Software Engineering-BSCE-301L

LAB DA-4:

Sequence, Collaboration, Activity Diagram

Dr . Saurabh Agrawal

Faculty Id: 20165

School of Computer Science and Engineering

VIT, Vellore-632014

Tamil Nadu, India

CASE TOOLs – Sequence Diagram

❑The sequence diagram represents the flow of messages in the system and is also termed as an event diagram.

❑It helps in envisioning several dynamic scenarios.

❑It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time.

❑In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page.

❑It incorporates the iterations as well as branching.

CASE TOOLs – Sequence Diagram

❑Purpose of a Sequence Diagram

To model high-level interaction among active objects within a system.

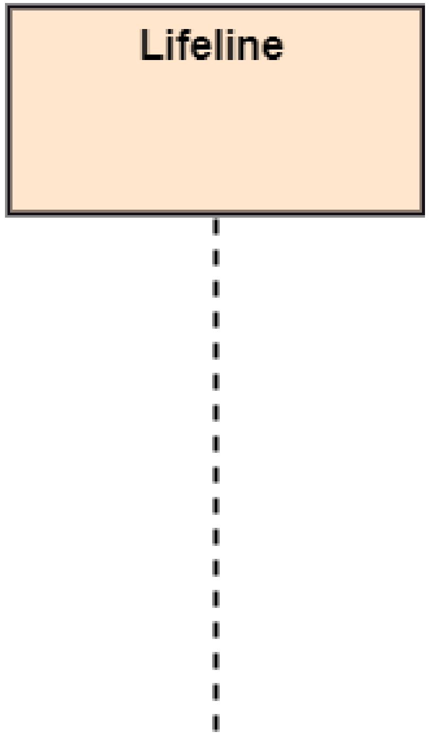
To model interaction among objects inside a collaboration realizing a use case.

It either models generic interactions or some certain instances of interaction.

CASE TOOLs – Sequence Diagram

❑Notations of a Sequence Diagram:

❑**Lifeline:** An individual participant in the sequence diagram is represented by a lifeline. It is positioned at the top of the diagram.



CASE TOOLs – Sequence Diagram

❑Notations of a Sequence Diagram:

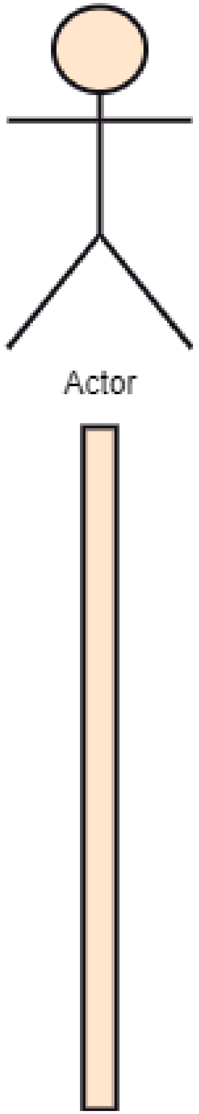
❑**Actor** : A role played by an entity that interacts with the subject is called as an actor.

❑ It is out of the scope of the system.

❑It represents the role, which involves human users and external hardware or subjects.

❑An actor may or may not represent a physical entity, but it purely depicts the role of an entity.

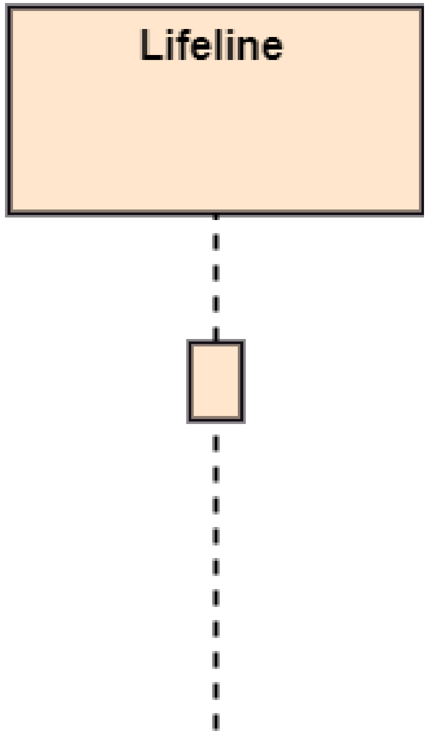
❑Several distinct roles can be played by an actor or vice versa.



CASE TOOLs – Sequence Diagram

❑Notations of a Sequence Diagram:

❑**Activation:** It is represented by a thin rectangle on the lifeline. It describes that time period in which an operation is performed by an element, such that the top and the bottom of the rectangle is associated with the initiation and the completion time, each respectively.



CASE TOOLs – Sequence Diagram

❑Notations of a Sequence Diagram:

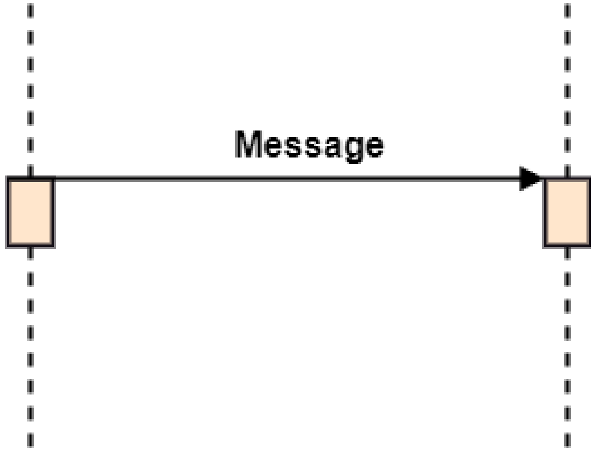
❑**Messages:** The messages depict the interaction between the objects and are represented by arrows. They are in the sequential order on the lifeline. The core of the sequence diagram is formed by messages and lifelines.

CASE TOOLs – Sequence Diagram

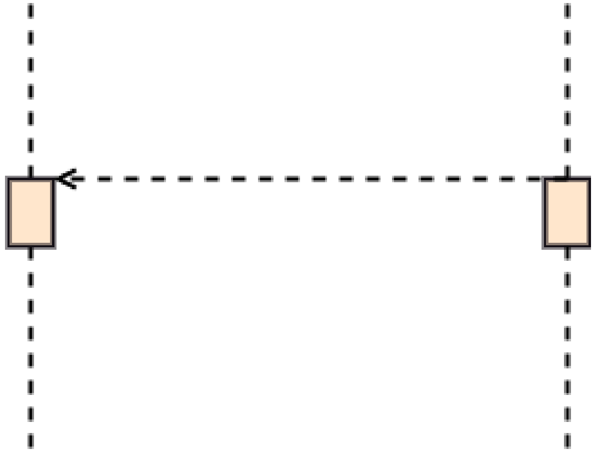
❑Notations of a Sequence Diagram:

❑Messages:

- **Call Message:** It defines a particular communication between the lifelines of an interaction, which represents that the target lifeline has invoked an operation.



- **Return Message:** It defines a particular communication between the lifelines of interaction that represent the flow of information from the receiver of the corresponding caller message.

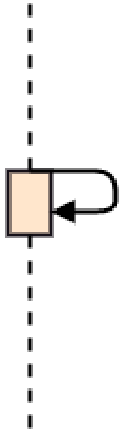


CASE TOOLs – Sequence Diagram

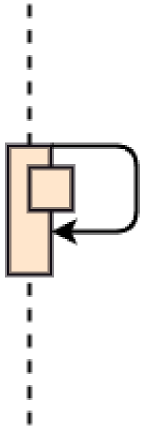
❑Notations of a Sequence Diagram:

❑Messages:

- **Self Message:** It describes a communication, particularly between the lifelines of an interaction that represents a message of the same lifeline, has been invoked.



- **Recursive Message:** A self message sent for recursive purpose is called a recursive message. In other words, it can be said that the recursive message is a special case of the self message as it represents the recursive calls.

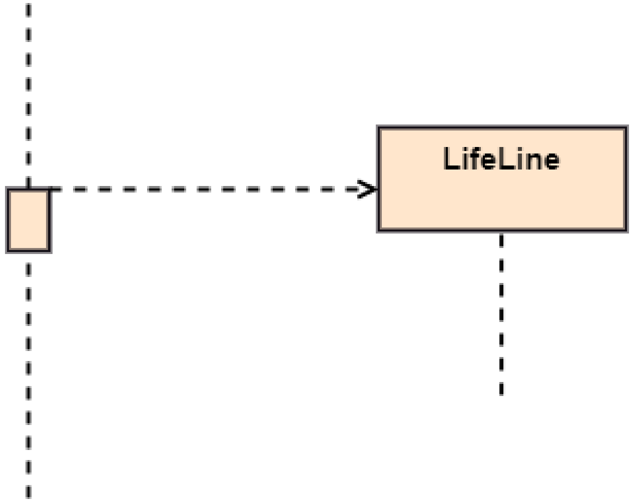


CASE TOOLs – Sequence Diagram

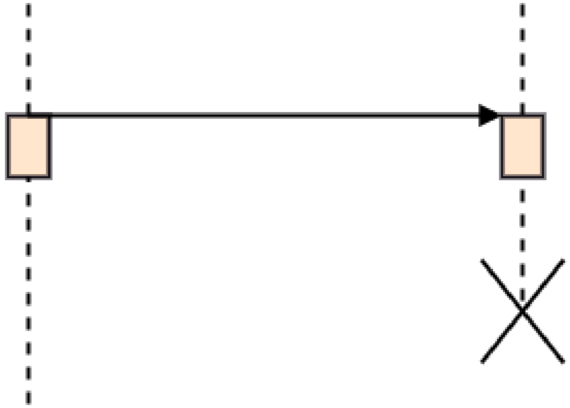
❑Notations of a Sequence Diagram:

❑Messages:

- **Create Message:** It describes a communication, particularly between the lifelines of an interaction describing that the target (lifeline) has been instantiated.



- **Destroy Message:** It describes a communication, particularly between the lifelines of an interaction that depicts a request to destroy the lifecycle of the target.

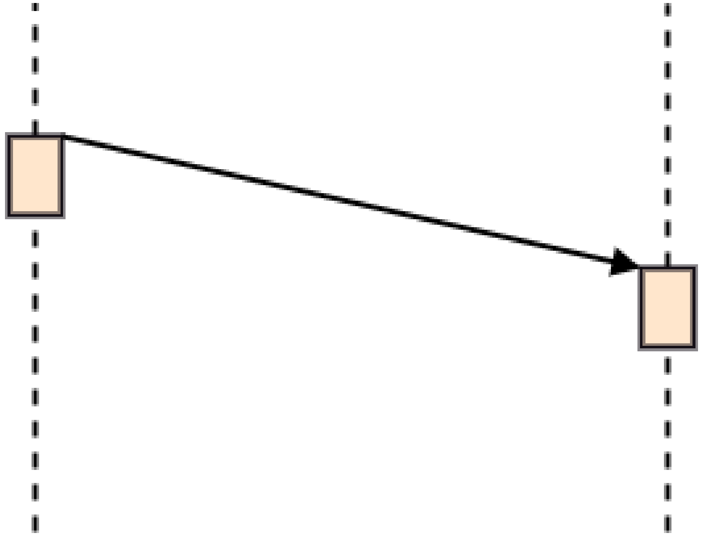


CASE TOOLs – Sequence Diagram

❑Notations of a Sequence Diagram:

❑Messages:

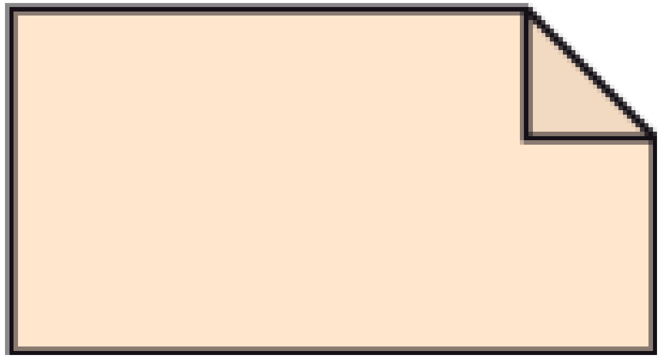
- **Duration Message:** It describes a communication particularly between the lifelines of an interaction, which portrays the time passage of the message while modeling a system.



CASE TOOLs – Sequence Diagram

❑Notations of a Sequence Diagram:

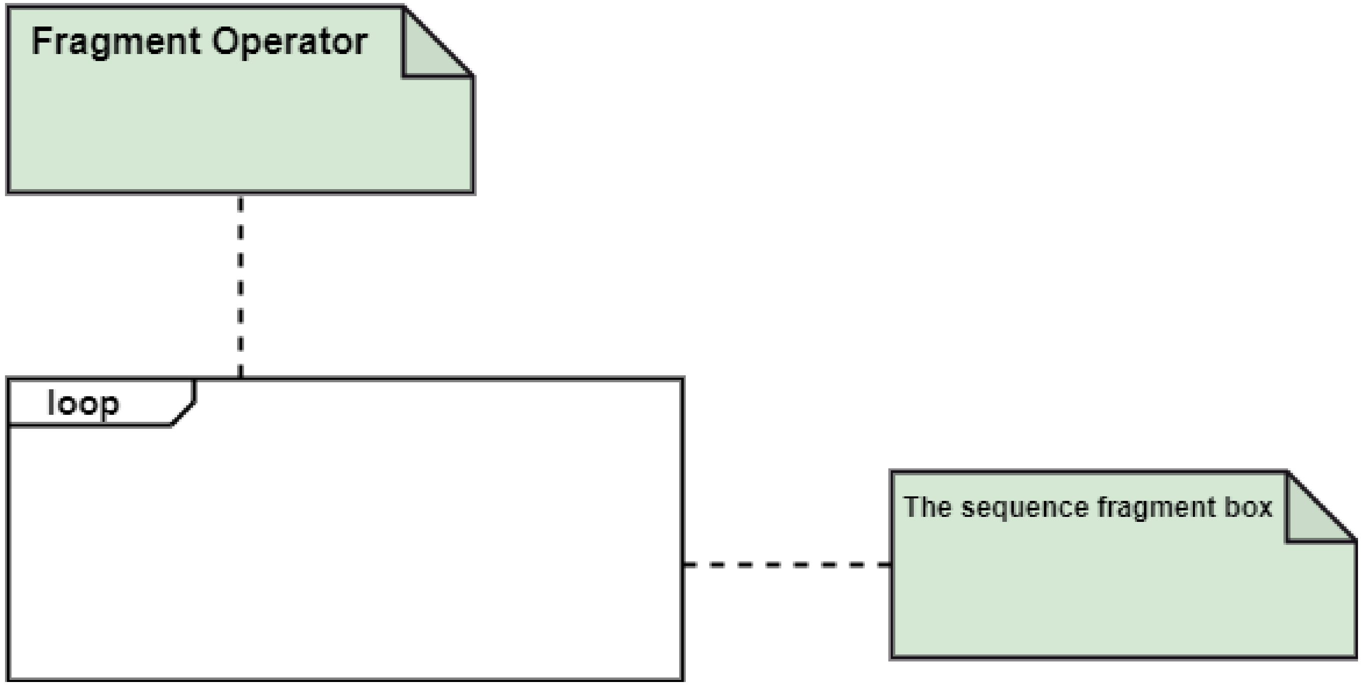
❑**Note:** A note is the capability of attaching several remarks to the element. It basically carries useful information for the modelers.



CASE TOOLs – Sequence Diagram

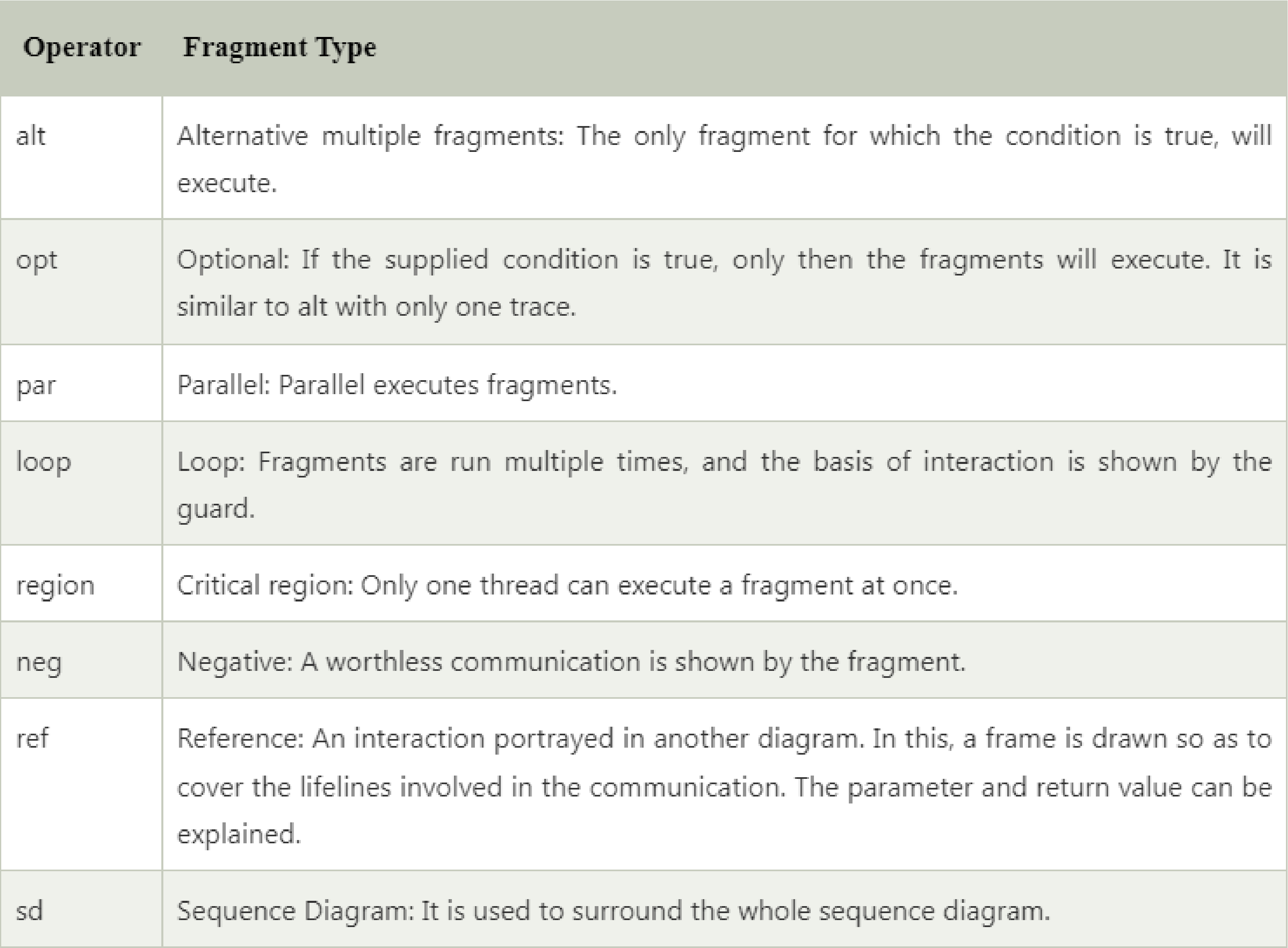
❑Sequence Fragments:

1. Sequence fragments have been introduced by UML 2.0, which makes it quite easy for the creation and maintenance of an accurate sequence diagram.
2. It is represented by a box called a combined fragment, encloses a part of interaction inside a sequence diagram.
3. The type of fragment is shown by a fragment operator.



CASE TOOLs – Sequence Diagram

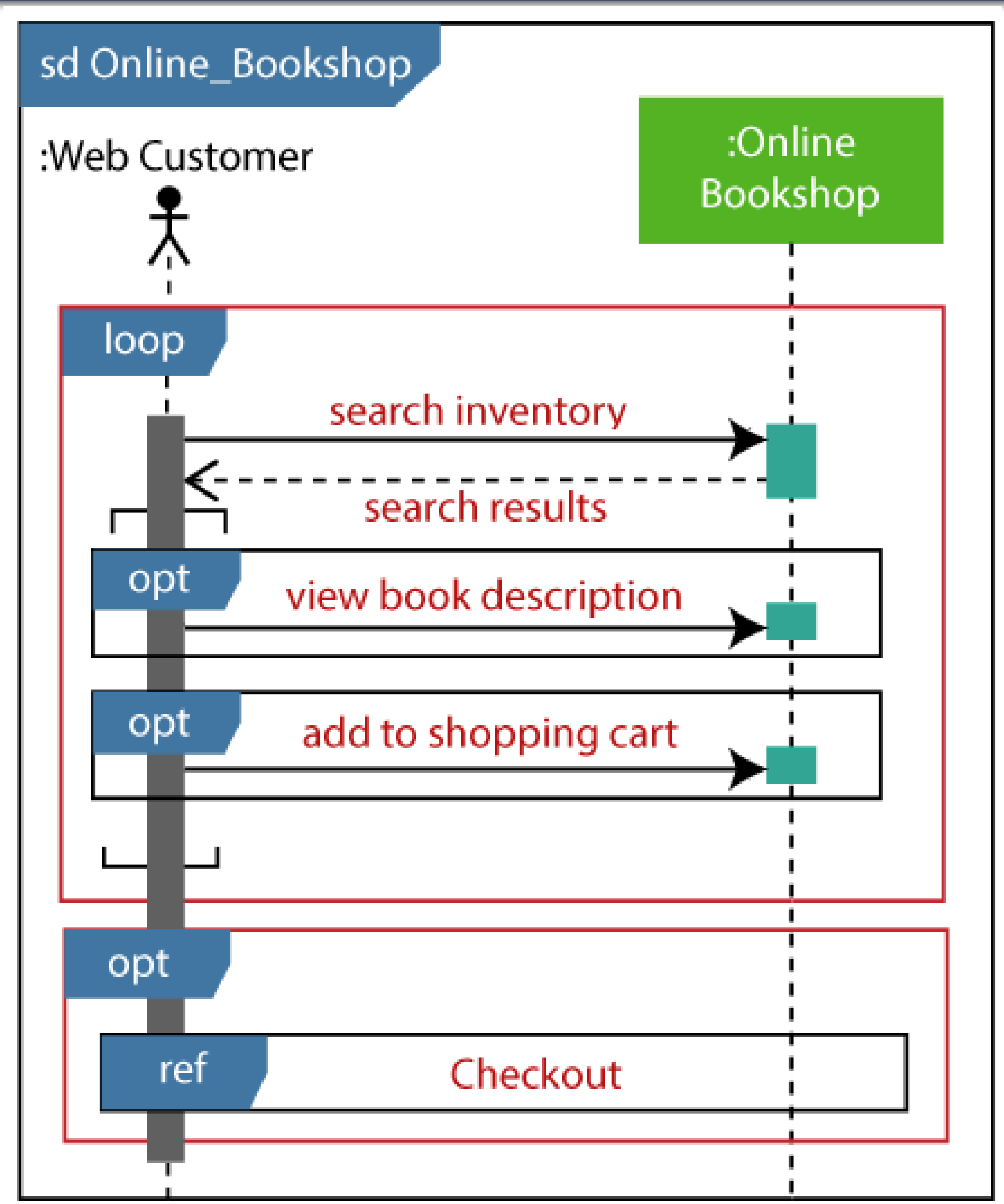
❑Types of Fragments:



Operator	Fragment Type
alt	Alternative multiple fragments: The only fragment for which the condition is true, will execute.
opt	Optional: If the supplied condition is true, only then the fragments will execute. It is similar to alt with only one trace.
par	Parallel: Parallel executes fragments.
loop	Loop: Fragments are run multiple times, and the basis of interaction is shown by the guard.
region	Critical region: Only one thread can execute a fragment at once.
neg	Negative: A worthless communication is shown by the fragment.
ref	Reference: An interaction portrayed in another diagram. In this, a frame is drawn so as to cover the lifelines involved in the communication. The parameter and return value can be explained.
sd	Sequence Diagram: It is used to surround the whole sequence diagram.

CASE TOOLs – Sequence Diagram

- ❑Example of a high-level sequence diagram for online bookshop is given below.
- ❑Any online customer can search for a book catalog, view a description of a particular book, add a book to its shopping cart, and do checkout.



CASE TOOLs – Collaboration Diagram

❑The collaboration diagram is used to show the relationship between the objects in a system.

❑Both the sequence and the collaboration diagrams represent the same information but differently.

❑Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming.

❑An object consists of several features.

❑Multiple objects present in the system are connected to each other.

❑The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

CASE TOOLs – Collaboration Diagram

❑**Notations of a Collaboration Diagram:** Following are the components of a component diagram that are enlisted below:

1. **Objects**
2. **Actors**
3. **Links**
4. **Messages**

CASE TOOLs – Collaboration Diagram

❑Notations of a Collaboration Diagram:

❑**Object:** The representation of an object is done by an object symbol with its name and class underlined, separated by a colon.

❑In the collaboration diagram, objects are utilized in the following ways: The object is represented by specifying their name and class.

1. It is not mandatory for every class to appear.
2. A class may constitute more than one object.
3. In the collaboration diagram, firstly, the object is created, and then its class is specified.
4. To differentiate one object from another object, it is necessary to name them.

❑ **Actors:** In the collaboration diagram, the actor plays the main role as it invokes the interaction. Each actor has its respective role and name. In this, one actor initiates the use case.

CASE TOOLs – Collaboration Diagram

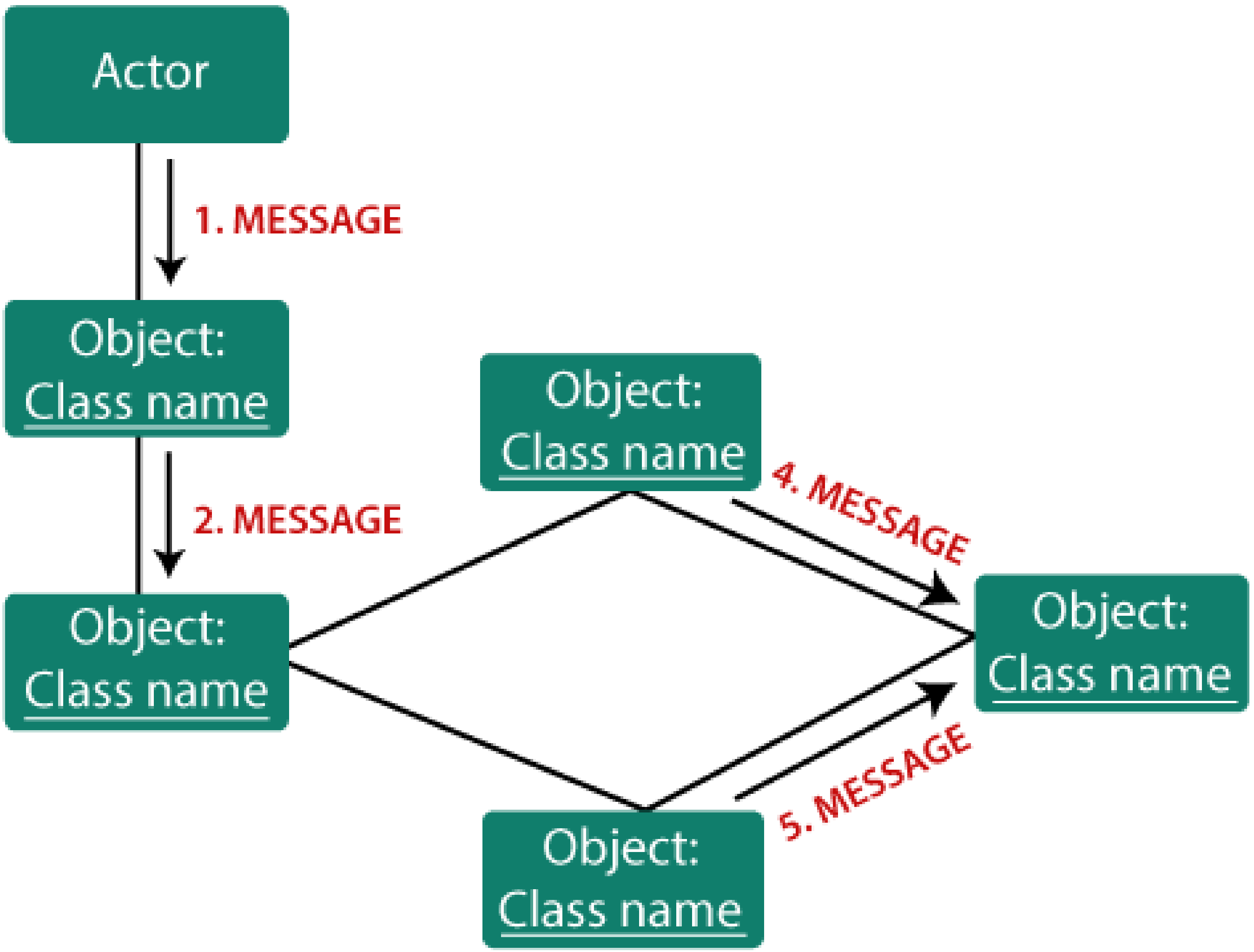
❑**Notations of a Collaboration Diagram:**

❑**Links:** The link is an instance of association, which associates the objects and actors. It portrays a relationship between the objects through which the messages are sent. It is represented by a solid line. The link helps an object to connect with or navigate to another object, such that the message flows are attached to links.

❑**Messages:** It is a communication between objects which carries information and includes a sequence number, so that the activity may take place. It is represented by a labeled arrow, which is placed near a link. The messages are sent from the sender to the receiver, and the direction must be navigable in that particular direction. The receiver must understand the message.

CASE TOOLs – Collaboration Diagram

❑Components of a Collaboration Diagram:



CASE TOOLs – Collaboration Diagram

❑When to use a Collaboration Diagram?

❑The collaborations are used when it is essential to depict the relationship between the object.

❑Both the sequence and collaboration diagrams represent the same information, but the way of portraying it quite different.

❑The collaboration diagrams are best suited for analyzing use cases.

1. To model collaboration among the objects or roles that carry the functionalities of use cases and operations.
2. To model the mechanism inside the architectural design of the system.
3. To capture the interactions that represent the flow of messages between the objects and the roles inside the collaboration.
4. To model different scenarios within the use case or operation, involving a collaboration of several objects and interactions.
5. To support the identification of objects participating in the use case.
6. In the collaboration diagram, each message constitutes a sequence number, such that the top-level message is marked as one and so on. The messages sent during the same call are denoted with the same decimal prefix, but with different suffixes of 1, 2, etc. as per their occurrence.

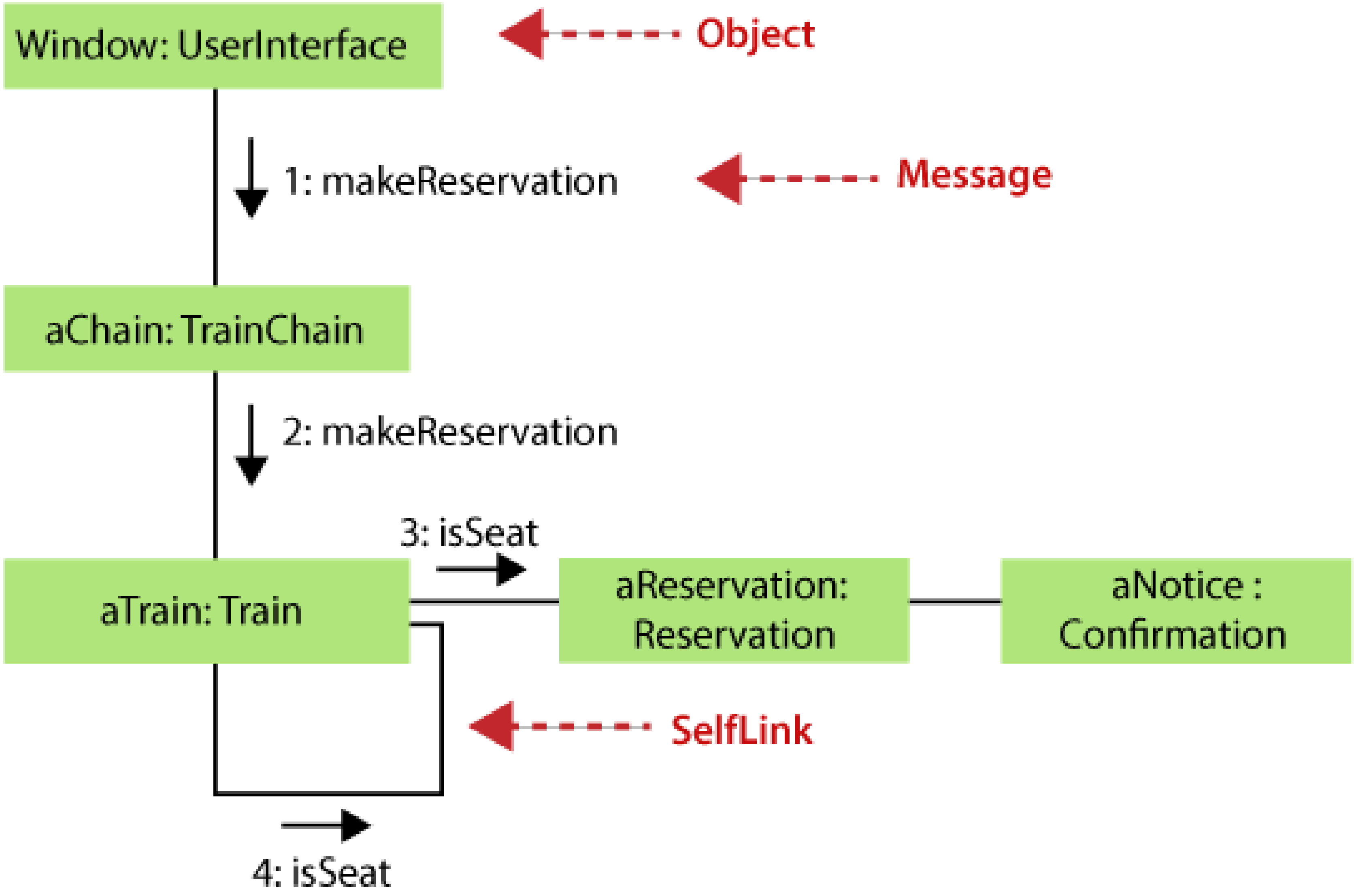
CASE TOOLs – Collaboration Diagram

❑Steps for creating a Collaboration Diagram

1. Determine the behavior for which the realization and implementation are specified.
2. Discover the structural elements that are class roles, objects, and subsystems for performing the functionality of collaboration.
   - Choose the context of an interaction: system, subsystem, use case, and operation.
3. Think through alternative situations that may be involved.
   - Implementation of a collaboration diagram at an instance level, if needed.
   - A specification level diagram may be made in the instance level sequence diagram for summarizing alternative situations.

CASE TOOLs – Collaboration Diagram

❑Example of Collaboration Diagram



CASE TOOLs – Activity Diagram

❑In the UML, activity diagram is used to demonstrate the flow of control within the system rather than the implementation.

❑It models the concurrent and sequential activities.

❑The activity diagram helps in envisioning the workflow from one activity to another.

❑It put emphasis on the condition of flow and the order in which it occurs.

❑The flow can be sequential, branched, or concurrent, and to deal with such kinds of flows, the activity diagram has come up with a fork, join, etc.

❑It is also termed as an object-oriented flowchart.

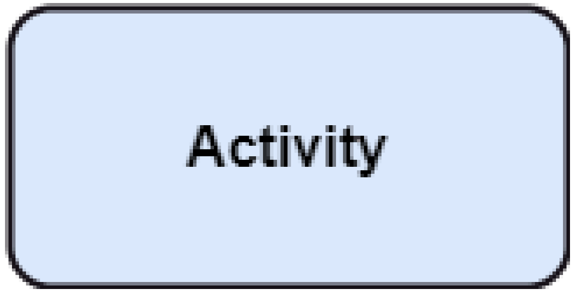
❑It encompasses activities composed of a set of actions or operations that are applied to model the behavioral diagram.

CASE TOOLs – Activity Diagram

❑Components of an Activity Diagram

1. Activities :

- The categorization of behavior into one or more actions is termed as an activity.
- In other words, it can be said that an activity is a network of nodes that are connected by edges.
- The edges depict the flow of execution. It may contain action nodes, control nodes, or object nodes.
- The control flow of activity is represented by control nodes and object nodes that illustrates the objects used within an activity.
- The activities are initiated at the initial node and are terminated at the final node.

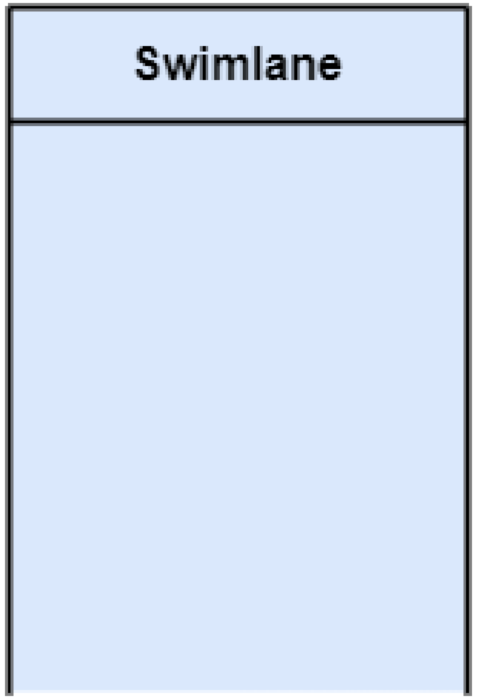


CASE TOOLs – Activity Diagram

❑Components of an Activity Diagram

2. Activity partition /swimlane

- The swimlane is used to cluster all the related activities in one column or one row.
- It can be either vertical or horizontal. It used to add modularity to the activity diagram.
- It is not necessary to incorporate swimlane in the activity diagram.
- But it is used to add more transparency to the activity diagram.

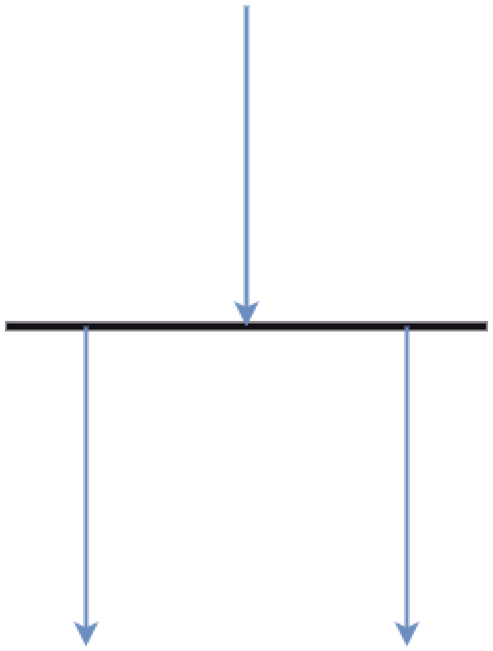


CASE TOOLs – Activity Diagram

❑Components of an Activity Diagram

3. Forks

- Forks and join nodes generate the concurrent flow inside the activity.
- A fork node consists of one inward edge and several outward edges.
- It is the same as that of various decision parameters.
- Whenever a data is received at an inward edge, it gets copied and split crossways various outward edges.
- It split a single inward flow into multiple parallel flows.

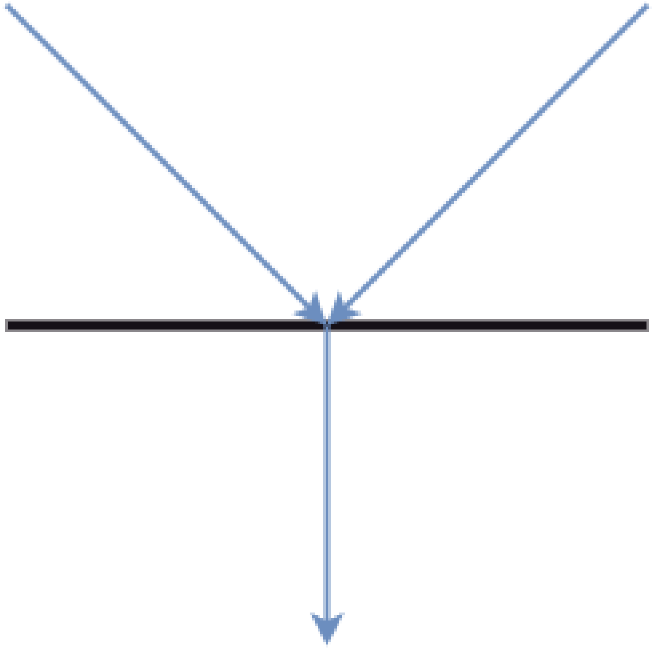


CASE TOOLs – Activity Diagram

❑Components of an Activity Diagram

4. Join

- Join nodes are the opposite of fork nodes.
- A Logical AND operation is performed on all of the inward edges as it synchronizes the flow of input across one single output (outward) edge.



CASE TOOLs – Activity Diagram

❑Components of an Activity Diagram

5. Pin

- It is a small rectangle, which is attached to the action rectangle.
- It clears out all the messy and complicated thing to manage the execution flow of activities.
- It is an object node that precisely represents one input to or output from the action.

CASE TOOLs – Activity Diagram

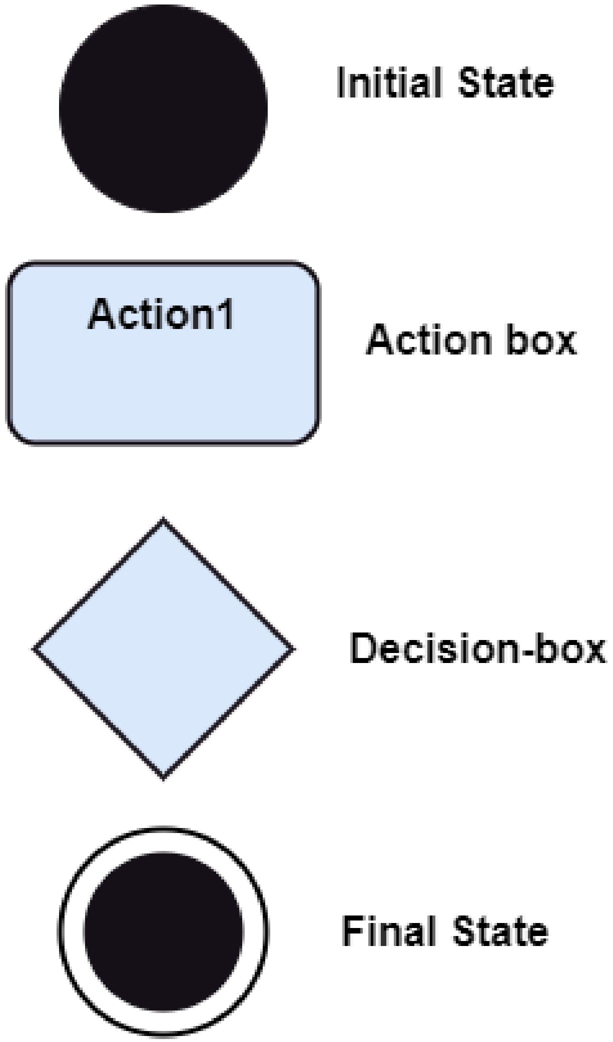
❑Activity diagram constitutes **following notations**:

❑**Initial State:** It depicts the initial stage or beginning of the set of actions.

❑**Final State:** It is the stage where all the control flows and object flows end.

❑**Decision Box:** It makes sure that the control flow or object flow will follow only one path.

❑**Action Box:** It represents the set of actions that are to be performed.



CASE TOOLs – Activity Diagram

❑Why use Activity Diagram?

❑An event is created as an activity diagram encompassing a group of nodes associated with edges.

❑To model the behavior of activities, they can be attached to any modeling element.

❑It can model use cases, classes, interfaces, components, and collaborations.

❑It mainly models processes and workflows.

❑It envisions the dynamic behavior of the system as well as constructs a runnable system that incorporates forward and reverse engineering.

❑It does not include the message part, which means message flow is not represented in an activity diagram.

❑It is the same as that of a flowchart but not exactly a flowchart itself. It is used to depict the flow between several activities.

CASE TOOLs – Activity Diagram

❑How to draw an Activity Diagram?

❑An activity diagram is a flowchart of activities, as it represents the workflow among various activities.

❑They are identical to the flowcharts, but they themself are not exactly the flowchart.

❑In other words, it can be said that an activity diagram is an enhancement of the flowchart, which encompasses several unique skills.

❑Since it incorporates swimlanes, branching, parallel flows, join nodes, control nodes, and forks, it supports exception handling.

❑A system must be explored as a whole before drawing an activity diagram to provide a clearer view of the user.

❑All of the activities are explored after they are properly analyzed for finding out the constraints applied to the activities.

❑Each and every activity, condition, and association must be recognized.

CASE TOOLs – Activity Diagram

❑**How to draw an Activity Diagram?**

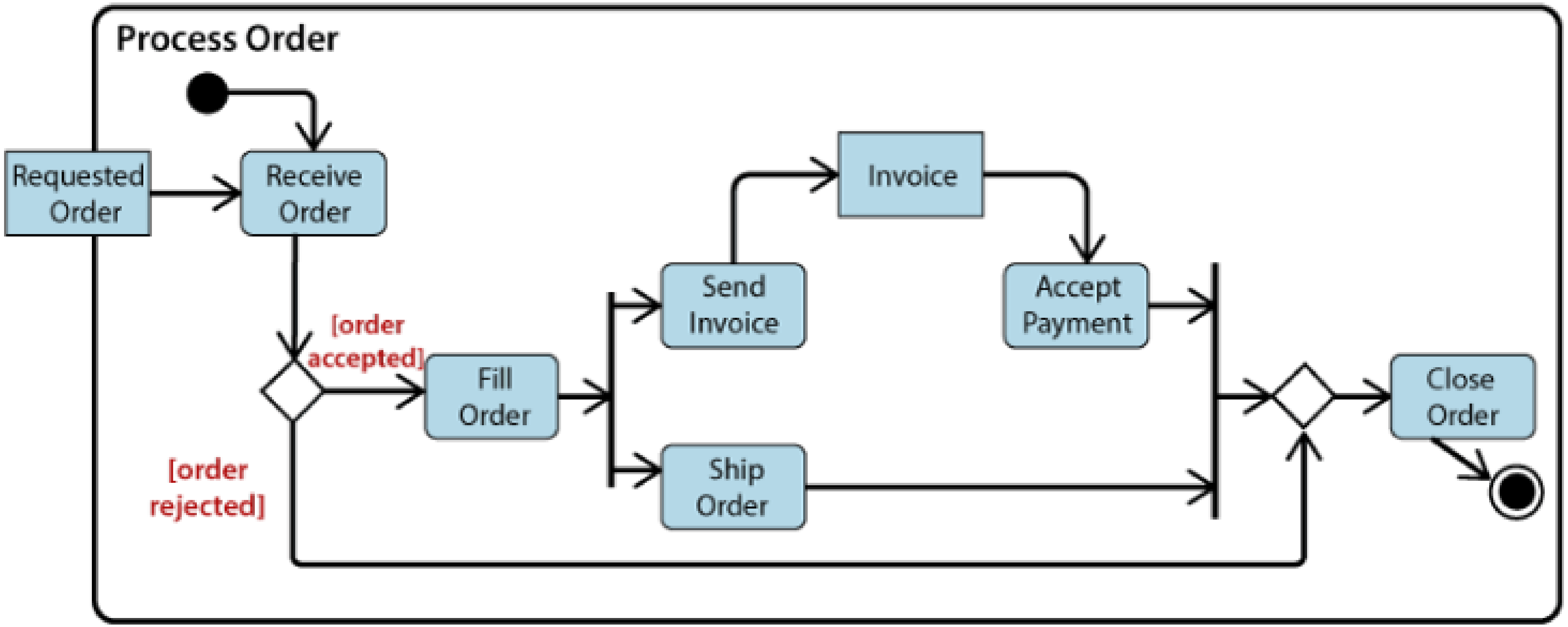
❑After gathering all the essential information, an abstract or a prototype is built, which is then transformed into the actual diagram.

❑Following are the rules that are to be followed for drawing an activity diagram:

1. A meaningful name should be given to each and every activity.
2. Identify all of the constraints.
3. Acknowledge the activity associations.

CASE TOOLs – Activity Diagram

❑**Example:** Here the input parameter is the Requested order, and once the order is accepted, all of the required information is then filled, payment is also accepted, and then the order is shipped. It permits order shipment before an invoice is sent or payment is completed.



CASE TOOLs – Activity Diagram

❑When to use an Activity Diagram?

1. To graphically model the workflow in an easier and understandable way.
2. To model the execution flow among several activities.
3. To model comprehensive information of a function or an algorithm employed within the system.
4. To model the business process and its workflow.
5. To envision the dynamic aspect of a system.
6. To generate the top-level flowcharts for representing the workflow of an application.
7. To represent a high-level view of a distributed or an object-oriented system.

Note for Students

- ❑This power point presentation is for lecture, therefore it is suggested that also utilize the text books and lecture notes.