Activity Diagram

- An activity diagram is used to display the sequence of activities for a system functionality.
- Activity diagrams show the sequence of activities in a process, including sequential and parallel activities, and decisions that are made to complete the process.
- Activity diagrams are used to describe business processes and use cases to document the implementation of system processes.
- An activity diagram is usually created for one use case and may show the different possible scenarios.
- An activity diagram is essentially a flowchart that shows activities performed by a system. This flowchart represents the flow from one activity to another activity. It models the concurrent and sequential activities.
- It is also called object-oriented flowchart.
- The basic purpose of activity diagrams is to capture the dynamic behavior of the system.

Activities

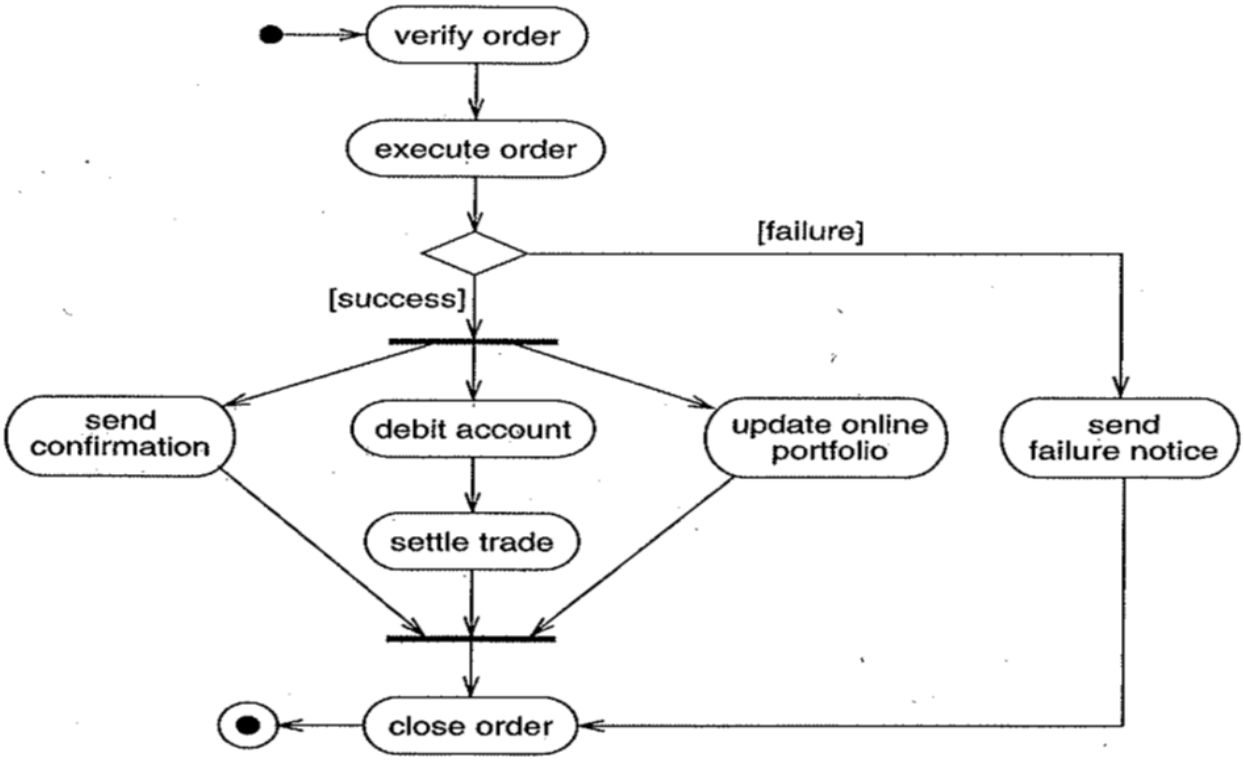
- The steps of an activity diagram are the activities which are basically operations.
- These activities are specifically actions with respect to events from a state model.
- One activity can be started after completion of a previous activity or multiple activities can be performed simultaneously(concurrent activities).

Consider the following **activity diagram** for a **stock trade processing**.

Stock trade can be processed when received by an online stock broker.

Here, ***elongated ovals*** show *activities* and a ***arrows*** show *sequence of activities*. The ***diamond*** shows a *decision point* and***heavy bar*** shows *splitting or merging of concurrent flows/threads.*

The online stock broker first verifies the order against the customer's account, then execute it with the stock exchange. if the order executes successfully, the system does three things concurrently i.e. mails trade conformation to the customer, updates the online portfolio to reflect the results of the trade, and settles the trade by debiting the account. When all three concurrent threads (or activities) have been completed, the system merges control into a single thread (or activity) and closes the order. If the order execution fails, then the system sends a failure notice the customer and closes the order.



Activity diagram for stock trade processing.

Benefits or Usage of activity diagrams

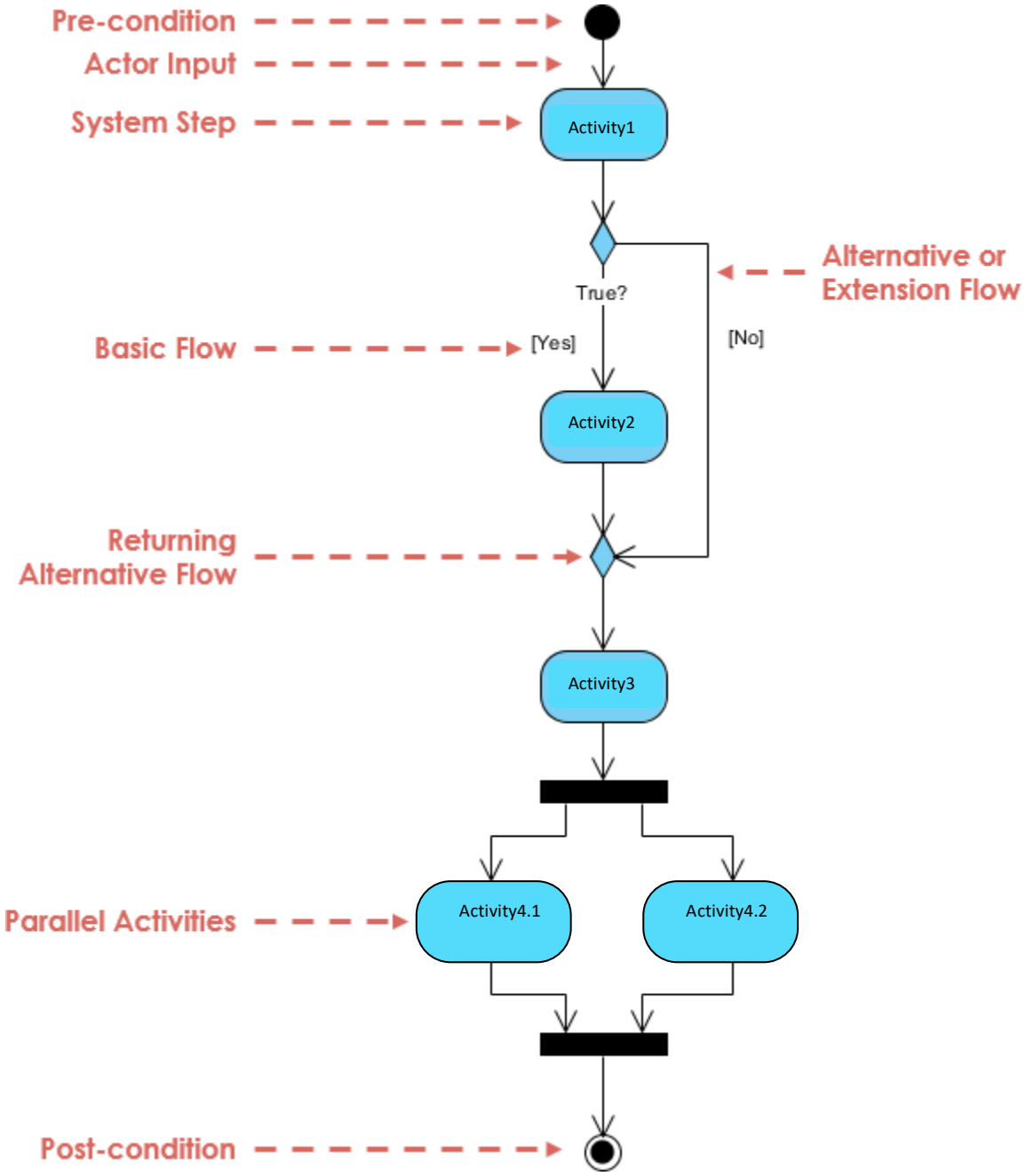
Activity diagrams present a number of benefits to users which are as follows.

- Describe the steps performed in a UML use case.
- Illustrate a business process or workflow between users and the system.
- Simplify and improve any process by clarifying complicated use cases.
- Demonstrate the logic of an algorithm.
- Model software architecture elements, such as method, function, and operation.

Steps to create Activity Diagram

- Identify candidate use cases, through the examination of business workflows
- Identify pre- and post-conditions (the context) for use cases
- Model workflows between/within use cases
- Model complex workflows in operations on objects
- Model in detail complex activities in a high level activity Diagram

The following is the basic activity diagram with usual UML notations



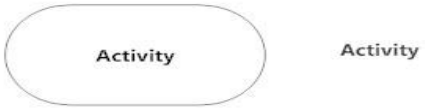
Activity Diagram Notations and Symbols:

Initial State or Start Point

A small filled circle followed by an arrow represents the initial action state or the start point for any activity diagram.

Activity or Action State

An activity or action state represents the non-interruptible action of objects. Activity or action state is represented by using a rectangle with rounded corners or elongated ovals.



Action Flow or Control flow

Action flows, also called edges and paths, illustrate the transitions from one activity to another. They are usually drawn with an arrowed line.

Decisions and Branching

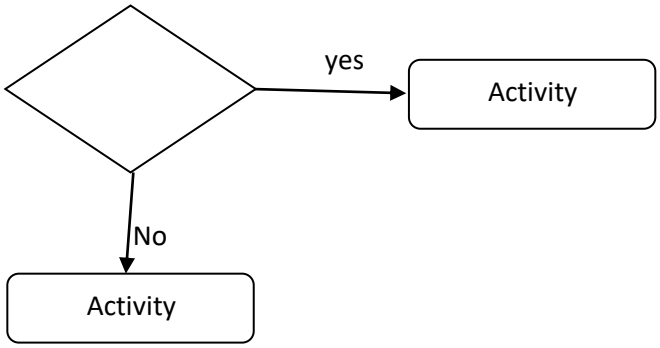
Used to represent a conditional branch point with one input and multiple outputs.

A diamond represents a decision with alternate paths. When an activity requires a decision prior to moving on to the next activity, add a diamond between the two activities.

The outgoing alternates should be labeled with a condition or guard expression.

Guards

In UML, guards are statements written next to a decision diamond that must be true before moving next to the next activity.

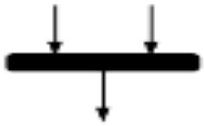


Synchronization

A fork node is used to split a single incoming flow into multiple concurrent flowsi.e. a fork represents splitting of a single activity into multiple parallel or concurrent activities. It is represented as a straight, slightly thicker line in an activity diagram.

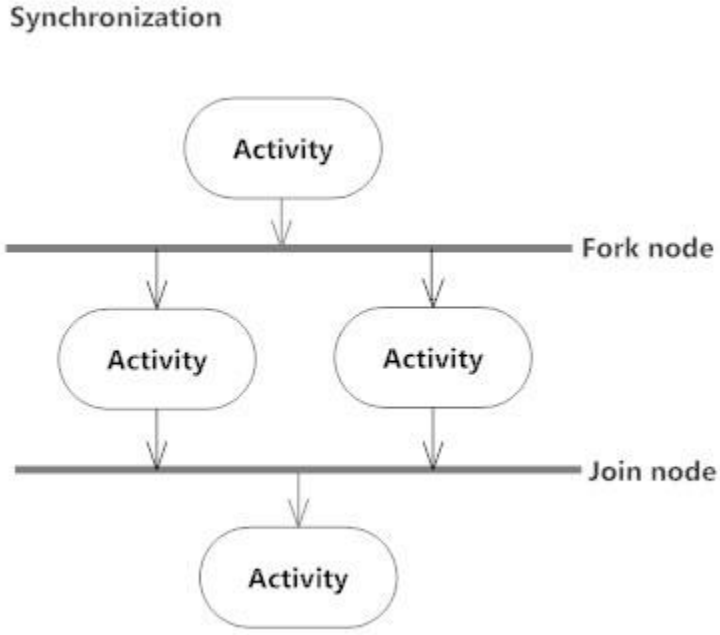
Fork Symbol

A join node joins multiple concurrent flows back into a single outgoing flow.**A join** combines multiple concurrent activities and re-introduces them to a flow where only one activity occurs at a time.



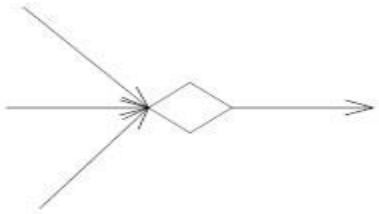
Join Symbol

A fork and join mode used together are often referred to as synchronization.



Merge Event

A merge event brings together multiple flows that are not concurrent.



Merge

Sent and Received Signals:

Signals:

Activities may involve interactions with external people, systems, or processes**. In activity diagrams, signals represent interactions with external systems or processes or people.**

Signals represent how activities can be modified from outside the system.

They usually appear in pairs of sent and received signals, because the state can't change until a response is received, much like synchronous messages in a sequence diagram.

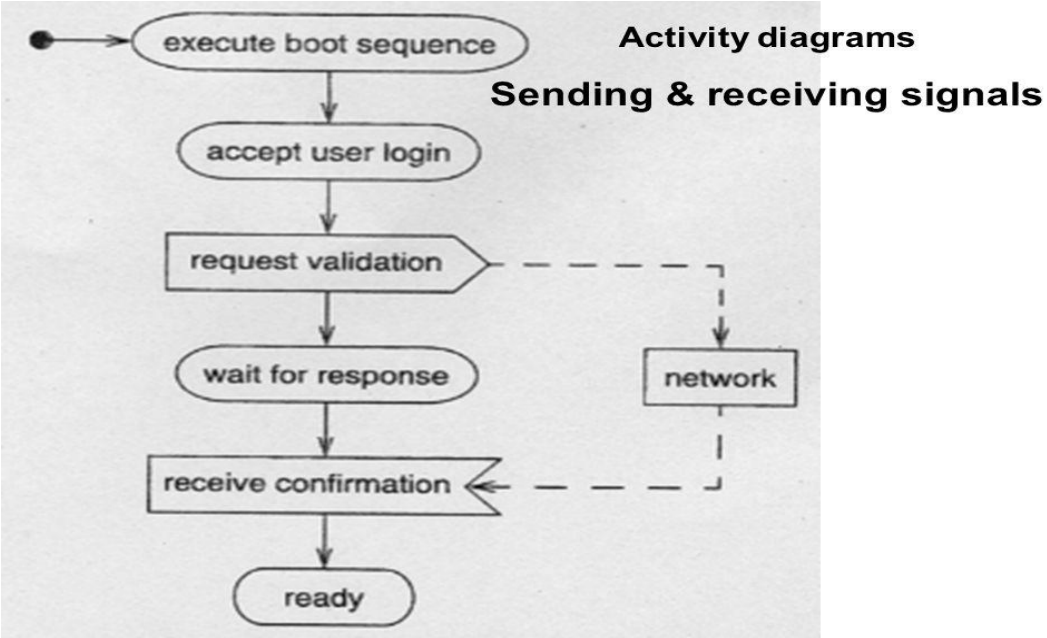
For example, an authorization of payment is needed before an order can be completed. Here, the payment activity is carried out with respect to an external entity like bank.

Send signal symbol(convex pentagon) indicates that a signal is being sent to a receiving activity

Receive signal symbol (concave pentagon) which demonstrates the acceptance of an event.

The following examples illustrates an activity diagram with sending and receiving signal.

Consider a workstation that is turned on.It goes through a boot sequence and then prompted for user log in.After the entry of user_name and password,the workstation queried the network to validate the user.Upon validation ,the workstation then finishes the start up process.



Object Flows:

Activity diagram sometimes shows objects that are input to an activity or output from the activities.

Objects input to an activity means objects as arguments values to the operation which initiates the activity and objects output to an activity means the returning object values from the operation after completion of the activity.

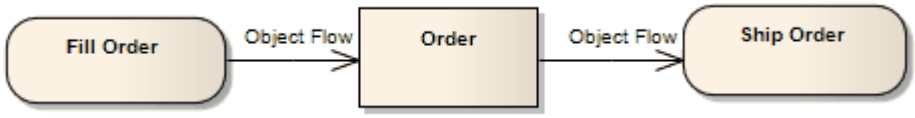
In UML the object values are represented in square bracket below the object name.

:Airplane [at gate]	:Airplane [taxiing]	:Airplane [in flight]

at gate, *taxiing* and *in flight* are object values.

In Activity diagrams, there are several ways to define the flow of data between objects.

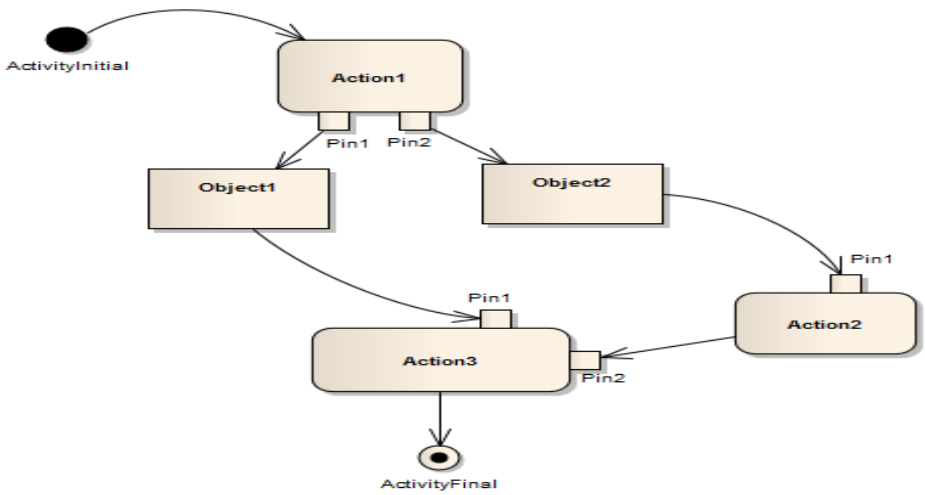
The following diagram depicts a simple Object Flow between two actions, Fill Order and Ship Order, both accessing order information. The order object is the result of the operation to perform the Fill Order activity and the order object is taken as input parameter for the operations to perform Ship order activity.



Pins are also used to show object flows. These are the small squares attached to the activities where objects are taken as parameter to the operations. In the following diagram Action Pins are used to reflect the order.

Consider the following example which represent object flows with **pin** :

1. Execute Action1. This action produces two output parameters: Object1 and Object2.
2. Execute Action2. This action requires Object2 as an input parameter. It DOES NOT REQUIRE Object1 as an input parameter.
3. Execute Action3. This action requires both Object1 and Object2 (output parameters of Action1) as input parameters.



Activity diagram examples with object flows:

In the following diagram , an airplane goes through several states as it leaves the gate, flies and then lands again.

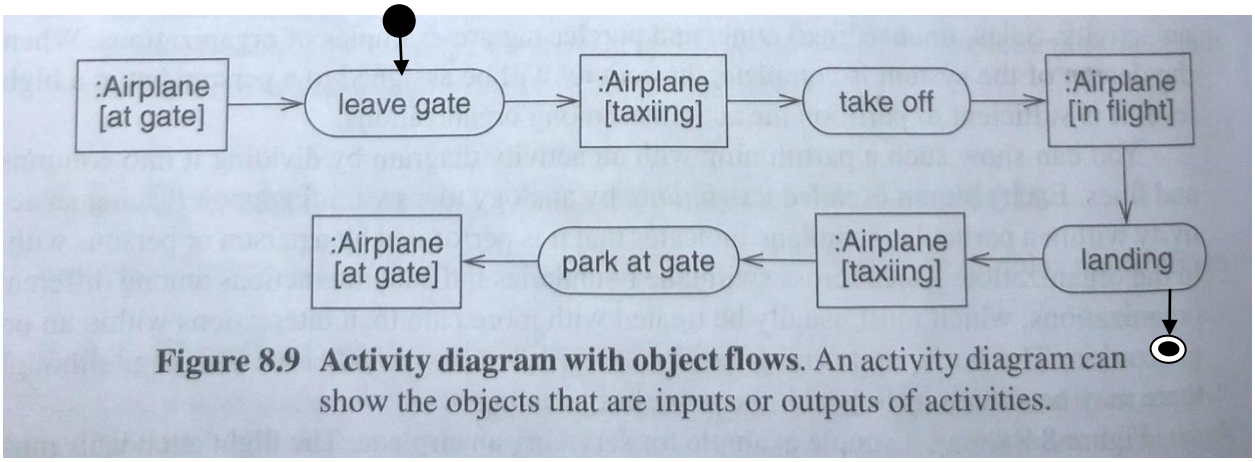
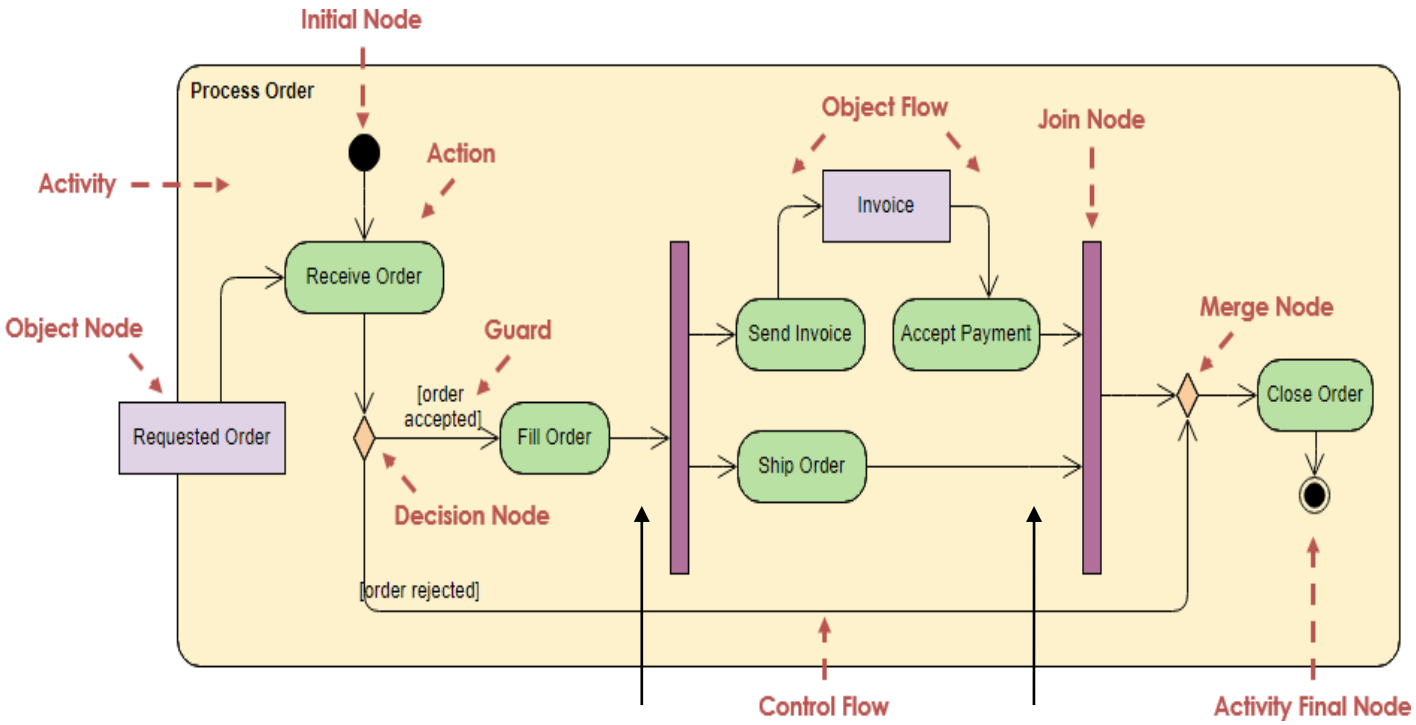


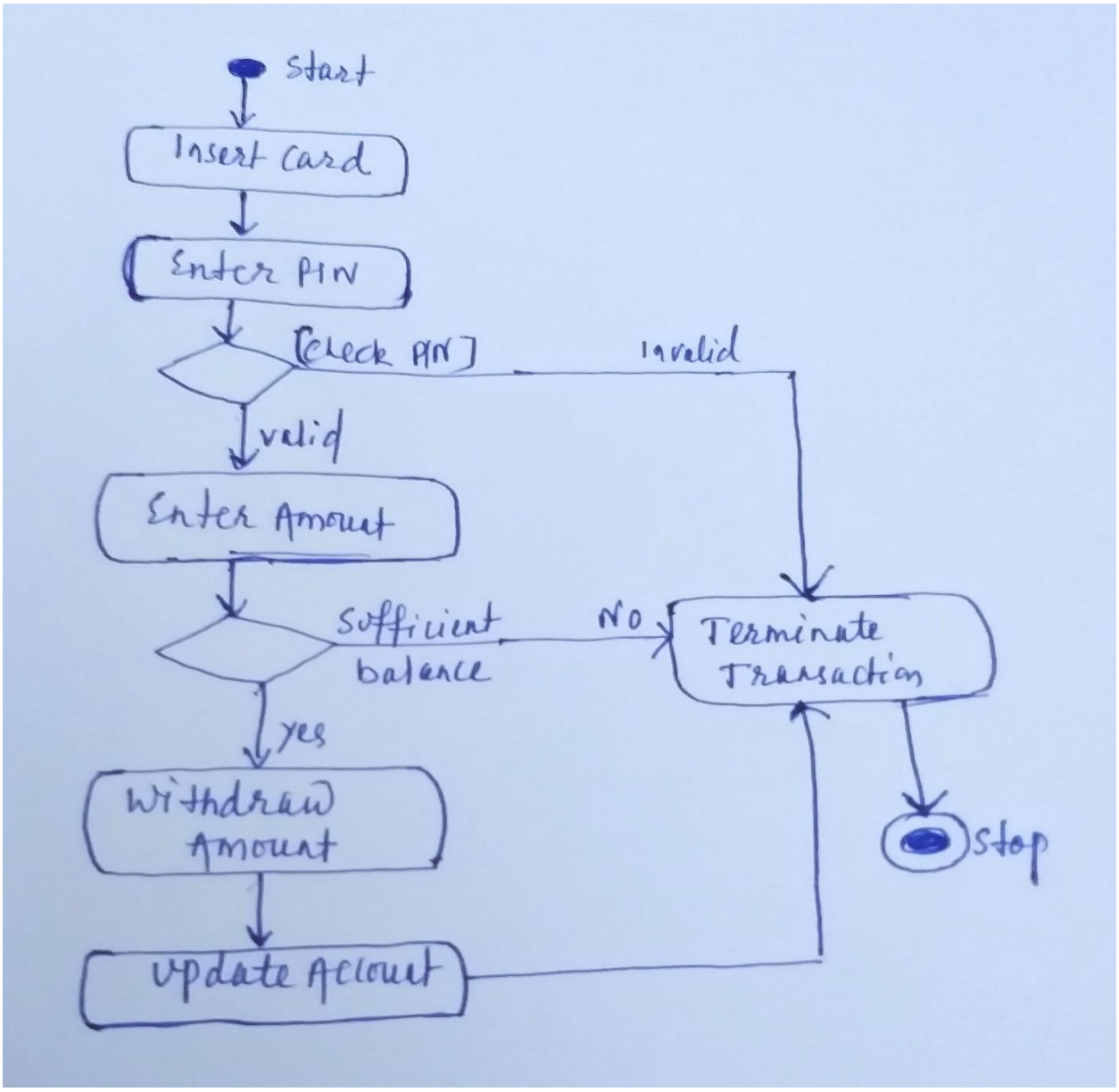
Figure 8.9 Activity diagram with object flows. An activity diagram can show the objects that are inputs or outputs of activities.

The following is the activity diagram for Process Order Activity with object flows



Activity diagrams tell you what happens, but they do not tell you who does what.

Consider the following example of an activity diagram for AMT withdrawal transaction



The above activity diagram does not show which people or organization performs a particular activity

In programming, this means that the diagram does not convey which class is responsible for which activity.

In domain modelling, this means that the diagram does not convey which people or organizations are responsible for each activity.

Swimlanes are used to show which activities are performed by which organisation in the activity diagram.

Swimlanes:

Swimlanes describe which people or oraganization is responsible for the activities being performed in the activity diagram and how they are responsible.

The activity diagram only represents the activities being performed, but Swimlane describes who does what in a process or activity performed in an activity diagram.

To use swimlanes, the activity diagrams must be portioned into vertical zones separated by lines. Each zone represents the responsibilities of a particular people or organization or class .

Swimlanes group related activities into one column.

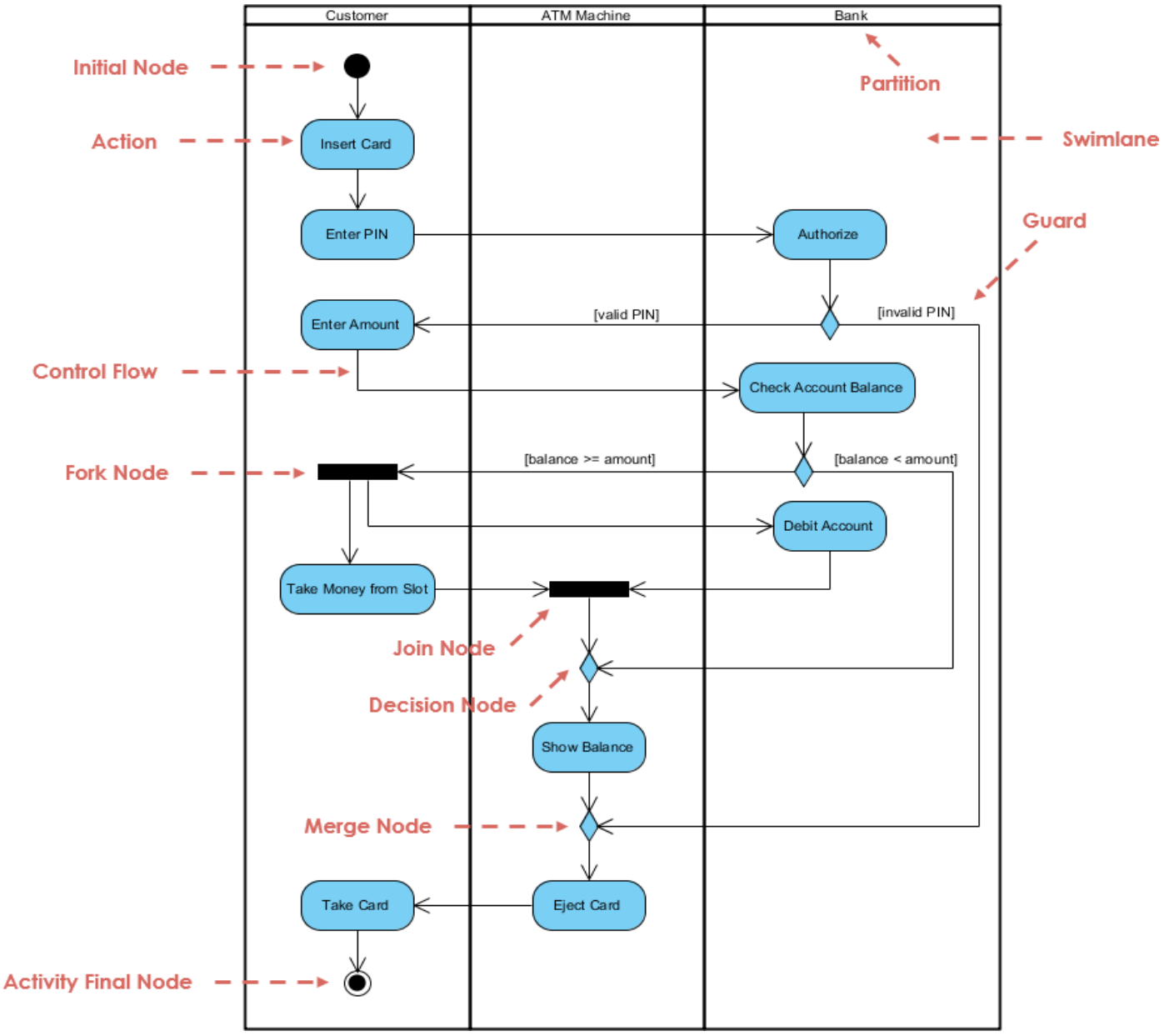
Swimlanes are used to add modularity to the activity diagram.

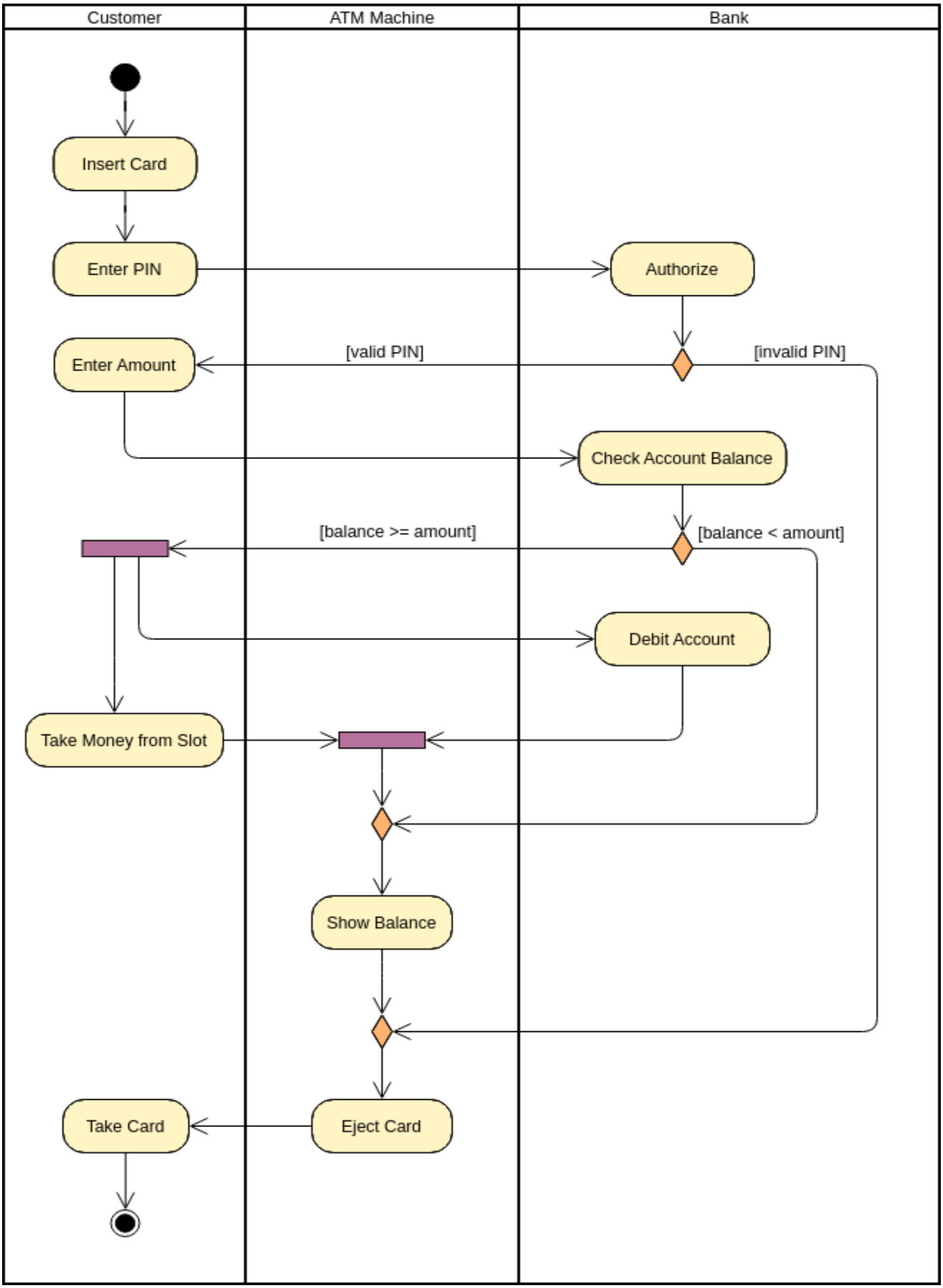
It is not mandatory to use swimlanes in activity diagram. They usually give more clarity to the activity diagram

Example of activity diagram with swimlanes:

Activity diagram for Withdraw money from an ATM .

The three involved classes (people or organizations) of the activity are Customer, ATM, and Bank. represented **swimlanes** that determine which object is responsible for which activity.


Customer
ATM Machine
Bank
Initial Node
Partition
Action
Swimlane
Insert Card
Enter PIN
Authorize
Guard
Enter Amount
[valid PIN]
[invalid PIN]
Control Flow
Check Account Balance
Fork Node
[balance >= amount]
[balance < amount]
Debit Account
Take Money from Slot
Join Node
Decision Node
Show Balance
Merge Node
Take Card
Eject Card
Activity Final Node


Customer
ATM Machine
Bank
Insert Card
Enter PIN
Authorize
Enter Amount
[valid PIN]
[invalid PIN]
Check Account Balance
[balance >= amount]
[balance < amount]
Debit Account
Take Money from Slot
Show Balance
Take Card
Eject Card