ACTIVITY DIAGRAMS

EX-5

ACTIVITY DIAGRAM

- An activity diagram is a **behavioral diagram** i.e. it depicts the behavior of a system.
- An activity diagram portrays the control flow from a start point to a finish point showing the various decision paths that exist while the activity is being executed.
- We can depict both sequential processing and concurrent processing of activities using an activity diagram.
- They are used in business and process modelling where their primary use is to depict the dynamic aspects of a system.

DIFFERENCE BETWEEN AN ACTIVITY DIAGRAM AND A FLOWCHART

- An activity diagram is very **similar to a flowchart**
- Flowcharts were typically invented earlier than activity diagrams. Non programmers use Flow charts to model workflows.
- For example: A manufacturer uses a flow chart to explain and illustrate how a particular product is manufactured.
- So, programmers use activity diagrams (advanced version of a flowchart) to depict workflows.
- An activity diagram is **used by developers** to understand the flow of programs on a high level.

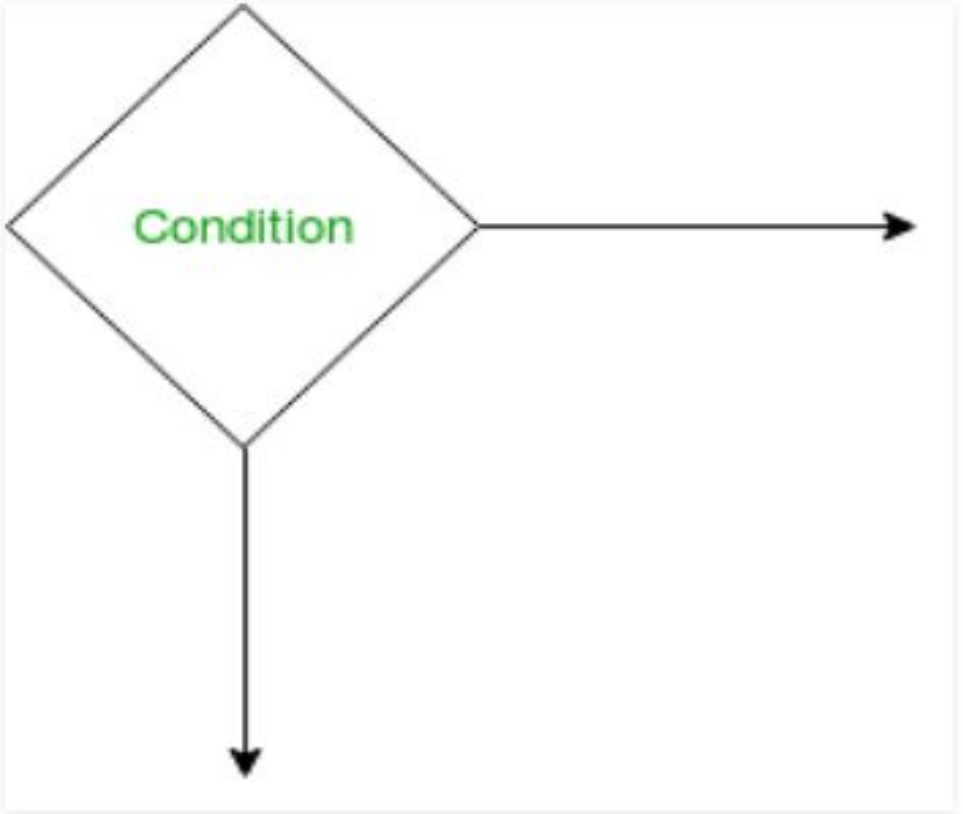
Activity Diagram Notations –

1. **Initial State –** The starting state before an activity takes place is depicted using the initial state.

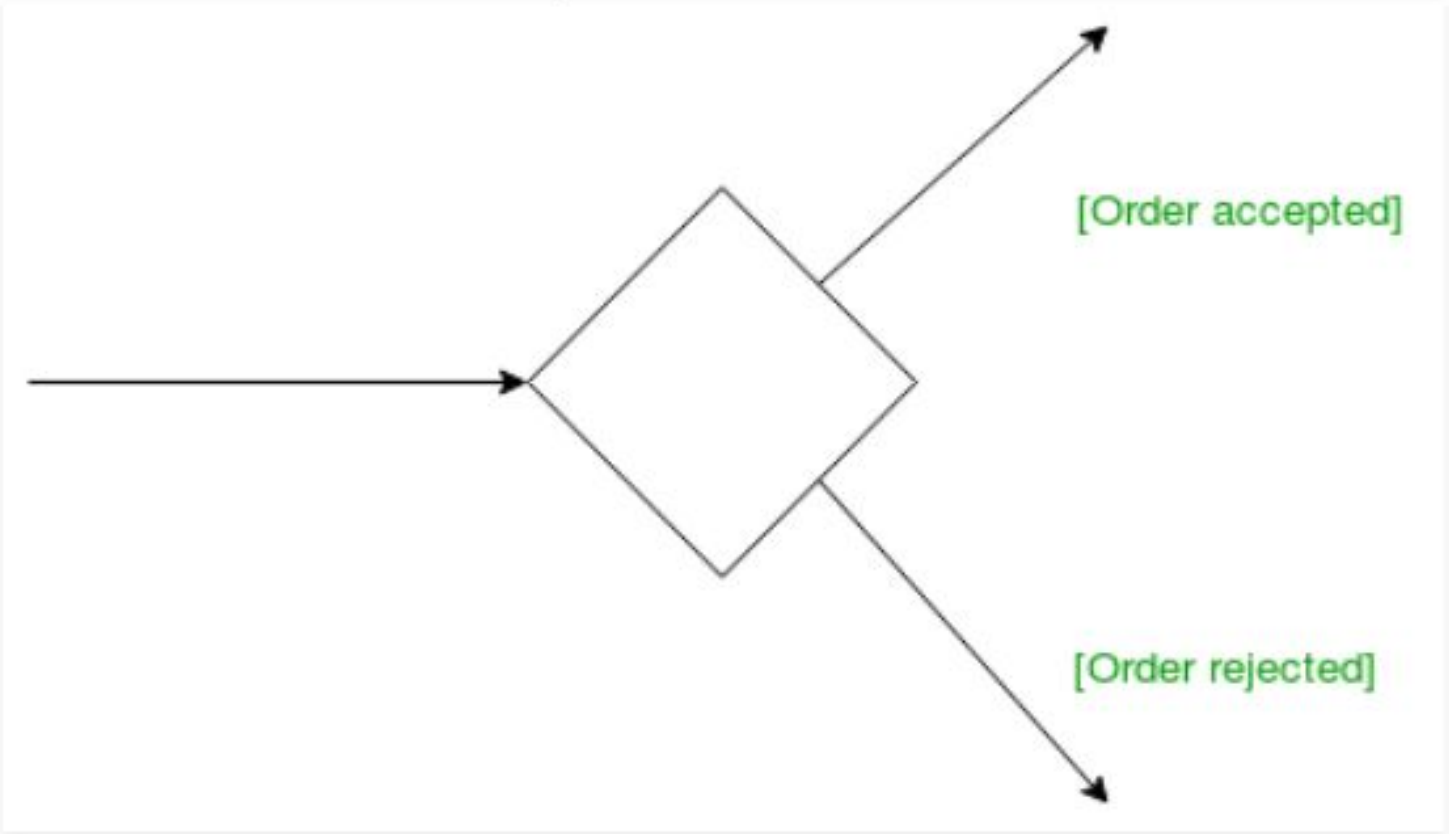
2. **Action or Activity State –** An activity represents execution of an action on objects or by objects. We represent an activity using a rectangle with rounded corners. Basically any action or event that takes place is represented using an activity.

3. **Action Flow or Control flows –** Action flows or Control flows are also referred to as paths and edges. They are used to show the transition from one activity state to another.

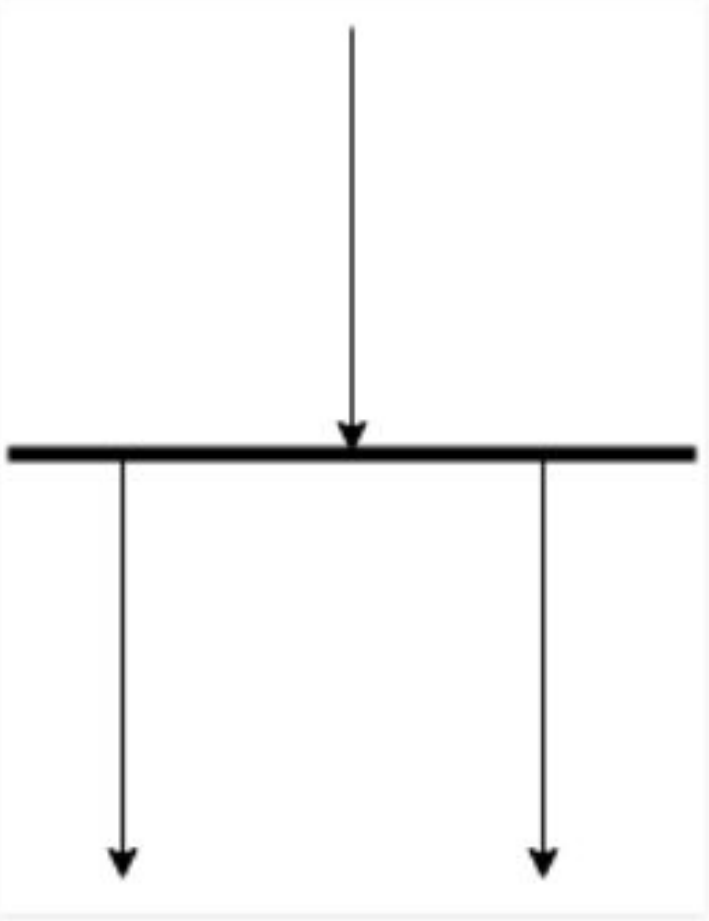
4. **Decision node and Branching –** When we need to make a decision before deciding the flow of control, we use the decision node.



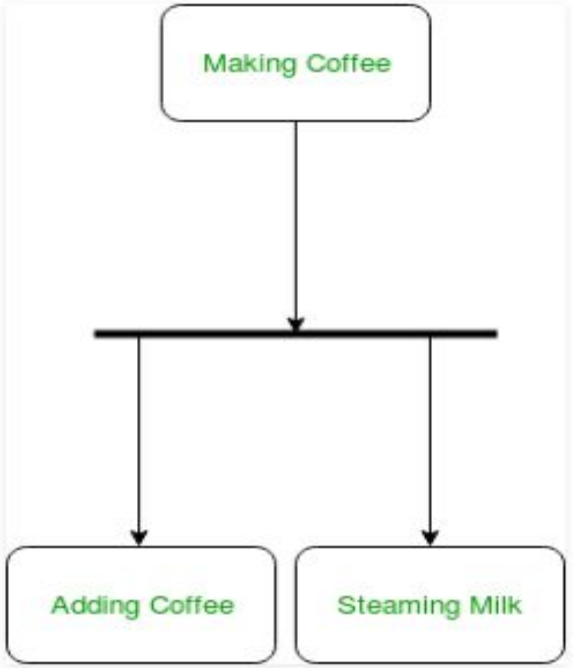
5. **Guards –** A Guard refers to a statement written next to a decision node on an arrow sometimes within square brackets.



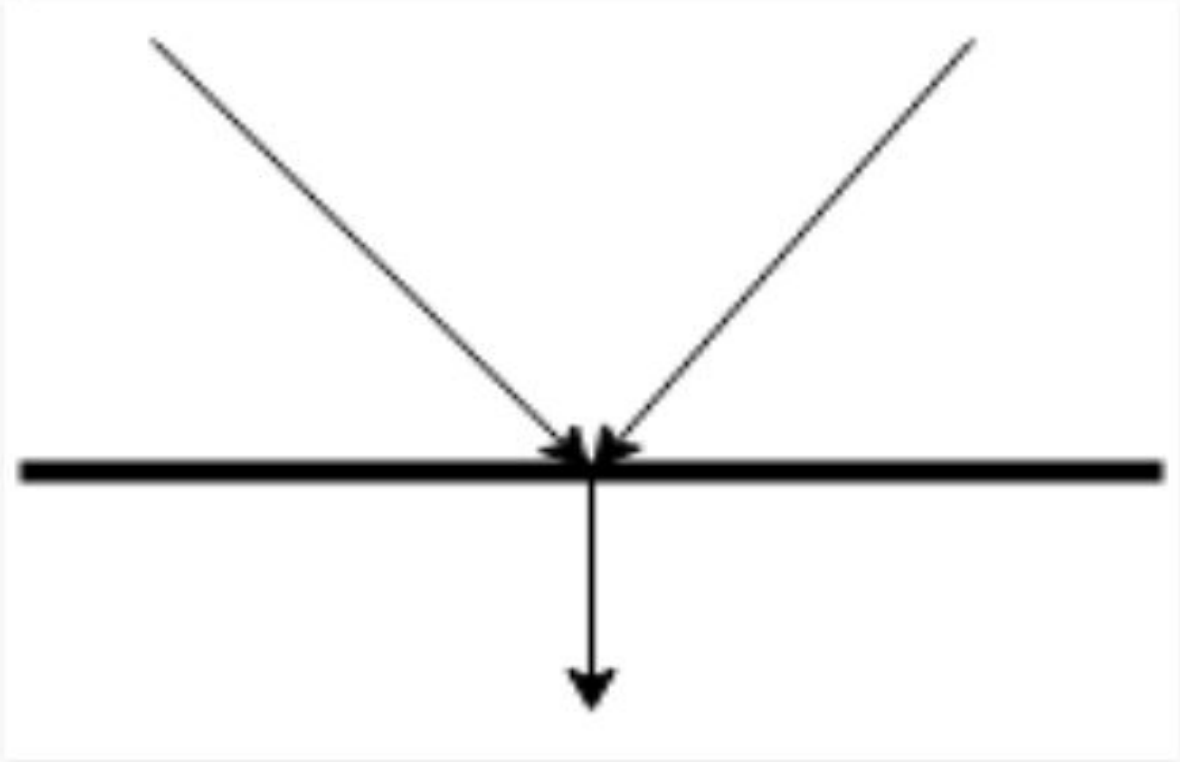
6. **Fork –** Fork nodes are used to support concurrent activities.



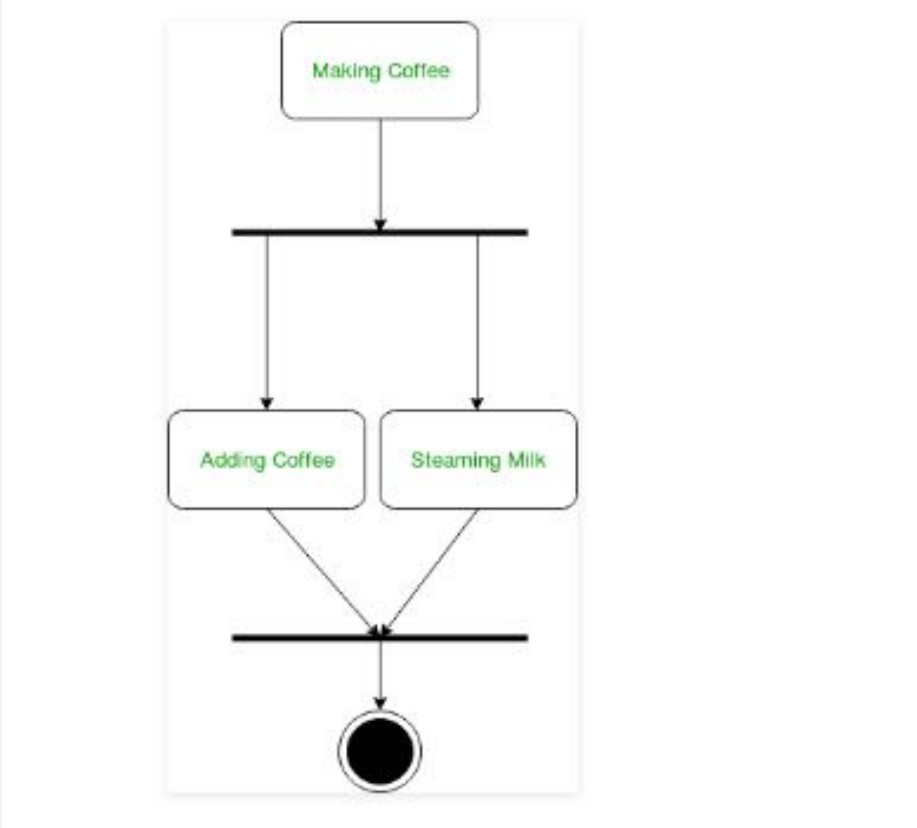
When we use a fork node when both the activities get executed concurrently i.e. no decision is made before splitting the activity into two parts. Both parts need to be executed in case of a fork statement.
We use a rounded solid rectangular bar to represent a Fork notation with incoming arrow from the parent activity state and outgoing arrows towards the newly created activities.
For example: In the example below, the activity of making coffee can be split into two concurrent activities and hence we use the fork notation.



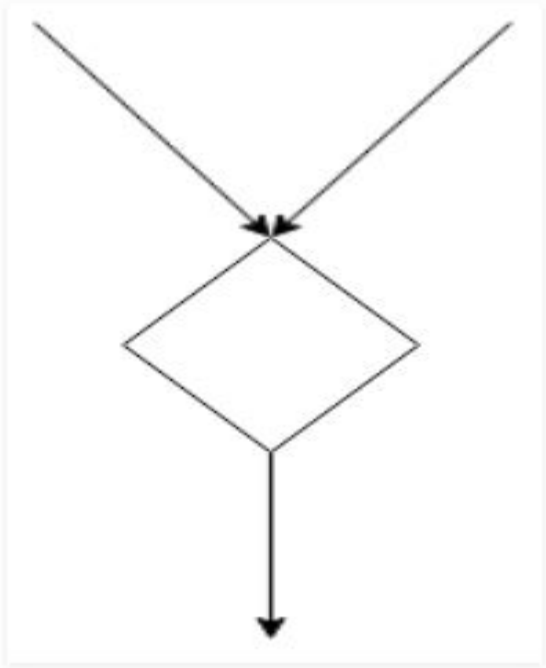
7. **Join –** Join nodes are used to support concurrent activities converging into one. For join notations we have two or more incoming edges and one outgoing edge.



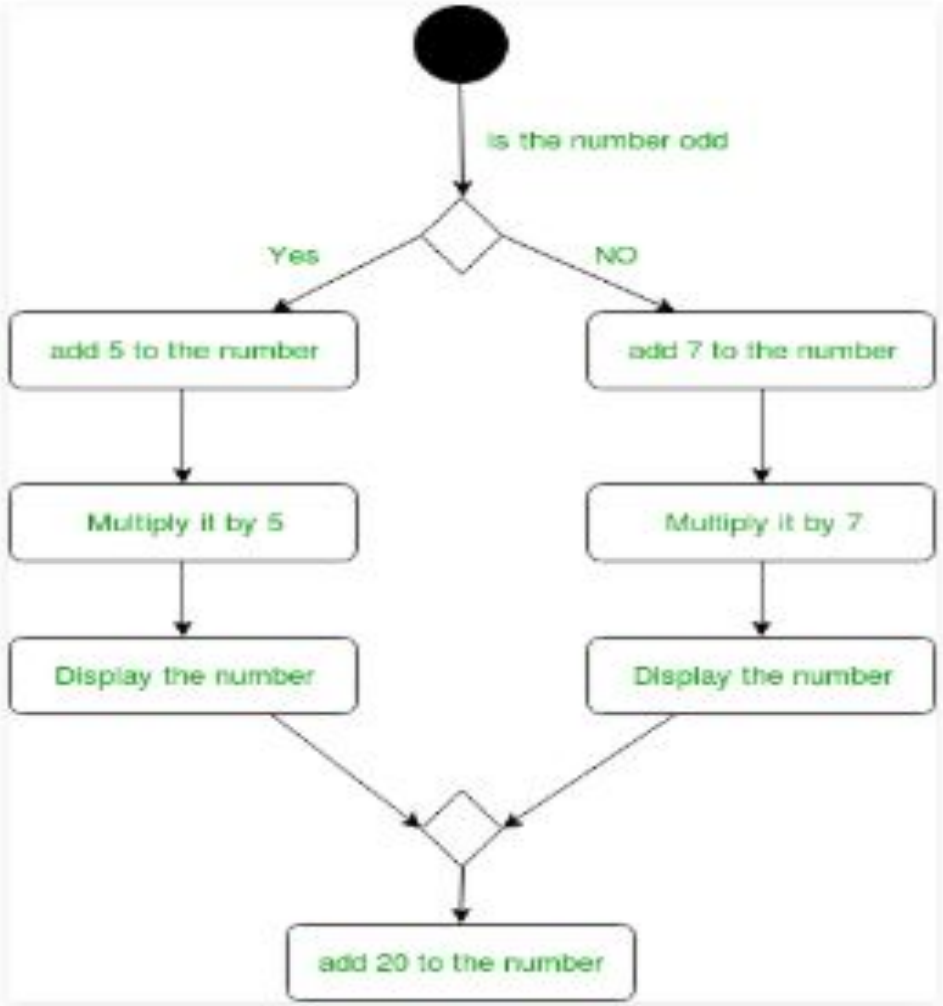
EXAMPLE



8. **Merge or Merge Event –** Scenarios arise when activities which are not being executed concurrently have to be merged. We use the merge notation for such scenarios. We can merge two or more activities into one if the control proceeds onto the next activity irrespective of the path chosen.



EXAMPLE


is the number odd
Yes
NO
add 5 to the number
add 7 to the number
Multiply it by 5
Multiply it by 7
Display the number
Display the number
add 20 to the number

9. **Swimlanes –** We use swimlanes for grouping related activities in one column. Swimlanes group related activities into one column or one row. Swimlanes can be vertical and horizontal. Swimlanes are used to add modularity to the activity diagram. It is not mandatory to use swimlanes. They usually give more clarity to the activity diagram. It's similar to creating a function in a program. It's not mandatory to do so, but, it is a recommended practice.

11. **Final State or End State –** The state which the system reaches when a particular process or activity ends is known as a Final State or End State. We use a filled circle within a circle notation to represent the final state in a state machine diagram. A system or a process can have multiple final states.

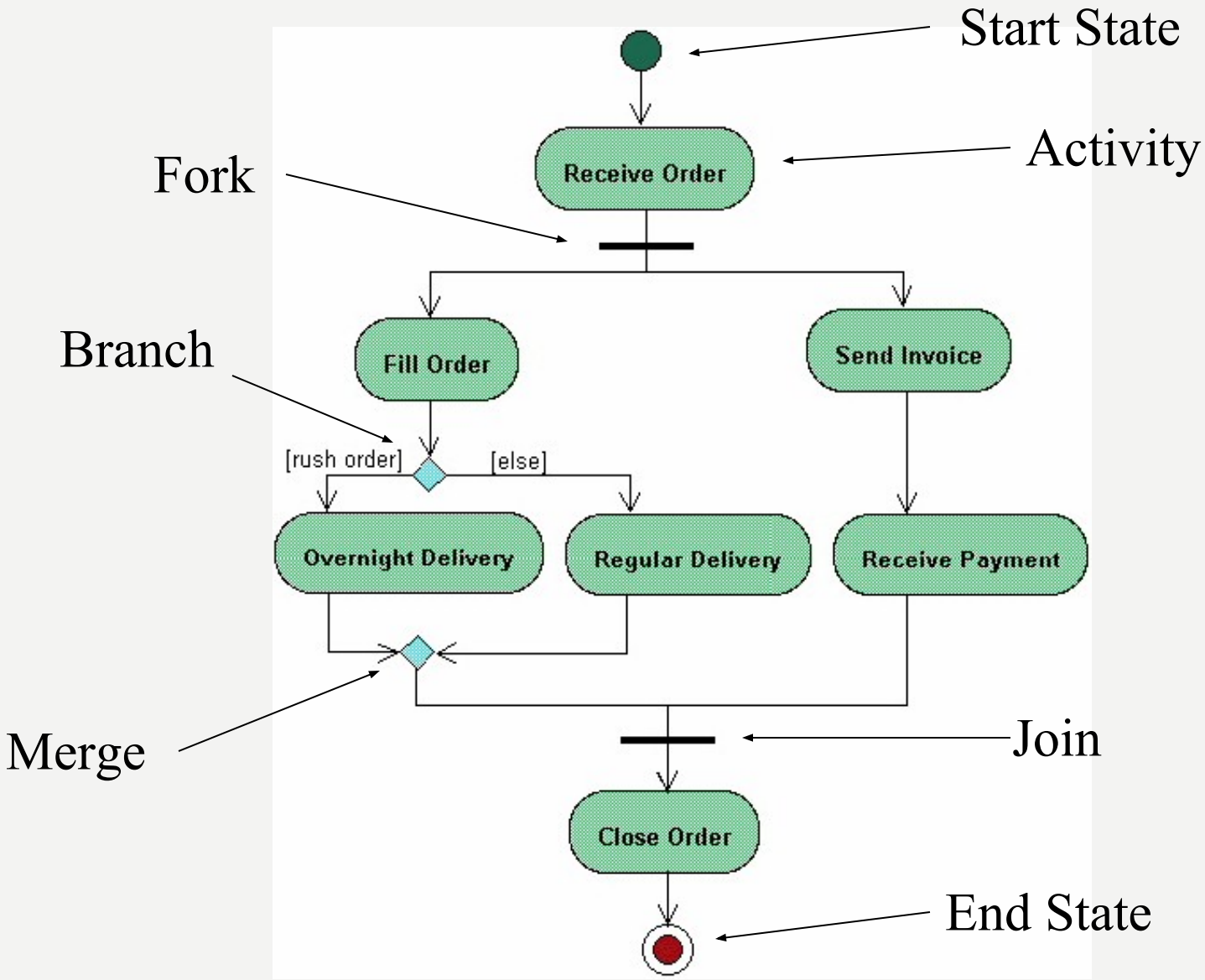
How to Draw an activity diagram –

1. Identify the initial state and the final states.
2. Identify the intermediate activities needed to reach the final state from he initial state.
3. Identify the conditions or constraints which cause the system to change control flow.
4. Draw the diagram with appropriate notations.

Uses of an Activity Diagram –

- Dynamic modelling of the system or a process.
- Illustrate the various steps involved in a UML use case.
- Model software elements like methods,operations and functions.
- We can use Activity diagrams to depict concurrent activities easily.
- Show the constraints, conditions and logic behind algorithms.

ACTIVITY DIAGRAM EXAMPLE

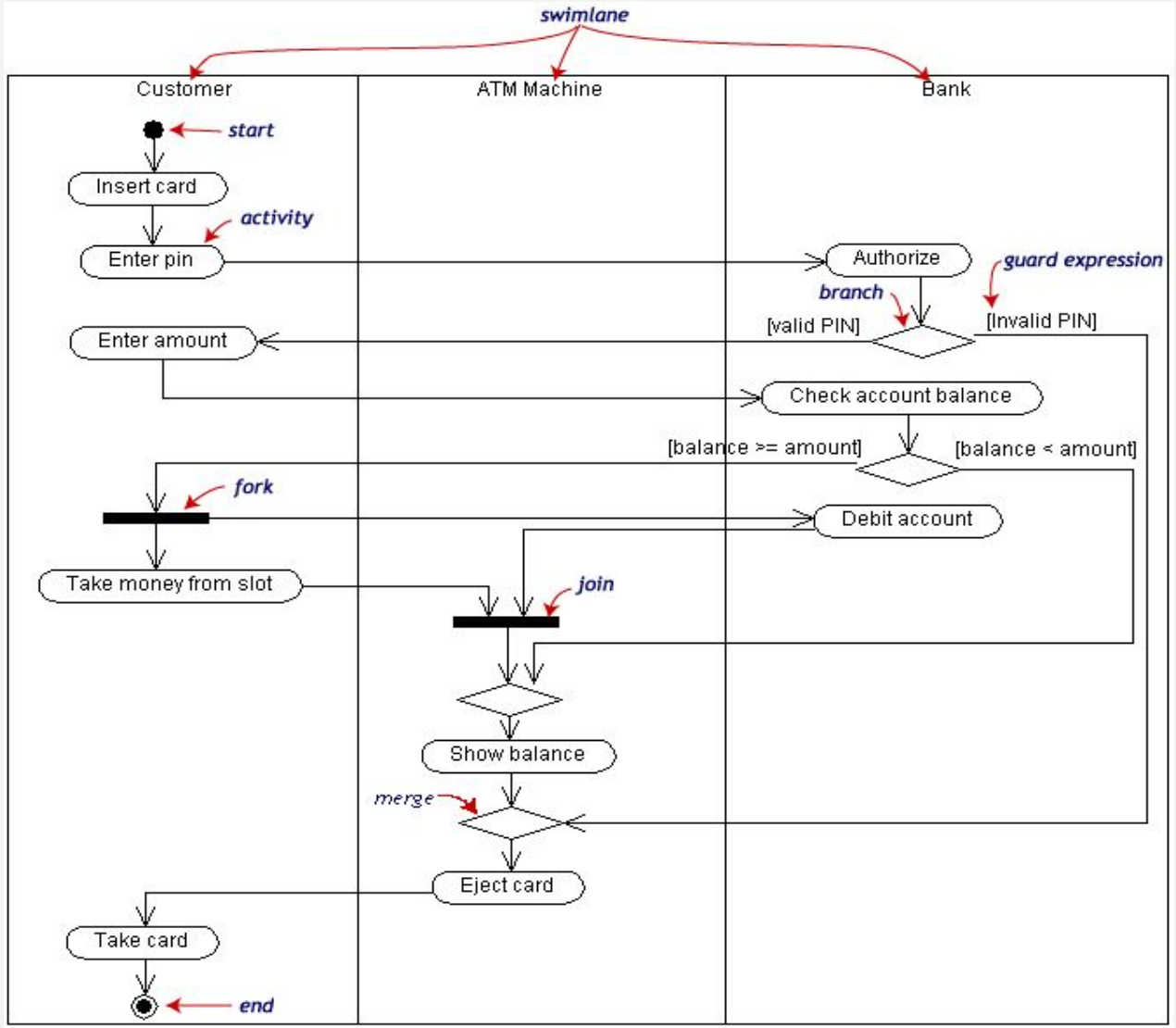


USE CASE

- Withdraw money from a bank account through an ATM

SWIMLANE

- Allows the modeler to represent the flow of activities described by the use-case and at the same time indicate which actor (if there are multiple actors involved in a specific use-case) or analysis class has responsibility for the action described by an activity rectangle


swimlane
Customer
ATM Machine
Bank
start
Insert card
activity
Enter pin
Authorize
guard expression
branch
[valid PIN]
[Invalid PIN]
Enter amount
Check account balance
[balance >= amount]
[balance < amount]
fork
Debit account
Take money from slot
join
Show balance
merge
Eject card
Take card
end

REFERENCES

- Activity Diagrams
- http://pigseye.kennesaw.edu/~dbraun/csis4650/A&D/UML_tutorial/activity.htm
- http://isds.bus.lsu.edu/cvoc/learn/bpr/cprojects/spring1998/modeling/activity.html
- http://www-106.ibm.com/developerworks/rational/library/2802.html