UNIT 3 BEHAVIORAL MODELING USING UML

Structure **Page no.**

3.0 INTRODUCTION

Object-Oriented Modeling (OOM) is used for modeling the application from the initial stage of the software development. There are different types of objects used in modelling. Every object has its shape and size. It is used to store the information in a single unit. Every value has assigned a unique name which is identified from its reference. Every element of an object has its own property.

Objects are created and executed by describing them inside the program. Each object is defined in two ways: system-defined objects and user-defined objects. Every object has its class and instances. A class of objects is defined by declaring the instances of the class.

The object's instance is based on the set of rules in the class. In an object, a data element is also known as a property, while the object contains for processing the values are called methods. Objects cannot store only the data, but they can store the instructions on what to do with the data.

A model is a representation of something (object, system, process) in one medium or another. A model of software is designed in a language is called UML. The classification of UML diagrams is shown in figure 3.1. It can represent semantics and notation forms which include text and picture. We can divide it into three parts: structural classification, dynamic behaviour, and model management.

- The first part is the structural classification, which describes the relationship between one object to another object; while the classifier describes the Static view, Use Case view, and Implementation view.
- Dynamic behaviour depicts the behaviour of a system with time. Behaviour can be changed with time and can be represented using a State Machine, Activity, and Interaction Diagrams.

- Model management is the third part that describes the organization in a hierarchical structure.

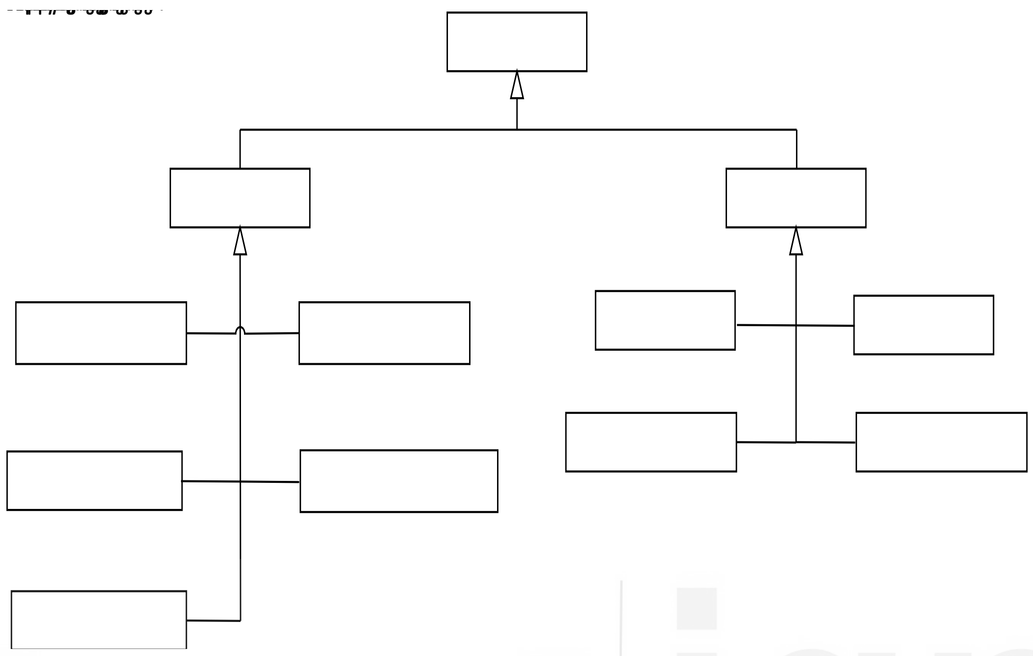


Figure 3.1: Classification of UML Diagram

3.1 OBJECTIVES

After going through this unit, you should be able to:

- Explain behavioural models,
- Explain the rules and roadmap of sequence and communication diagrams,
- Describe the process used to produce sequence and communication diagrams,
- Explain the relation between the behavioural models, structural and functional models and,
- Create sequence diagram, collaboration diagram, and CRUDE matrix.

3.2 BASICS OF BEHAVIORAL MODELING

The behavior model helps to narrate the internal parameters of the entire system, which supports all the activities of an organization. It also expresses the system's internal logic without determining how the process is to be executed. The detailed design and implementation phase describe the complete operation of the object. When an analyst is attempting to understand the fundamental application domain of a problem, then it should consider both the structural aspects of the problem. An interactive process integrates the individual behavioural model, functional and structural models.

The objective of the behavioural models is to display the fundamental objects in a problem domain that will work together to form a collaboration to support each of the use-cases. At the same time, the structural model represents the objects and the relationships between them.

This unit, focus on various behavior models with the help of interaction diagrams (sequence and communication diagrams) and activity diagrams. It is easy to provide a complete structure of the dynamic aspects of the evolving business information system with the help of behaviour model. This unit first, describe the behavioural

ignou THE PEOPLE'S UNIVERSITY

models and their components. Then describe each diagram and its notations, how they are created, and how they are related to the functional and structural model of the existing system.

3.3 INTERACTIONS

In every active system, objects interacts with each other by passing messages. An interaction is a behaviour of objects, by passing a set of messages among objects, we try to achieve some objectives.

An interaction is a behaviour that contains a set of messages that substitute a set of objects to accomplish the defined objectives. A message is a description of communication between two or more objects that process the information with the expectation of a result.

OBJECT DIAGRAM

If we use an object diagram, then the unlimited number of instances are available in nature and need to be considered its impact on the system.

Following things should be considered and understood clearly during the construction of the object diagram.

- Objects are used to construct the object diagrams.
- Links are used for connecting the objects.
- Objects and links are used to construct an object diagram.
- The object should have a unique identifying name.
- The most significant elements are to be identified.
- The association between objects should be explained.
- All values of distinct instances of objects should be present in the object diagram.
- Comments should be attached as per requirement.

The core elements of interactions are as shown below:

INSTANCE: In UML, every block is represented by the class. It is used to create instances or objects of the class. Every object is associated with the class. As represented in figure 3.2, the patient captures patient information. All the attributes (Name, DOB, Address, Contact No) of the patient and methods (e.g., insert and delete) are represented in figure 3.2. An instance of the aPatient:Patient class (1st Instance: Name=Akash, DOB= 10 April 1980, Address = H.No. 106, South city, Lucknow, India, Contact No= (0091) (9452784588)) is generated.

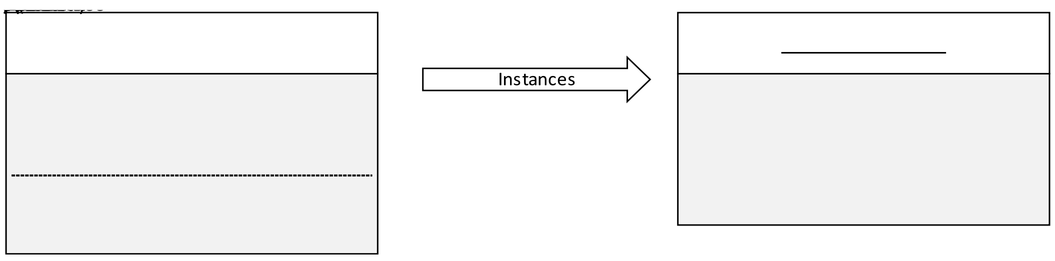


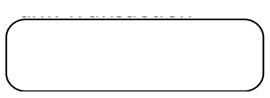
Figure 3.2: Class and Instance

ACTION: It is a named element that represents a movement from one stage to another stage. It represents a single step within the operation. Every action has some input and output. It is represented by round-cornered rectangles. Action is usually

ignou THE PEOPLE'S UNIVERSITY

used as a verb or noun for the action with some description. Some of the action names are given below:

- Fill Order
- Check-In
- Course Enrolment
- Display Error



OPERATION: Features of a class are represented in the form of attributes and operations. Operation is represented in a form of block. The top rows represent the name of the class, the middle row represents the class's attributes, and the third row represents operations of the class. The attributes represent what a class is, and the operations express what a class does. As shown in figure 3.3, Class name Clock, attributes (hour, minute, second), and operation (displayTime).

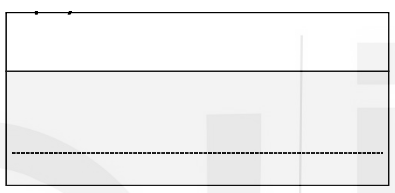
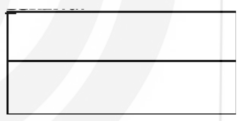
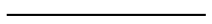


Figure 3.3: Operation

SIGNAL: When an activity performs any operation then it sends and receives messages, which is known as signals. It is an asynchronous event. In the given figure <<CSE201: Seminar>> represent the signal and the parameter of signal is term= “Final”.



LINK: It is used for establishing a relationship between objects and classes of the system. Link is represented by a line. It is a conceptual or physical connection between two objects. For example, Akash studies at the University, explains the connection of a student with a University.



LINK ATTRIBUTES: It assigns the value of a specific instance of the link; for each pair of the object, there is a particular value associated with each attribute. As shown in figure 3.4, "orderNumber" is a link attribute. The system represents that the customer and order placed into the Number, and each order has an orderNumber.

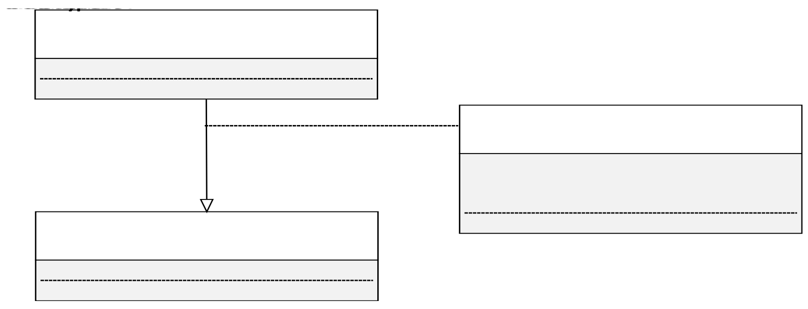


Figure 3.4: Link Attributes

An object is an instance of a class. The object can be represented as the instance of a class, with pathname and attributes. If more than one object is used, it can relate to links.

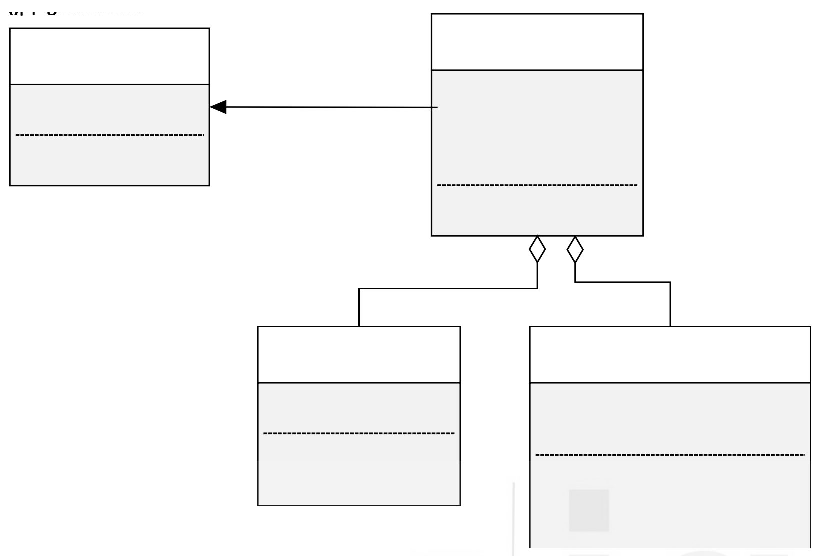


Figure 3.5: Class Diagram

The notation for class, attribute, and operation is shown in figure 3.5. Customer and order are two classes of the class diagram. These classes represent a one-to-many relationship because a customer can issue multiple orders. Customer class have two functions: sendOrder() and receiveOrder() while Order class have two functions: confirm() and close(). Order classes have abstract classes, which are represented as SpecialOrder and NormalOrder classes. The two inherited classes use two additional functions like dispatch() and receive() with all the properties.

In figure 3.6, the object diagram describes the scholar who can go to attend "Seminar" can be "Wait Listed" for attending it, or it may be a "Teaching Assistant" for it. In the object diagram, Alok and Amar are research assistants in CSE 201, while Shruti is teaching assistance "TA" for CSE 201, Shreya is research assistance "RA", and Neha is "Wait Listed". Object diagram also describes that the Seminar is the instance of the Course Research Methodology "RM" class. Here, the object diagram generates the three relationships between seminars and research assistants and the two relationships between the course and seminars.

ignou THE PEOPLE'S UNIVERSITY

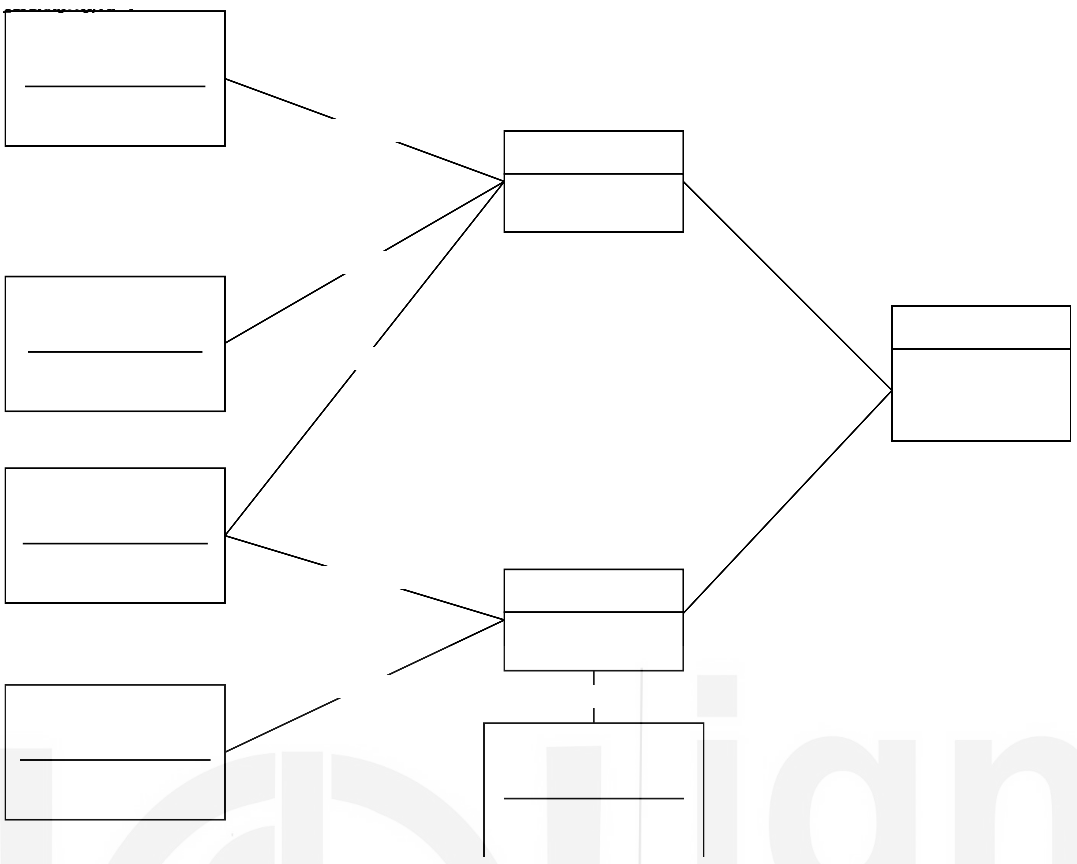


Figure 3.6: Object Diagram

Check Your Progress 1

1) Behavioural model describes:

   a) External parameter

   b) Internal parameter

   c) Both (a) and (b)

   d) None of the above

2) Every action has.........

   a) Input

   b) Output

   c) Input and output

   d) None of the above

3) Signals perform any operation by..........

   a) Sending messages

   b) Receiving messages

   c) Sending and receiving messages

   d) None of the above

ignou THE PEOPLE'S UNIVERSITY

4) Link is used to establish the relation between..................

   a) Attributes and Class
   b) Objects and Attributes
   c) Objects and Class
   d) None of the above

3.4 USE-CASE AND USE-CASE DIAGRAM

Objectives of applying Object-Oriented principles in systems modelling add flexibility so that system development can be easily maintained and provide reusability. The model of a system behaves static and dynamic. Dynamic behavior measures the performance of the system at the time of the execution. A Use Case diagram represent the dynamic behavior of the system. Use Case diagram does not describe what a system does to accomp lish a particular goal of the customer. It is a sequence of actions, in various iterations during the performance of the system. It is represented in a form of an ellipse.

In UML, the dynamic behavior of the system can be measured in various ways. A use-case diagram is used to represent internal or external factors to make the system more impressive. Use-case diagram contains actors, use-cases, and their relationships. It is used to model the system of an application. A single Use-Case diagram is used to represent a particular functionality of a system. Every Use-Case has a unique name that discriminates it from other use-cases. To model the complete system, use-case diagrams are used.

OBJECTIVES OF USE-CASE DIAGRAM

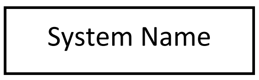
Use-case diagram is used to express the dynamic expressions of a system. It is used to determine the internal and external impact of system requirements. These specifications are based on design requirements. Therefore, when a system is required to collect its functionalities, use-cases are prepared, and actors are used. The objective of the use-case diagram can be described as follows:

- To identify the requirement of a system.
- To get the external perspective of a system.
- To identify the internal and external factors that influence the system.
- Show the relationship between actors.
- It also describes the flow of the event.

ELEMENTS OF USE CASE DIAGRAM

It is used to capture the business processing of a system. It is of two types: one is used for representing the business roles, and the other is used for business processing. The basic elements which constitute a Use Case diagram are as follows:

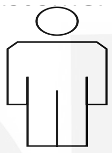
SYSTEM: A system boundary defines the scope of a system. Rectangles are used to draw system boundaries that contain a use-case, and actors are used outside the system boundaries. The system can work only with limited functionalities. Use-case needs to have appropriately defined limits.



ignou THE PEOPLE'S UNIVERSITY

USE CASE: It is used to describe a part of the system's functionality that enables access to the functionalities. It is represented using the ovals. Label the ovals with verbs that represent the system's function. The objective behind use-cases is to describe interactions. The flow of batch processing, which generally does not include interactions, can also be used as a use-case.

ACTORS: The users of a system perform the role of actors in the use-case diagram. An actor performs as an entity with fixed roles in each system. In the situation when one system is the actor of another system, label the actor system with the actor image. The system needs to manage the role of a person or entity which is working as an actor. For example, actors may have various roles like administrator or user. In each case, the appropriate functionalities of the Use Case are granted. All the actors are not part of the system. A person who performs services to the counter is also known as the actor. For example, for modeling a hospital application, a doctor and patient entity represent an actor in the application. Typically, an actor plays a role that a human, a hardware device, or maybe another system plays with the system.



ignou THE PEOPLE'S UNIVERSITY

ROLE OF ACTORS: An actor is a user or external system that moves a system. The role of an actor is equated with determining the behaviour of the associated system—for example, different types of actors perform the activity at the Bank ATM. Bank customers and card holders are the main actors, while customerAs and cardAs participate as a sub-actor of the system.

There are various actors associated with the ATM system, which are as follows:

- **A Bank Customer** can withdraw money, check the balance, deposit cash, and deposit cheque from the ATM.
- **A Card Holder** can use a credit card and debit card to withdraw the money from the ATM.
- **A customerIs** can access the ATM for various types of ATM transactions.
- **A cardAs** is responsible for credit card and debit card to withdraw the amount from the ATM.

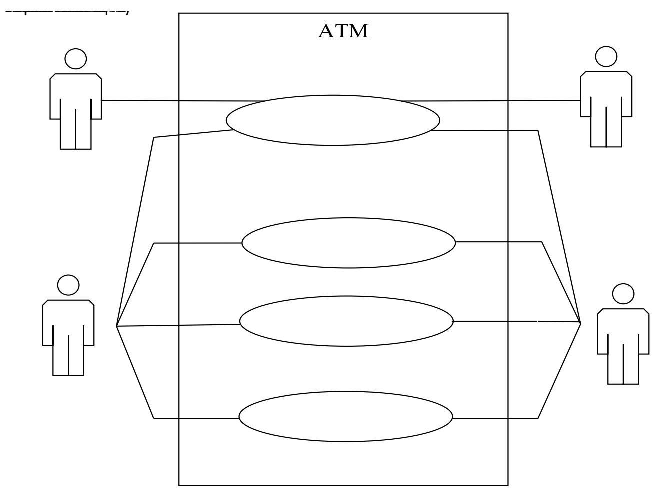


Figure 3.7: Role of Actors in ATM Application

HOW TO DRAW A USE CASE DIAGRAM?

Use Case diagrams are used to define the functionalities of a system. When the diagrams are considered for high-level analysis, objects, relationships, and their guidelines should be clearly defined. Actors can perform as a user, or they may perform as internal or external applications. We should consider the following items when drawing a use-case diagram.

- Identify all the Use Cases.
- Identify all the actors.
- Generalize the relationship between Use Cases and actors.

Following guidelines should be followed to draw a Use Case diagram.

- Use Case is used to identify the requirements.
- Use Case name reflects the functionalities of the object.
- We need to have appropriate names for actors.
- Relationships and dependencies are specified in the Use Case diagram.
- Name should be defined in such a manner that it reflects the functionalities it performs.
- Relationships and dependencies are clearly defined in the Use Case diagram.

Figure 3.8: Use Case Diagram of Order Management System

In figure 3.8, the Use Case diagram represents the Order management system. In this diagram, three Use Cases Food Order, LunchOrder, and BreakfastOrder and one actor in the role of user or customer are there. The LunchOrder and BreakfastOrder are the extensions of the Use Case Order. Therefore, they have extended relationships. In the given diagram, the actor Customer lies outside the system as it is an external user of the system.

DEPENDENCIES

Every model consists of many Use Cases. These Use Cases are used to define the different functionalities of a system. There are two types of dependencies known as *include and extended* dependencies, which are used in Use-Case. Dependencies information is used by the Use Case to execute a system functionalities. How these

ignou THE PEOPLE'S UNIVERSITY

Use Cases are related to one another is also defined using dependencies. These dependencies are defined as follows:

INCLUDE DEPENDENCIES

Specialized types of dependencies are known as include dependencies. There are different possibilities to log the activities of a project management system. These possibilities may be as project managers, resource managers, and system administrators to interact with the system. Figure 3.9 elaborates the Use Case, while the second figure defines the activities of the project manager, resource manager, and system administrators as they log into the system and they are using the Use Cases of the diagram.

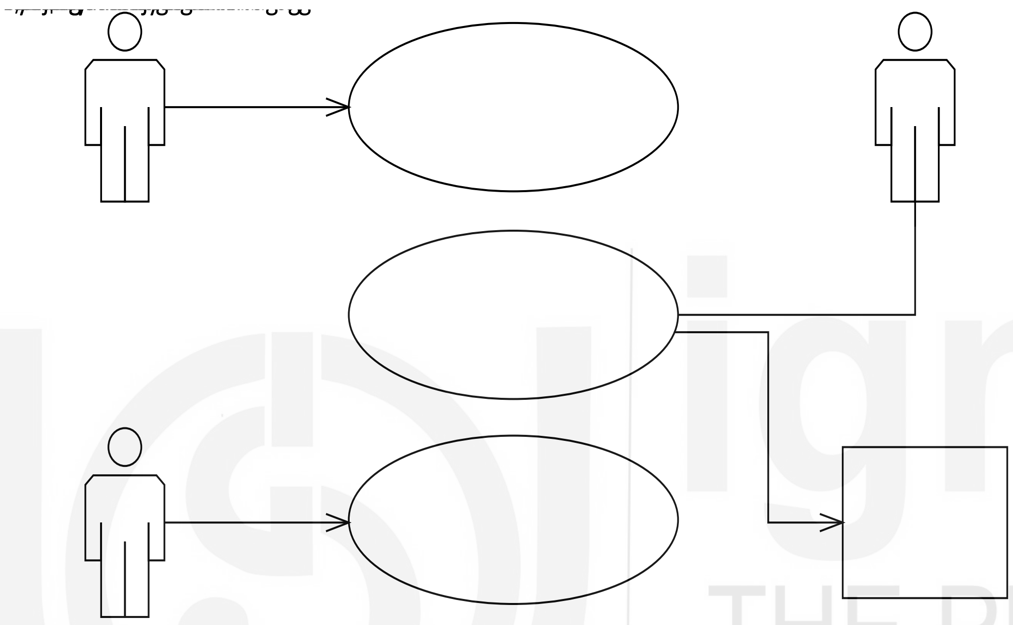


Figure 3.9: Use Cases with Common Behaviour

Include dependency is used to make the relationship between one Use Case (called a base case) to another Use Case (called an inclusion Use Case). The base Use Case is called the inclusion Use Case. It is shown by the dashed arrow line from the base Use Case to the inclusion case, as shown in figure 3.10. The base Use Case monitors the behaviour sequence of the step to include the inclusion Use Case.

ignou THE PEOPLE'S UNIVERSITY

[illegible]

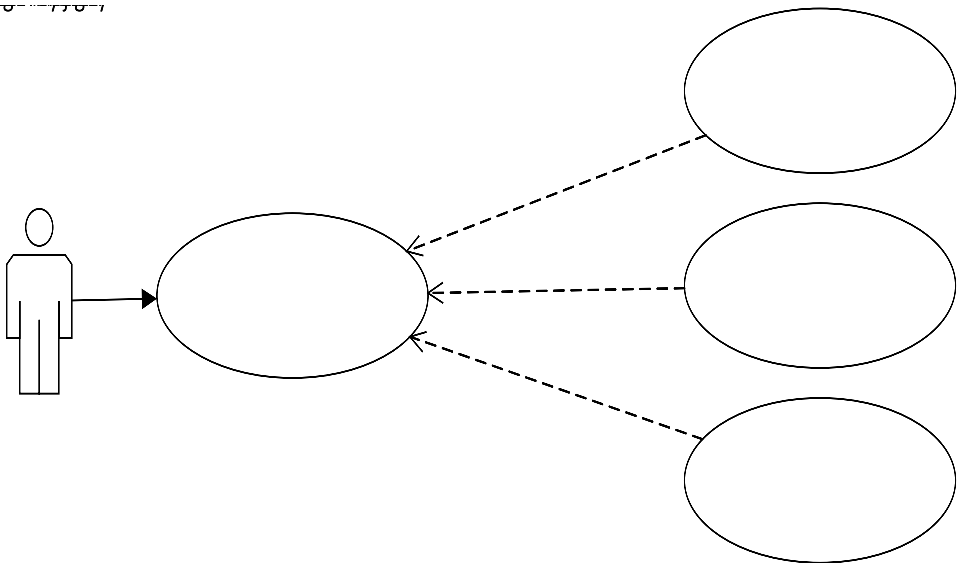


Figure 3.10: Include Dependency

Include dependency behaviour is the same as a Use Case. It is common to other Use Cases and added as different Use Cases which can be reused in other Use Cases. In figure 3.9, the log activity of the Use Case is included in the Manage Project, Manage Resource, and Administer System Use Cases.

EXTEND DEPENDENCIES

In figure 3.11, the Use Case with optional behaviour is made with various activities and tasks. Various activities are performed during the execution of the project. The Project Manager manages projects by maintaining the system activities and its tasks. It also addresses the situations by adding the behaviour of Use Cases.

Figure 3.11: Use Case with optional Behaviour

An extended dependency extends the Use Case from one Use Case (known as the extension Use Case) to another Use Case (known as base Use Case). A single Use Case may cover multiple Use Cases or multiple Use Cases covered by the single Use Case by using the extended keyword. The base Use Case detects the behaviour sequence of the extending Use Case added in the project. In figure 3.12, extend line is denoted by the dashed arrow line to extend the above Use Cases.

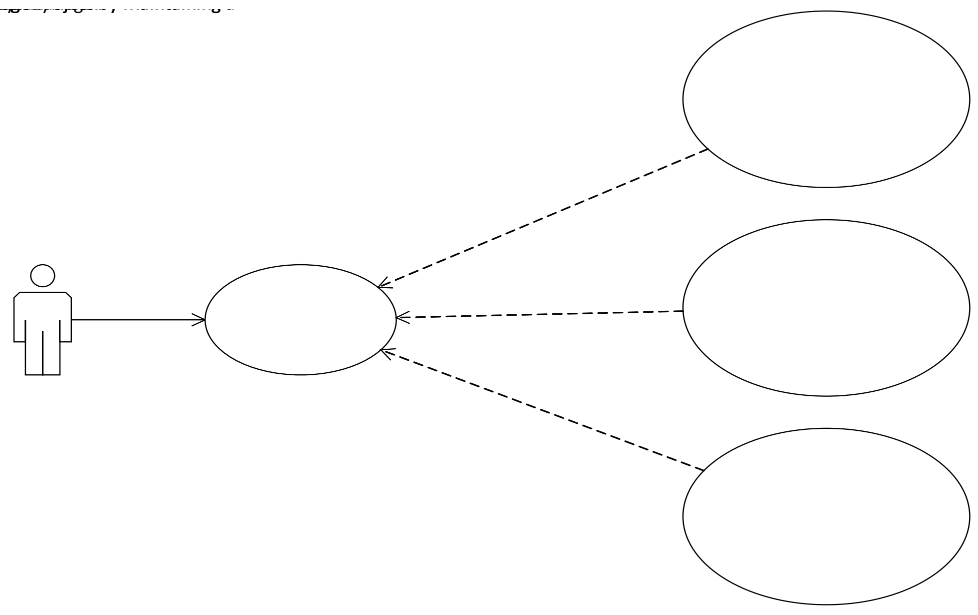


Figure 3.12: Extend Dependencies

If we insert another behaviour sequence in a Use Case at any location, then it is known as an extension point. Extension point is represented inside the compartment with an extension point followed by a colon with a proper description of the location of the extension point in the Use Case behaviour sequence. A Use Case can have more than one extension point. This allows for different source Use Cases to extend this target Use Case.

WHERE TO USE A USE CASE DIAGRAM?

Different types of diagrams are used to model the dynamic behaviour of a system in UML. Each model has its specific purpose. Use Case diagram is used to collect the system requirement and actors.

It specifies the flows and events of a system. But remember that Use Case never describes the details of the system, such as how behaviour is implemented. All the input, output, and function are mentioned inside the black box. This information is very useful for high-level design to get a complete picture of the system. Use Case diagrams can be used for:

- To analyze the requirement analysis of a system.
- To prepare the model of a system.
- Requirement analysis and high-level design.
- Forward and reverse engineering.
- Modelling the basic idea behind the system.

CHECK YOUR PROGRESS 2

1) A Use Case diagram is a ………… behaviour.

   a) Static

   b) Dynamic

   c) Both (a) and (b)

   d) None of them

2) Which element is not part of the Use Case diagram?

ignou THE PEOPLE'S UNIVERSITY

a) System
b) Actor
c) Use Case
d) Class

3) An Actor is a part of …………………… system.

a) Internal
b) External
c) Both (a) and (b)
d) None of them

4) Which is not a type of dependencies ………………………..

a) Include
b) Exclude
c) Extend
d) Remove

5) Draw an Automated Trading Housing System for Use Case?

………………………………………………………………………………………
……..……………………………………………………………………………
…………………………...

3.5 INTERACTION DIAGRAM

An interaction diagram contains a set of objects and their relationships, including the messages with them. It is used to model the dynamic behaviour of the system. It includes sequence diagram and collaboration diagram. The primary difference between class diagrams and interaction diagrams is that class diagrams describe the structure while the others describe the behaviour. The objective of modelling is used to represent the class diagram is at the class level, whereas the interaction diagrams represents the object level.

OBJECTIVES OF INTERACTION DIAGRAM

The interaction diagram is used to visualize the interactive behaviour of the system. It represents how one or more objects in the system connect and communicate with each other.

Interaction diagram focus on the dynamic behaviour of the system. It provides us with the dynamic context of an interaction between one or more lifelines of the system.

In UML, the interaction diagram is used for various purposes, which are as follows:

- It is used to monitor the dynamic behaviour of a system.
- It represents the structural aspects of various objects in the system.
- It represents the sequence of interactions within a system.
- It provides the means of visualizing the real-time data.
- It can be used to explain the architecture of an object-oriented or distributed system.

- It shows the communication and sequence of messages passing in the system.

Some of the important points which are to be identified before drawing the interaction diagram are as follows:

- Participation of objects in the interaction.
- The flow of messages among the objects.
- Messages flowing sequence.
- Objects are defined in an organized form.

Sequence diagram and collaboration are used to represent the interaction diagram which is discussed here:

3.5.1 SEQUENCE DIAGRAM

A sequence diagram describes the sequence order of interaction between objects. It also describes how performance is accomplished of the objects in a system. It flows from left to right. These diagrams focused on time and show the order of the interaction by applying the vertical axis of the diagram to represent time, and what messages are sent and when. The sequence diagram focused on a time sequence of messages, while the collaboration diagram focused on the structural organization of the objects that send and receive messages.

In figure 3.13, four sequence diagrams (Customer, FoodOrder, LunchOrder, and BreakfastOrder) are represented. These diagrams show the message sequence for LunchOrder and the BreakfastOrder. The sequence diagram is used to understand the time sequence of message flows from one object to another. Methods are invoked of an object and flow messages from one object to another. The first method is invoked as sendOrder() from Order object while the next call is confirm() is a method of specialOrder object. It describes the method calls from one object to another, and this is also the actual sequence when the system is running.

Figure 3.13: Sequence diagram of Order Management System

3.5.2 COLLABORATION DIAGRAM

A collaboration diagram represents elements as they interact over time and how they are related as shown in figure 3.14 in the form of object organization. In the collaboration diagram, the method call sequence is indicated by some numeric technique. The number indicates how the methods are called one after another. We can use the same order management system of sequence diagram to describe the collaboration diagram.

Sequence diagrams are similar to method calls which do not describe the object organization, whereas the collaboration diagram shows the object diagram.

Two diagrams are used for the selection of the requirement. If the time sequence is preferred, then the sequence diagram is used, and when the organization is required, then a collaboration diagram may be preferred.

Figure 3.14: Collaboration Diagram of Order Management System

ignou THE PEOPLE'S UNIVERSITY

CHECK YOUR PROGRESS 3

1) Interaction diagrams consist of______
   a) Objects and Relationship
   b) Object and Attributes
   c) Objects and Class
   d) Attributes and Class
2) What is a Sequence Diagram?
   a) Objects
   b) Methods
   c) Sequence
   d) Relationships
3) What is Collaboration Diagram?
   a) Object Diagram
   b) Class Diagram
   c) Method Diagram
   d) None of them

4) Draw a collaboration diagram for Student Management System.

   …………………………………………………………………………………………
   ……………………………………………………………………………

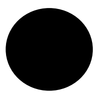
ignou THE PEOPLE'S UNIVERSITY

3.6 ACTIVITY DIAGRAM

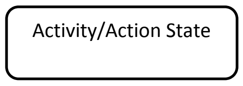
The activity diagram is one of the important diagram used for modelling dynamic aspects of the system. It shows the flow from one activity to another activity. It is an ongoing process within a state machine. Therefore, these processes may be performed as sequential, branched, or concurrent. It deals with various types of flow control by using various control nodes. This diagram consists of activities, links, relationships, etc.

Activity diagrams are the same as previous diagrams. It captures the dynamic behaviour of the system. An activity diagram is used to express message flow from one activity to another in a flow chart. It is a diagram that looks like a flow chart, but it is not the same. It shows different flows like parallel, branched, single, and concurrent. An activity diagram consists of the following behavioural elements:

Initial State/Start Point: It represents an activity diagram's initial state or start point. It is denoted by a small, filled circle. The notation for the initial state is given below.

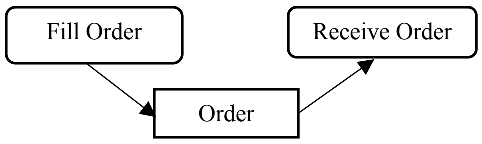


Activity or Action State: It represents the uninterruptible action of objects. It is represented by a rectangle with rounded corners or edges.

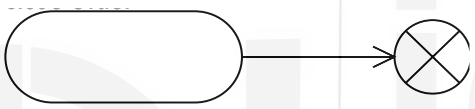


Action flow or Control flows: Action flow shows the edges and paths from one action state to another action. It is represented by an arrowed line.

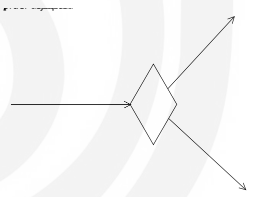
Object Flow: It is used for the creation and modification of objects. An object flow arrow shows an object actions, including the transition from one active state/object to the other object.



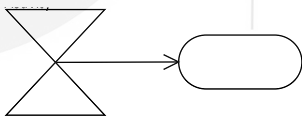
Flow Final Node: It shows a circle with a cross inside. It shows the end of a single control flow. The final activity node denotes the end of all control flows within the activity.



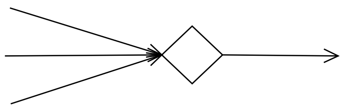
Decision: It is represented by a decision with alternate paths. It is like a flowchart where the decision is to be made by diamond, with the options written on the sides of the arrows issuing from the diamond.



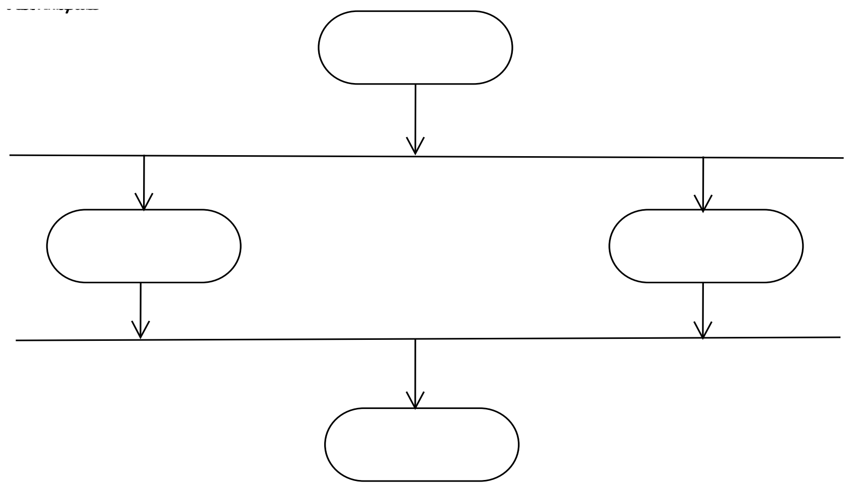
Time Event: It refers to an event that stops the flow for a time.



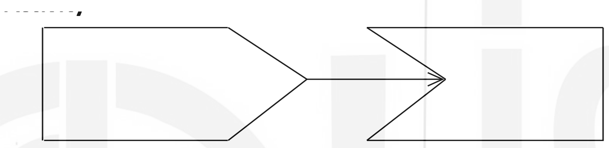
Merge Event: It is used to merge multiple flows that are not concurrent.



Synchronization: A fork node is used to split an incoming flow into multiple simultaneous flows in concurrent flow management process. It is represented by a straight, slimly thicker line in the activity diagram. A fork and join node used together are often referred to as synchronization.



Signals: Signals are used to represent an activity to send and receive a message. It is of two types: Input signal and Output signal. The input signal is represented by a concave polygon (Message receiving) and the output signal is represented by a convex polygon (message sending). The state of the signal can not be changed until a response is received.

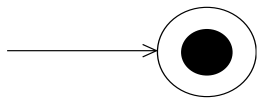


Swimlanes: It is a way of representation for activities performed by the individual actor or group activities in a single thread, as shown in figure 3.15

Figure 3.15: Activity Diagram- Swimlanes

ignou THE PEOPLE'S UNIVERSITY

Final State/End State: An arrow is used to indicate the circle, and the inside filled circle shows the end state or final state of the diagram.



The activity diagram shows the flowchart of activities. It shows the workflow between various model activities. It looks like flowcharts, but they are an advancement of the flowchart with some additional capabilities. Now let us see how to draw activity diagram.

HOW TO DRAW AN ACTIVITY DIAGRAM?

To draw an activity diagram, try to understand the elements used in the activity diagram. The user must know the main elements of an activity diagram. It is a function performed by the user. We need to understand how the activities are associated with constraints and conditions. If there are any constraints, they should be distinguished before developing an activity diagram. Before drawing an activity diagram, we should identify the following items:

- Identify all the activities in the system.
- Identify all the actors in the system.
- Find all the flows among the activities.
- Identify all the constraints in the activity diagram.
- Refine complicated activities and nested activities of the diagram.

Once all the above items are identified, imagine the complete structure of the entire system. This structure is then transformed into an activity diagram.

Activity diagrams show the flow from a start point to the endpoint with various decision paths that exist in the system. Figure 3.16, shows the activity diagram for the order management system. In this diagram, four main activities are identified which are associated with conditions. It represents the various situations of parallel execution of some activities. It is very useful for business modelling to explain the detailed modelling activity of the entire diagram. It is used to represent the decision paths, but the programmer's code and activity diagram does not exactly match. Figure 3.16, shows the four main activities, which are as follows:

- Send order by the customer
- Receipt of the order
- Confirm the order
- Dispatch the order

ignou THE PEOPLE'S UNIVERSITY

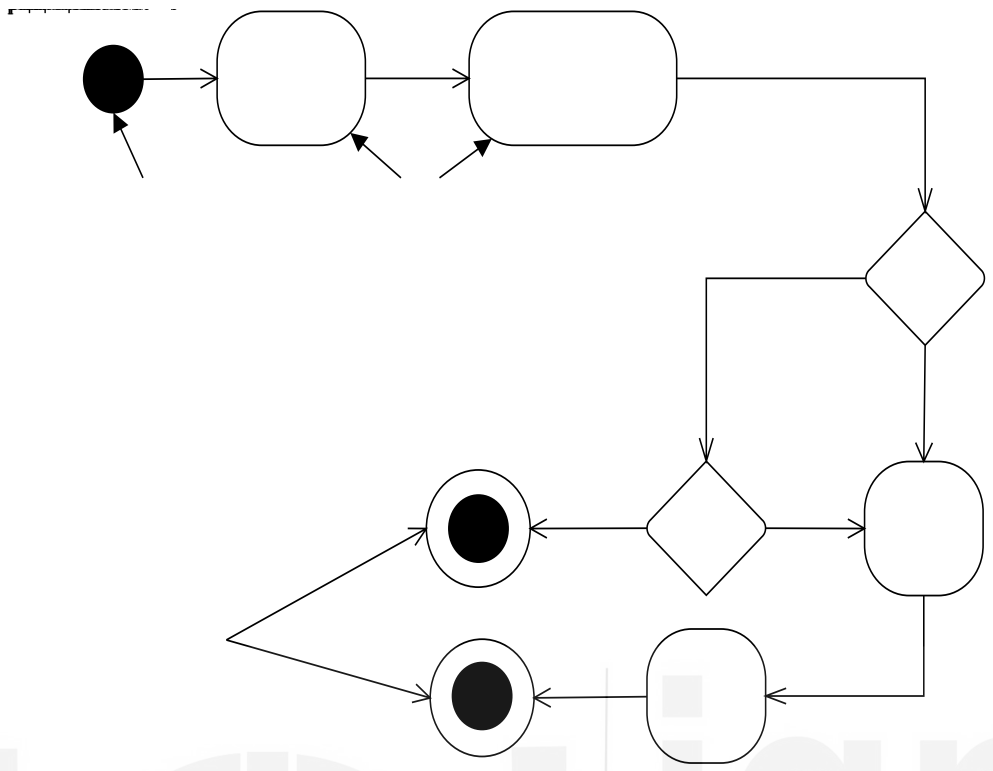


Figure 3.16: Activity Diagram of Order Management System

In the above diagram, the customer sends an order request, order request confirm for the receipt of the order, verify the conditions of the order, if it is a Lunch or Breakfast order. After verifying the condition of the order, activity to confirm the order is performed, dispatch the order, and then finally the end of the activity will be performed as the termination of the process.

WHERE TO USE ACTIVITY DIAGRAMS?

The activity diagram is used the same as other UML diagrams. The main difference between the activity diagram and other UML diagrams is to model the control flow from one activity to another activity. It does not show the messages in the diagram. This diagram is suitable for modeling the activity flow of the system. These diagrams describe the flow from one system to another. From the above descriptions, the activity diagram is used for high level-designing. Therefore, it gives a high-level performance of a system. It is mainly used for designing business activities or other real-life problems and solutions.

This diagram is implemented for business demands to model the various activities of the system. It has focused on business understanding rather than on implementation details. Activity diagram can be used for:

- To define the workflow of the model.
- To define the business requirements model.
- For understanding the system's functionalities.
- To explore the business requirements at a later stage.

3.7 CRUDE ANALYSIS

It is one of the useful techniques to identify the basic objects in the problem domain. The CRUDE analysis is used to perform various activities of an object in the project domain and collaborate in support of Use Case analysis. CRUDE matrix is used to represent each interaction among objects is labelled for the various purpose, which is as follows:

- Create - to create and store new data.
- Read - to retrieve and read data.
- Update - to change or modify then store the data.
- Delete - to delete or remove the data.
- Execute - to execute the data.

It is a way of representation to capture various activities and permissions within a system. It is an object-oriented approach in which a class-by-class matrix is used. Each cell in the matrix represents the interaction between instances of classes. It combines with a CRUDE Matrix with the analysis of user processes within the system, especially in the context of the actors and roles involved to complete the picture. The analysis helps to identify the usage of entities related to GUIs. Once a CRUDE matrix is completed for the entire system, the matrix can be scanned to ensure that every class can be instantiated. The table below shows a publishing service model to capture the information. In this case, the actors are Manager, Writer, Editor, and Publish. This model supports all CRUDE (Create, Read, Update, Delete and Execute) instances. Especially, a manager can create and delete instances but cannot read, update, and execute. The editor can read and update instances of a CRUDE matrix.

Table 3.1: Publishing Service CRUDE MATRIX

Action Role	**CREATE**	**READ**	**UPDATE**	**DELETE**	**EXECUTE**
Manager	X			X	
Writer	X	X	X	X	
Editor		X	X		
Publish		X	X		X

CHECK YOUR PROGRESS 4:

1) Activity diagram is used to mode____________________

 a) Static Behaviour

 b) Dynamic Behaviour

 c) Both (a) and (b)

 d) None of the above

2) The activity diagram, initial/start state can be shown by__________

 a) Circle

 b) Filled Circle

 c) Double Circle

 d) Half Circle

3) Activities define the ……………... of the model.

ignou THE PEOPLE'S UNIVERSITY

a) Business flow

b) Control Flow

c) Workflow

d) All the above

4) The CRUDE analysis does not perform the activities_____________

a) Create

b) Delete

c) Report

d) Update

5) Draw an activity diagram for email processing.

…………………………………………………………………………………………
………………………………………………………………………………………
…………………………..

3.8 SUMMARY

Modelling is used for the developing application/services of the software. Different types of objects are used in modelling. Every object behaves in different ways. The behaviour model describes the internal parameter of an information system. Model of a system behaves either static or dynamic. A Use Case diagram is a dynamic behaviour in the system to represent the dynamic expressions of the system. Use Case diagram is explained in detail in this unit. An interaction diagram is used to represent the dynamic behaviour, and it consists of objects, relationships and messages. The activity diagram represents the activities of dynamic behaviour. To capture the various activities of objects and permissions within a system is represented by the CRUDE matrix.

This unit provides the basic concepts of behavioural modelling notations. Prime diagrams of the behavioural model are presented and discussed. A design has been provided about various perspectives and their corresponding diagram. This unit also represents various elements used in behavior diagrams.

3.9 SOLUTIONS/ ANSWER TO CHECK YOUR PROGRESS

Check your Progress 1:

1. b
2. c
3. c
4. c

Check your Progress 2:

1. b
2. d
3. b
4. d
5. Use Case for Automated Trading House System

Figure 3.17: USE CASE for Automated Trading System

Check Your Progress 3:

1. a
2. c
3. a

4. Collaboration diagram for Student Management System

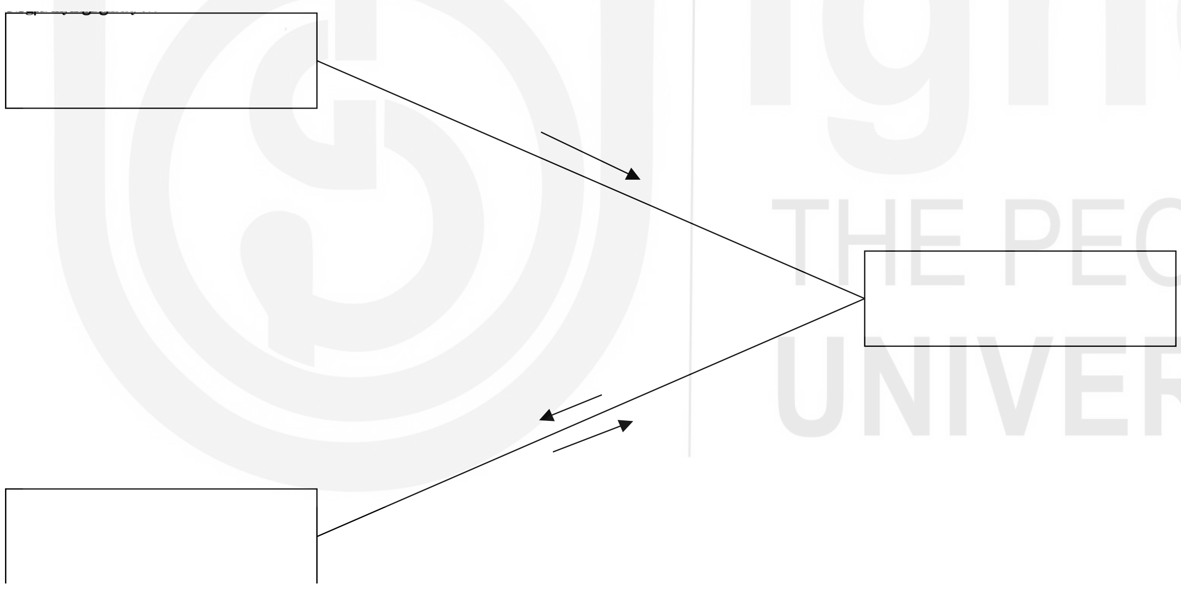


Figure 3.18: Collaboration Diagram for Student Management System

Check your Progress 4:

1. b
2. b
3. d
4. d
5. Activity diagram for Email Processing

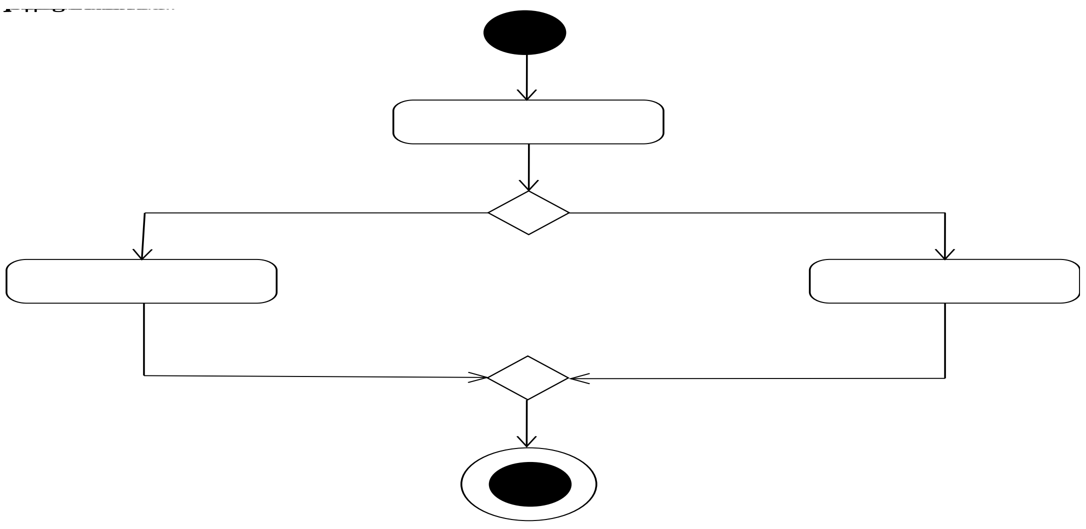


Figure 3.19: Activity Diagram of Email Processing

3.10 REFERENCE / FURTHER READING

- Grady Booch, James Rumbaugh and Ivar Jacobson,“ The Unified Modeling Language User Guide”, 2nd Edition, Addison-Wesley Object Technology Series, 2005.
- Grady Booch, Robert A. Maksimchuk, Michael W. Engle, Bobbi J. Young, Jim Conallen, Kelli A. Houston, “Object-Oriented Analysis and Design with Applications,” 3rd Edition, Addison-Wesley, 2007.
- James Rumbaugh, Ivar Jacobson, and Grady Booch, “Unified Modeling Language Reference Manual,” 2nd Edition, Addison-Wesley Professional, 2004.
- John W. Satzinger, Robert B. Jackson, and Stephen D. Burd, “Object-oriented analysis and design with the Unified process,” 1st Edition, Cengage Learning India, 2007.
- Brett McLaughlin, Gary Pollice,and Dave West, “Head First Object-Oriented Analysis and Design: A Brain Friendly Guide to OOA&D,” Shroff Publisher,First edition,2006.

ignou THE PEOPLE'S UNIVERSITY