base use case.

To illustrate this, consider a graphical application that allows users to draw polygons. We could have a use case draw polygon. Now, rectangle is a particular instance of polygon having four sides at right angles to each other. So, the use case draw rectangle inherits the properties of the use case draw polygon and overrides it's drawing method. This is an example of generalization relationship. Similarly, a generalization relationship exists between draw rectangle and draw square use cases.

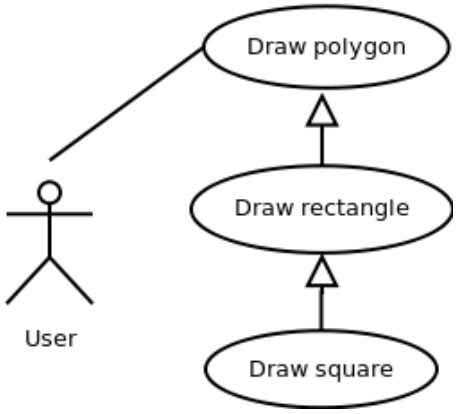


Figure 4.4 Generalization Relationship

Notation

Generalization relationship is depicted by a solid arrow from the specialized (derived) use case to the more generalized (base) use case.

Identifying Actors

Given a problem statements, the actors could be identified by asking the following questions:

Who gets most of the benefits from the system? (Primary actor)

Who are the people involved to keep the system working and running?

Will the system interact with other software / hardware?

Will the system take service from or provide service to any other system?

Identifying Use cases

Once the primary and secondary actors have been identified, their goals have to identify i.e. what are the functionalities they can obtain from the system. Any use case name should start with a verb like, "Withdraw Cash".

Interaction Diagram

From the name Interaction it is clear that the diagram is used to describe some type of interactions among the different elements in the model. So this interaction is a part of dynamic behavior of the system. This interactive behavior is represented in UML by two diagrams known as Sequence diagram and Collaboration diagram. The basic purposes of both the diagrams are similar. Sequence diagram emphasizes on time sequence of messages and collaboration diagram emphasizes on the structural organization of the objects that send and receive messages.

Sequence diagram

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence. Sequence diagrams are closely related to collaboration diagrams and both are alternate representations of an interaction. There are two main differences between sequence and collaboration diagrams: sequence diagrams show time-based object interaction while collaboration diagrams show how objects associate with each other.

A sequence diagram has two dimensions: typically, vertical placement represents time and horizontal placement represents different objects.

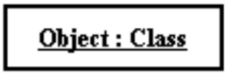
Need of Sequence Diagram:

Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces. This type of diagram is best used during early analysis phases because they are simple and easy to comprehend. **Sequence diagrams are normally associated with use cases.**

Elements of Sequence Diagram:

The following tools located on the sequence diagram toolbox enable to model sequence diagrams:

Class roles: Class roles describe the way an object will behave in context. Use the UML object symbol to illustrate class roles, but don't list object attributes.



Activation:

Activation boxes represent the time an object needs to complete task.

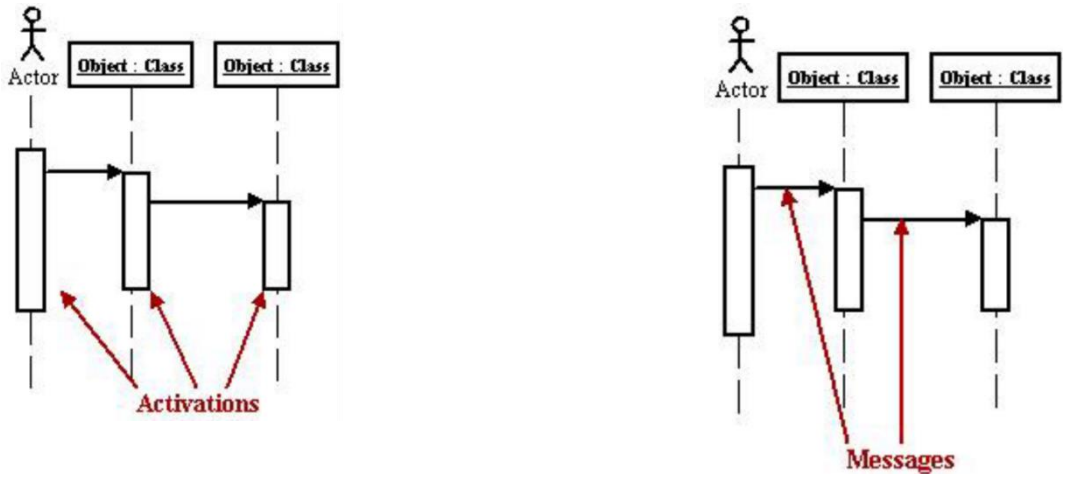


Figure 4.5 Activation and Messages

Messages:

Messages are arrows that represent communication between objects. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks.

Message Types, Life line and Destroying Objects:

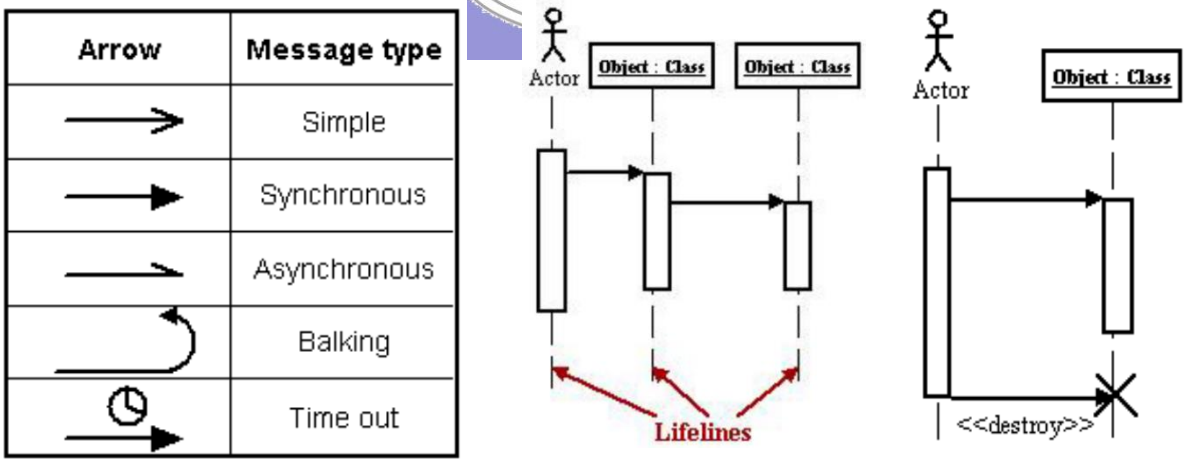


Figure 4.6 Message Types, Life line and Destroying Objects

Loops

A repetition or loop within a sequence diagram is depicted as a rectangle. Place the condition for exiting the loop at the bottom left corner in square brackets [ ].

Collaboration diagram

A collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages.

We form a collaboration diagram by first placing the objects that participate in the interaction as they exist in a graph, and then add the links that connect these objects as arcs of this graph. Finally adorn these links with the messages that objects sends and receive with sequence numbers.

Need of collaboration Diagram:

We use collaboration diagram to describe a specific scenario. Numbered arrows show the movement of messages during the course of scenario. A distinguishing feature of a collaboration diagram is that it shows the objects and their association with other objects in the system apart from how they interact with each other. The association between objects is not represented in a sequence diagram.

Elements of collaboration Diagram:

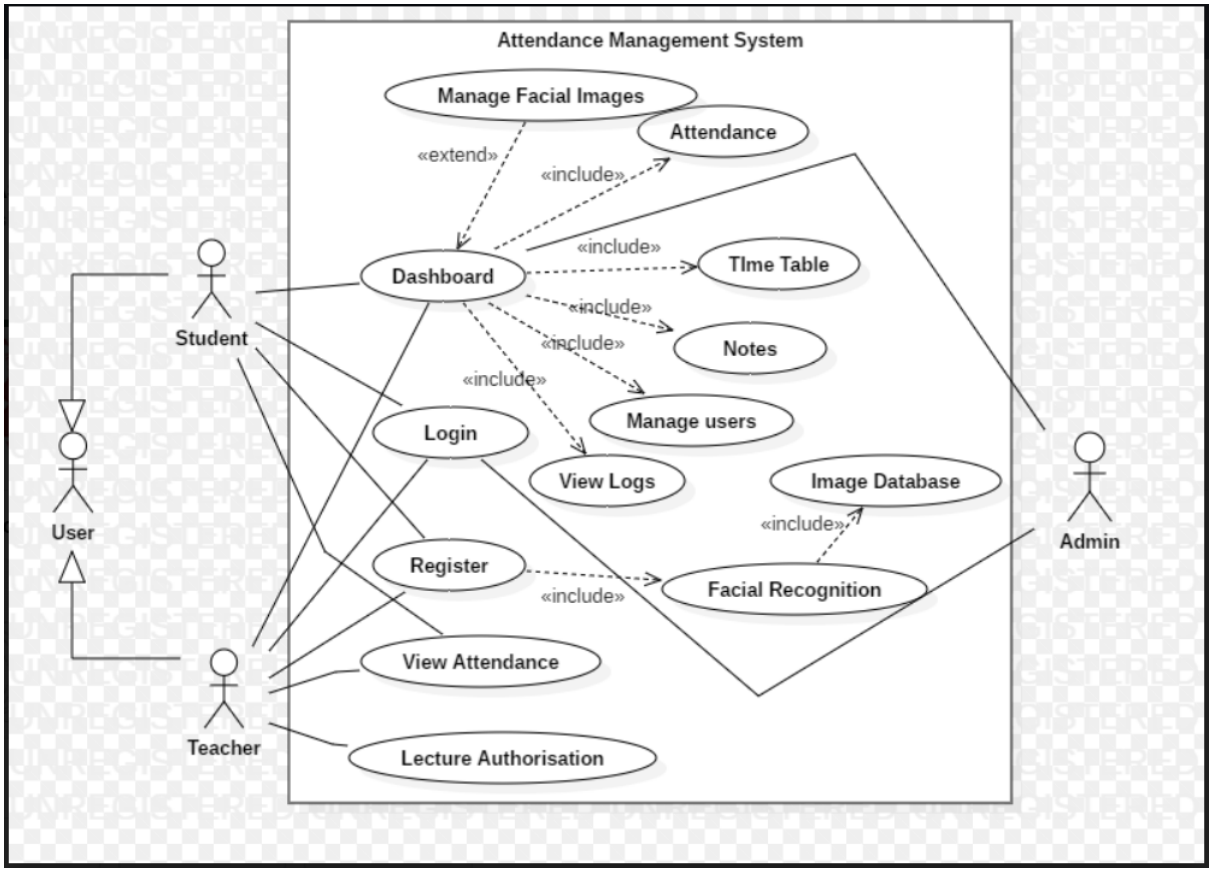
A sophisticated modelling tool can easily convert a collaboration diagram into a sequence diagram and the vice versa. Hence, the elements of a Collaboration diagram are essentially the same as that of a sequence diagram.

Procedure:

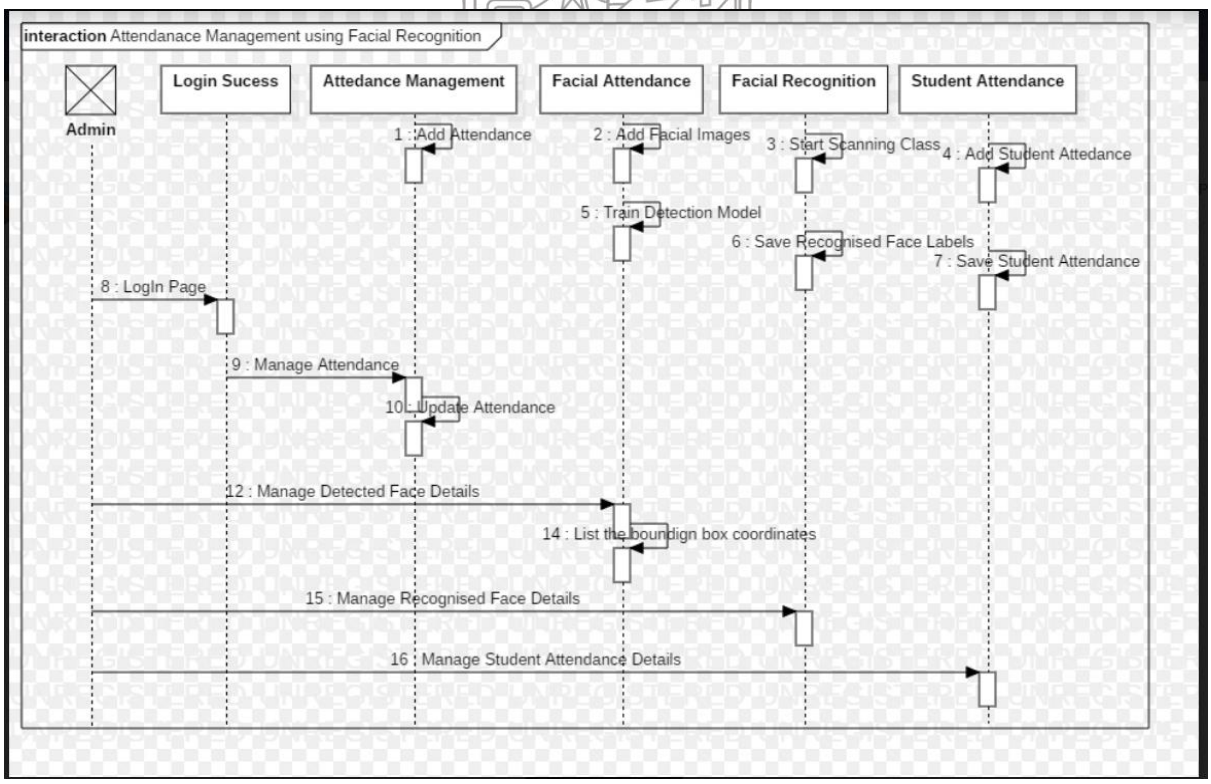
Prepare mentioned behavior diagrams for chosen problem using Rational Rose/ any other Open Source UML tool.

Results:

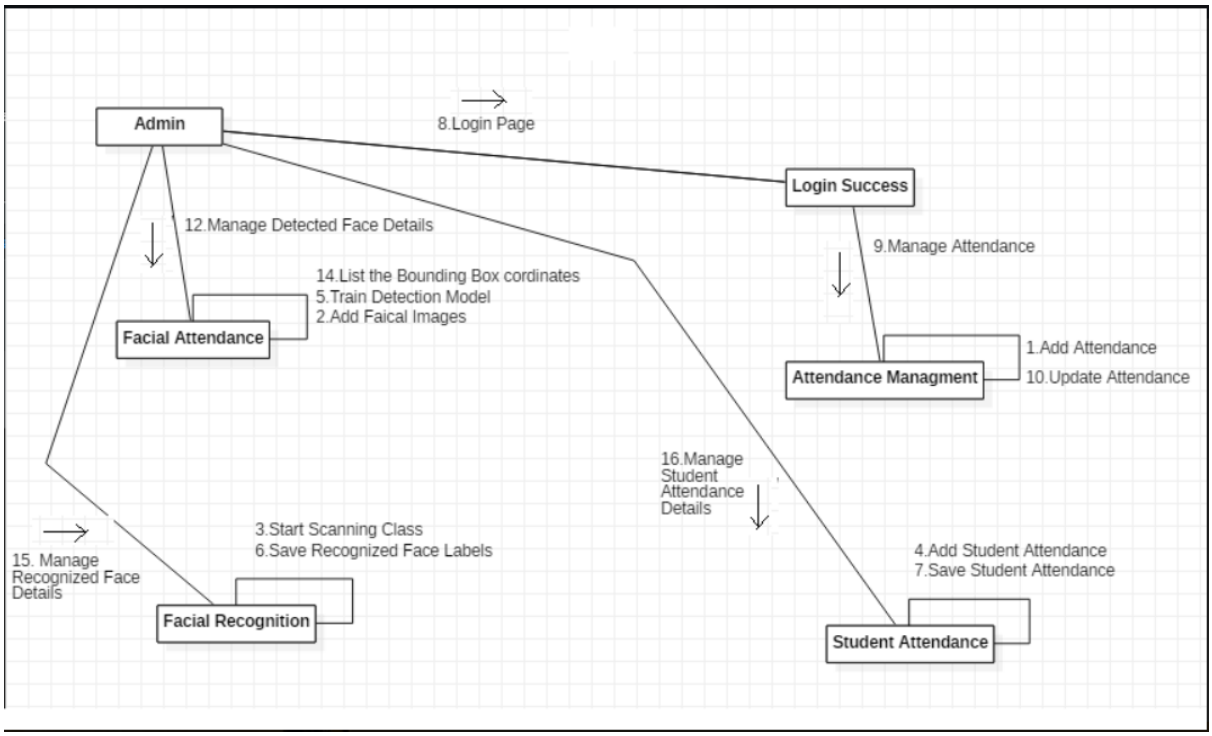
Use Case:



Sequence Diagram:



Collaboration Diagram:



Questions:

1. In a use case diagram, relationships between different actors are normally shown.

True False

Ans. **False**

2. A Communication diagram specifies a scenario.

True False

Ans. **True**

Outcomes: Model the requirements using UML.

Conclusion: In this experiment we understood the implementation of Use Case, Sequence and Collaboration diagram and also applied the same for our project.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References:

Books/ Website:

1.Michael Blaha, James Rumbaugh, "Object-Oriented Modeling and Design with UML", Prentice-Hall of India, 2nd Edition
2. Mahesh P. Matha, "Object-Oriented Analysis and Design using UML", Prentice-Hall of India
3. Timothy C Lethbridge, Robert Laganiere, "Object-Oriented Software Engineering – A practical software development using UML and Java", Tata McGraw-Hill, New Delhi.
4. http://www.uml-diagrams.org/uml-23-diagrams.html
5. http://vlabs.iitkgp.ernet.in/se/