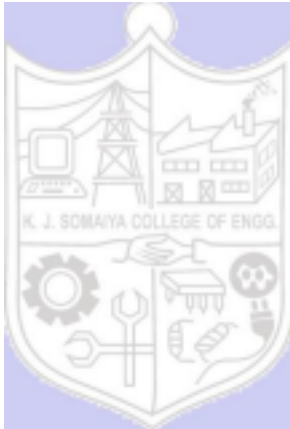**Experiment No. 5**

Title: Creation of UML Diagrams

Batch: B2 Roll No.: 16010421059 Experiment No.: 5

Aim: Creation of UML Diagrams

_____

Resources needed: IBM Rational Rose/Open Source UML Tool-star UML

_____

Theory

Overview of UML Diagrams

Structural
: element of spec. irrespective of time

- Class
- Component
- Deployment
- Object
- *Composite structure*
- *Package*

Behavioral
: behavioral features of a system / business process

- Activity
- State machine
- Use case
- *Interaction*

Interaction
: emphasize object interaction

- Communication(collaberati on)
- Sequence
- *Interaction overview*
- *Timing*

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

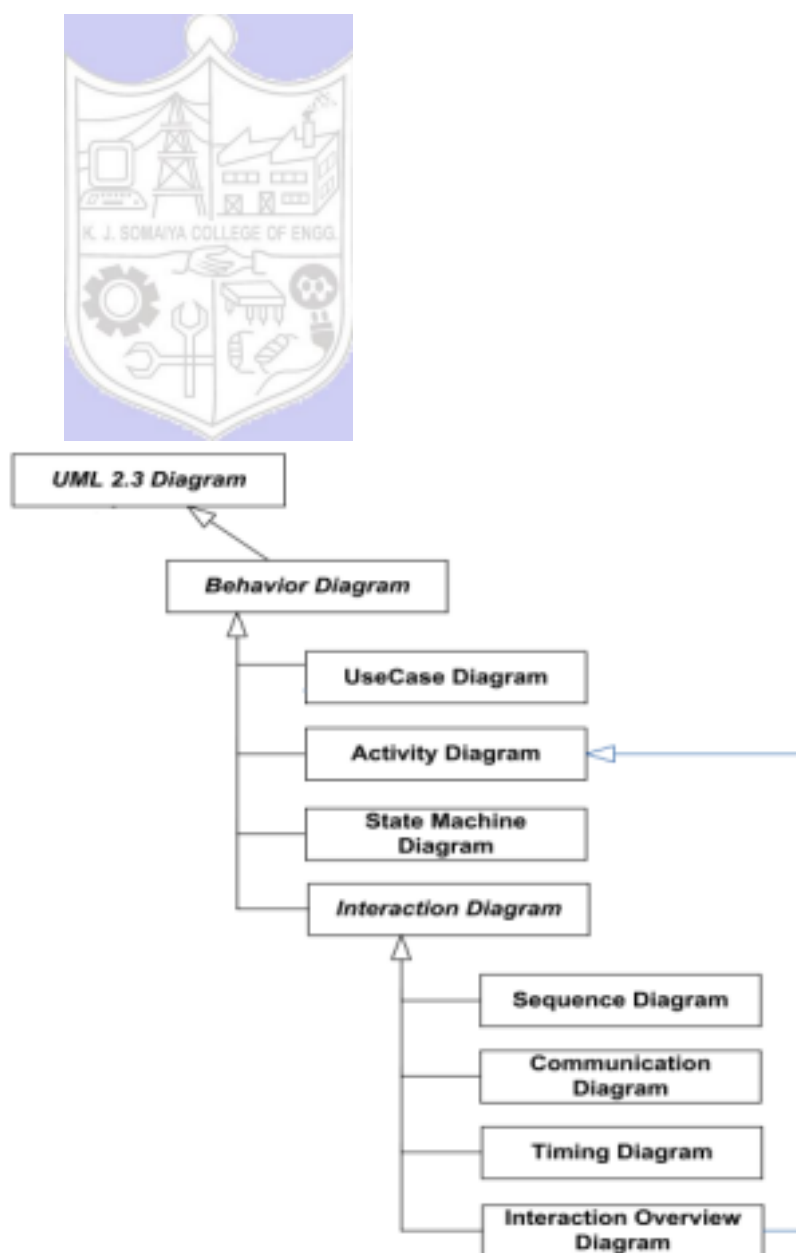Figure 4.1 UML Behaviour Diagram

I. Behavior 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.

1. 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).

2. 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.

3. Collaboration diagram

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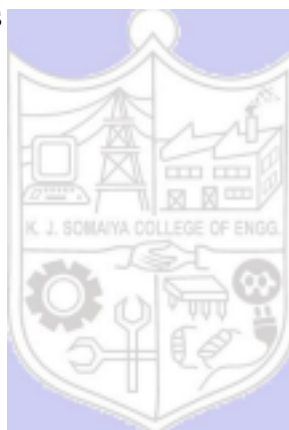
structural organization of the objects that send and receive messages.

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.

2 UML State machine diagram and activity diagram are both behavioural diagrams but have different emphases.

4. <u>Activity diagram</u> shows sequence and conditions for coordinating lower-level behaviors, rather than which classifiers own those behaviors. These are commonly called control flow and object flow models.

5. <u>State machine diagram</u> is used for modelling discrete behavior through finite state transitions. In addition to expressing the behavior of a part of the system, state machines can also be used to express the usage protocol of part of a system. These two kinds of state machines are referred to as behavioral state machines and protocol state machines.

II. Structure Diagrams:
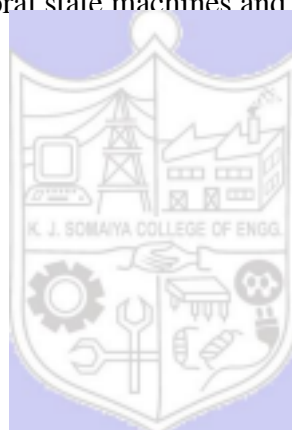
Structure diagram shows static structure of the system and its parts on different abstraction and implementation levels and how those parts are related to each other. The elements in a structure diagram represent the meaningful concepts of a system, and may include abstract, real world and implementation concepts.

Structure diagrams are not utilizing time related concepts; do not show the details of dynamic behavior. However, they may show relationships to the behaviors of the classifiers exhibited in the structure diagrams.

1. Class diagram is static structure diagram describing structure of a system on the (lowest) level of classifiers (classes, interfaces, etc.). It shows system's classifiers, their attributes, and the relationships between classifiers.

2.Object diagram shows instances of classifiers and links (instances of associations) between them.

3. Package Diagram shows packages and dependencies between the packages. Models allow to show different views of a system, for example, as multi-layered (aka multi-tiered) application

4. Component Diagram shows components and dependencies between them. This type of diagrams is used for Component-Based Development (CBD), to describe systems with Service-Oriented Architecture (SOA).

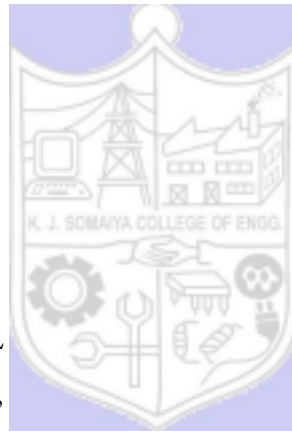**Composite Structure diagram** could be used to show:

   Internal structure of a classifier

Internal Structure diagrams show internal structure of a classifier - a decomposition of the classifier into its properties, parts and relationships.

   Behaviour of collaboration

Collaboration use diagram shows objects in a system cooperating with each other to produce some behavior of the system.

Deployment diagram shows execution architecture of a system that represents the assignment (deployment) of software artifacts to deployment targets (usually nodes).

Profile diagram is auxiliary UML diagram which allows defining custom stereotypes, tagged values, and constraints. The Profile mechanism has been defined in UML for providing a lightweight extension mechanism to the UML standard. Profiles allow to adapt the UML meta model for different platforms (such as J2EE or .NET), or domains (such as real-time or business process modeling).

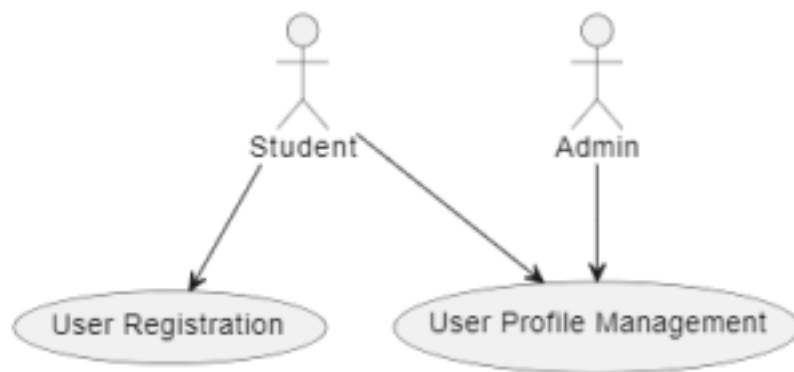Profile diagrams were first introduced in UML 2.0.

Procedure:

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

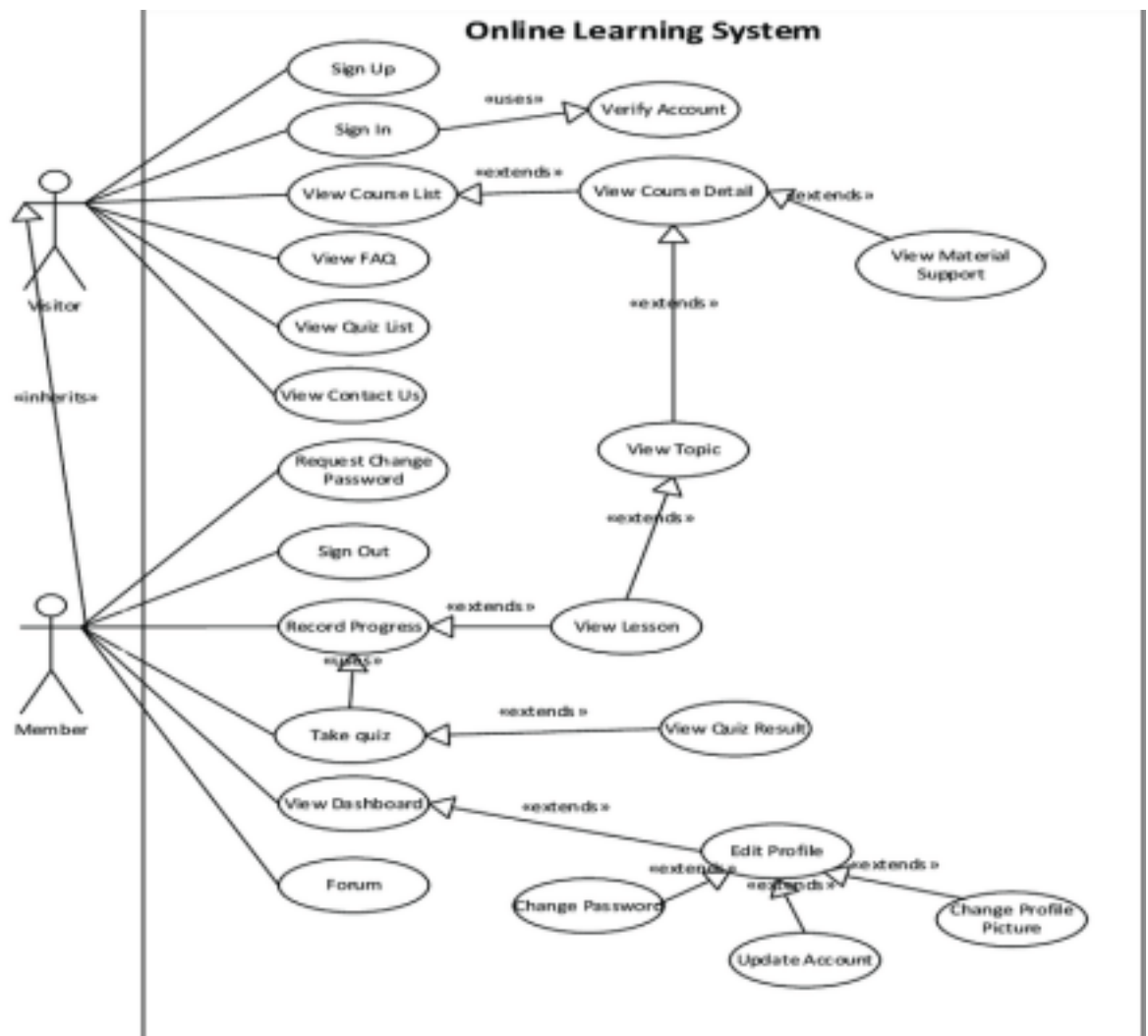_____

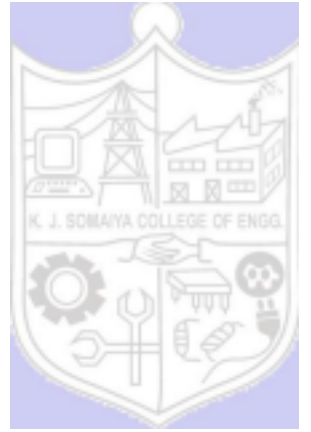**Results: Printout of mentioned behavior diagrams**

1. Use Case:
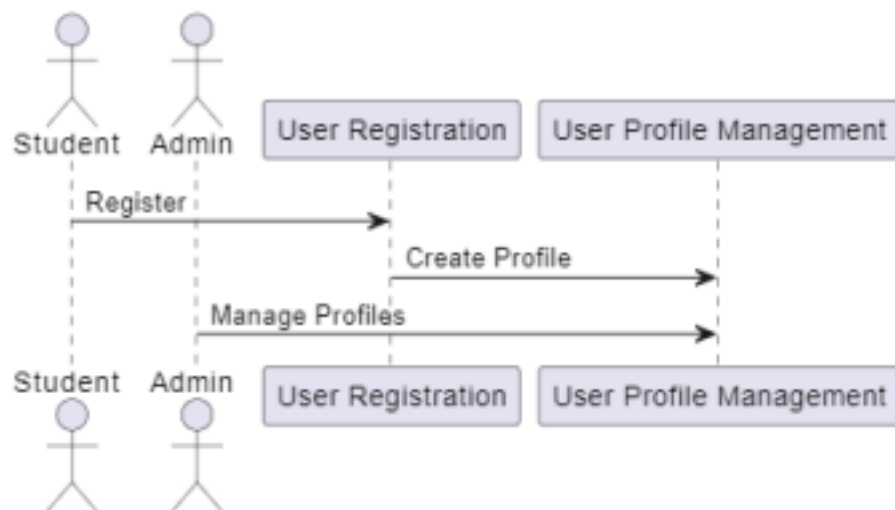
Student   Admin

User Registration   User Profile Management

Register

Create Profile

Manage Profiles

Student   Admin   User Registration   User Profile Management

Online Learning System

Sequence Diagram:
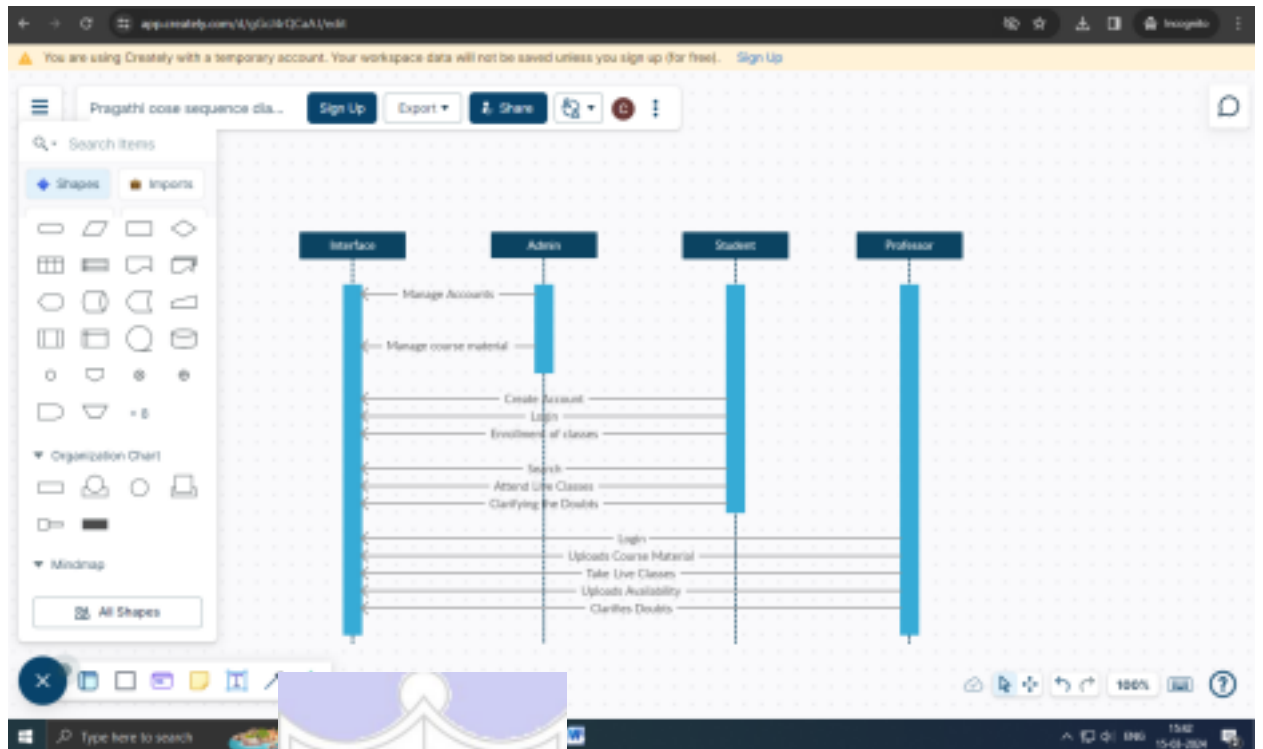
Communication/ Collaboration Diagram:

Activity Diagram:

State Chart:

_____Deployment :

Component:

_____

Class:
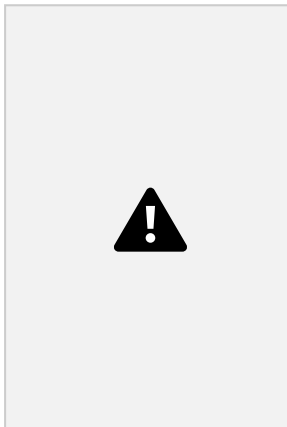
Questions:

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

   True False

2. A Communication diagram specifies a scenario.

 True False

3. Explain the difference between activity diagram and state chart diagram. 4.

Give example of deployment diagram for modelling a fully distributed system.

_____

Outcomes:

Conclusion:

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