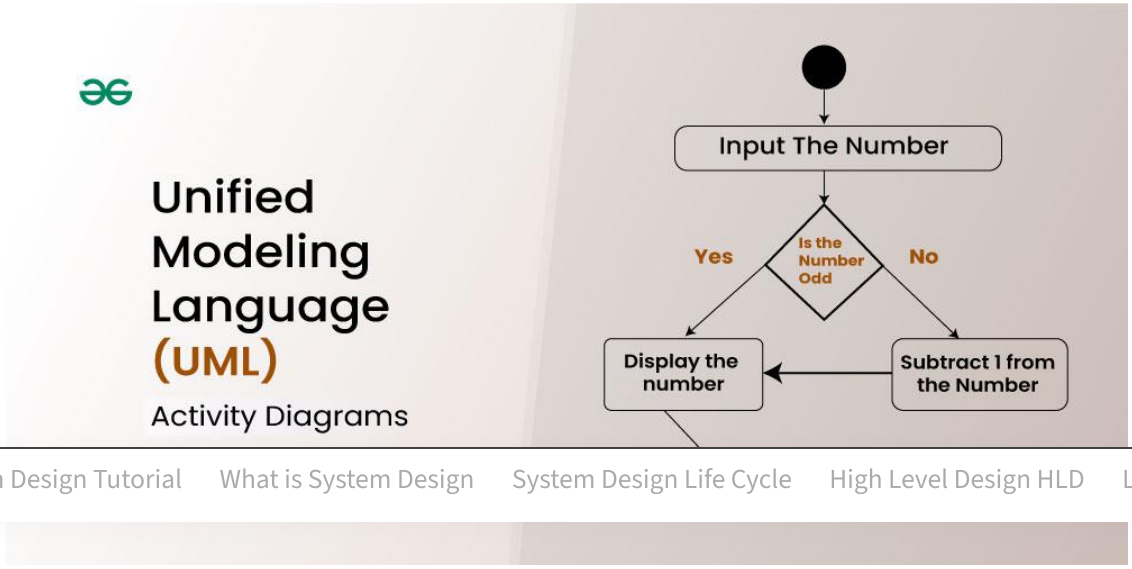
Activity Diagrams | Unified Modeling Language (UML)

Activity Diagrams are used to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. It is a type of behavioral diagram and we can depict both sequential processing and concurrent processing of activities using an activity diagram ie an activity diagram focuses on the condition of flow and the sequence in which it happens.



Important Topics for the Activity Diagrams

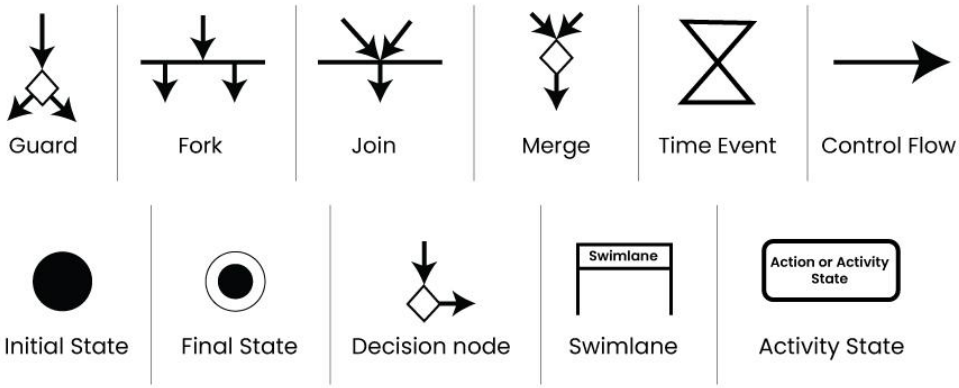
- What is an Activity Diagram?
- Activity Diagram Notations
- How to Draw an Activity Diagram in UML?
- What are Activity Diagrams used for?
- What are the Differences between an Activity diagram and a Flowchart?

1. What is an Activity Diagram?

Activity Diagrams are used to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. We can depict both sequential processing and concurrent processing of activities using an activity diagram ie an activity diagram focuses on the condition of flow and the sequence in which it happens.

- We describe what causes a particular event using an activity diagram.
- 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.
- They are used in business and process modeling where their primary use is to depict the dynamic aspects of a system.

2. Activity Diagram Notations

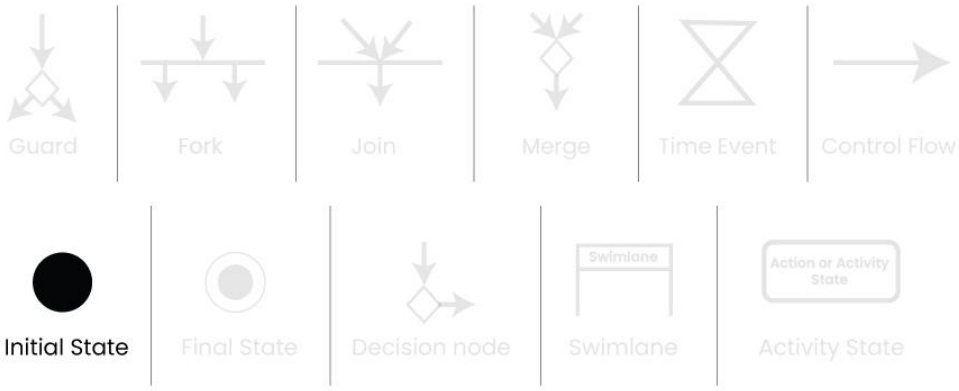


Activity Diagram Notations

In activity diagrams, the notations are like visual symbols that help represent different elements and actions in a simple way.

2.1. Initial State

The starting state before an activity takes place is depicted using the initial state.



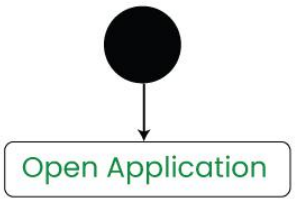
Initial State

A process can have only one initial state unless we are depicting nested activities. We use a black filled circle to depict the initial state of a system. For objects, this is the state when they are instantiated. The Initial State from the UML Activity Diagram marks the entry point and the initial Activity State.

For example:

Here the initial state of the system before the application is opened.

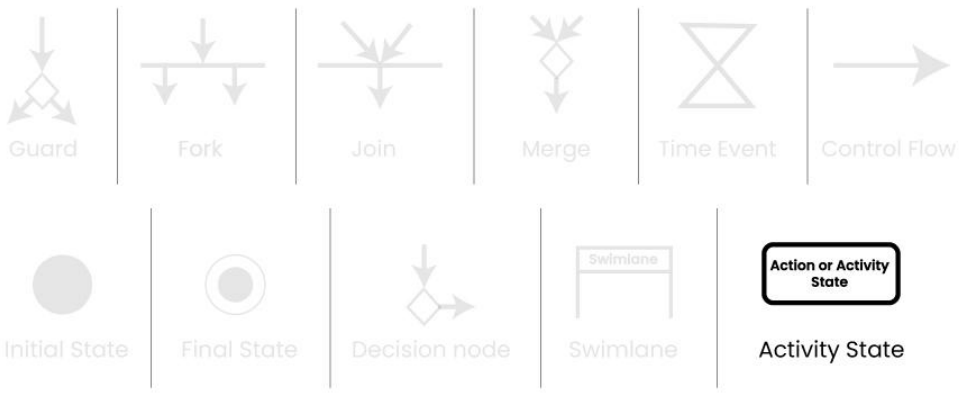
Initial State symbol being used



Unified Modeling Language (UML) | Activity Diagrams

2.2. 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.

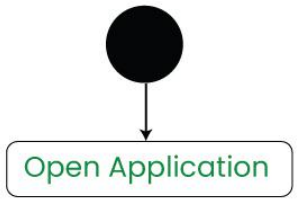


Activity State

For example:

Consider the previous example of opening an application, opening the application is an activity state in the activity diagram.

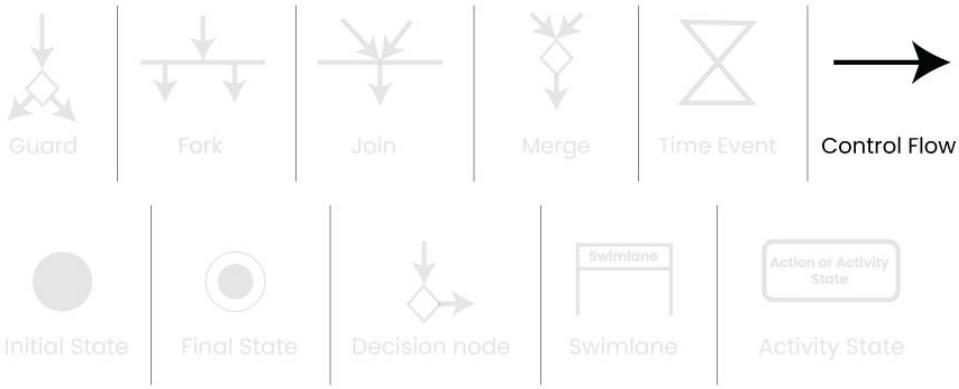
Activity State symbol being used



Unified Modeling Language (UML) | Activity Diagrams

2.3. 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 activity state.

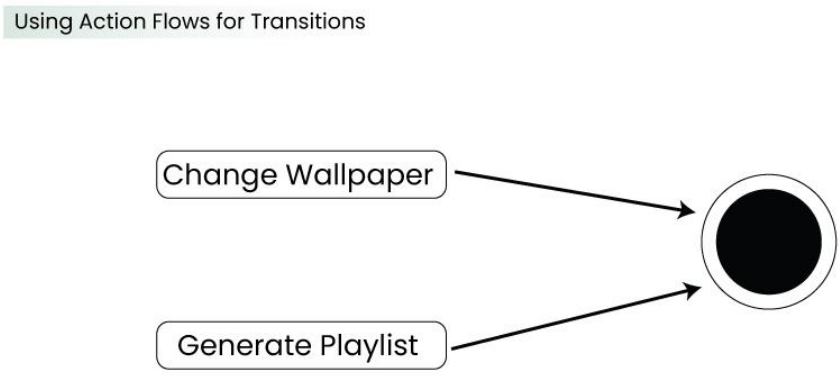


Control Flow

An activity state can have multiple incoming and outgoing action flows. We use a line with an arrow head to depict a Control Flow. If there is a constraint to be adhered to while making the transition it is mentioned on the arrow.

For example:

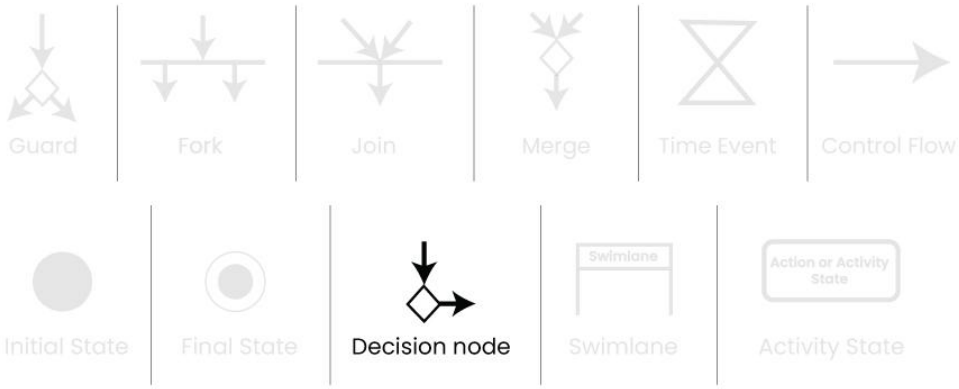
Here both the states transit into one final state using action flow symbols i.e. arrows.



Unified Modeling Language (UML) | Activity Diagrams

2.4. Decision node and Branching

When we need to make a decision before deciding the flow of control, we use the decision node. The outgoing arrows from the decision node can be labelled with conditions or guard expressions. It always includes two or more output arrows.



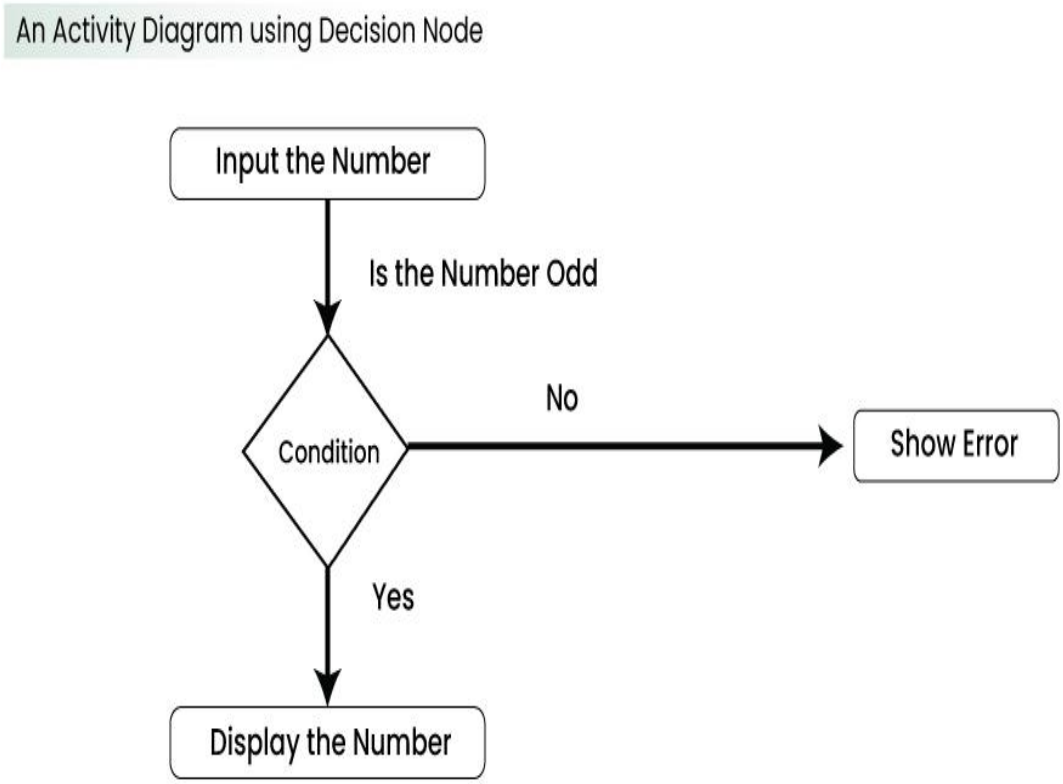
Decision Node

For example:

We apply the conditions on input number to display the result :

- *If number is odd then display the number.*

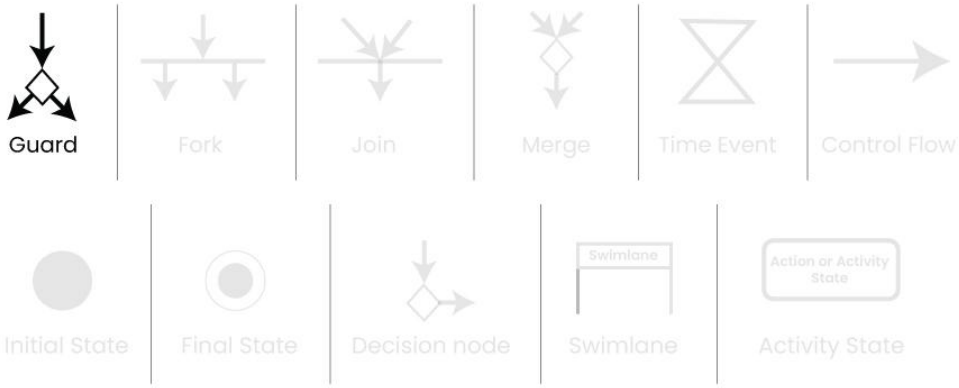
- *If number if even then display the error.*



Unified Modeling Language (UML) | Activity Diagrams

2.5. Guard

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



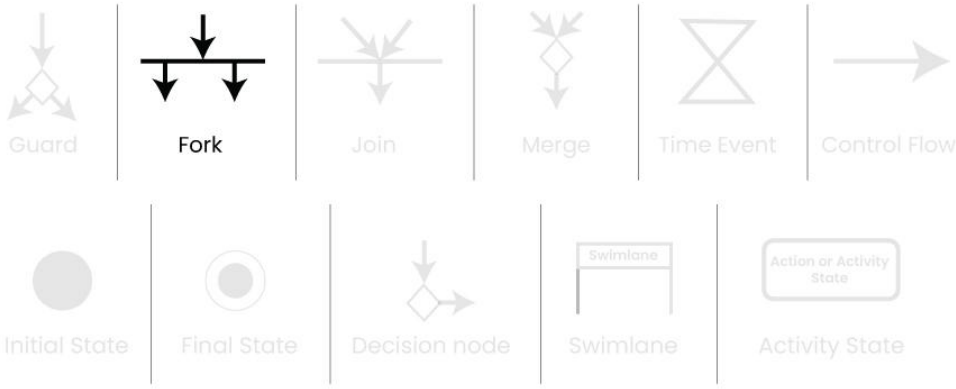
Guard

The statement must be true for the control to shift along a particular direction. Guards help us know the constraints and conditions which

determine the flow of a process.

2.6. 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.

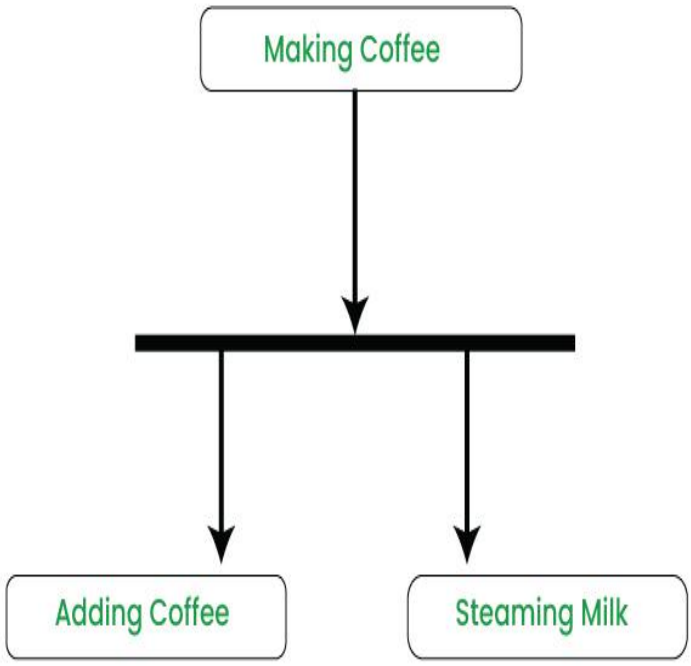


Fork

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.

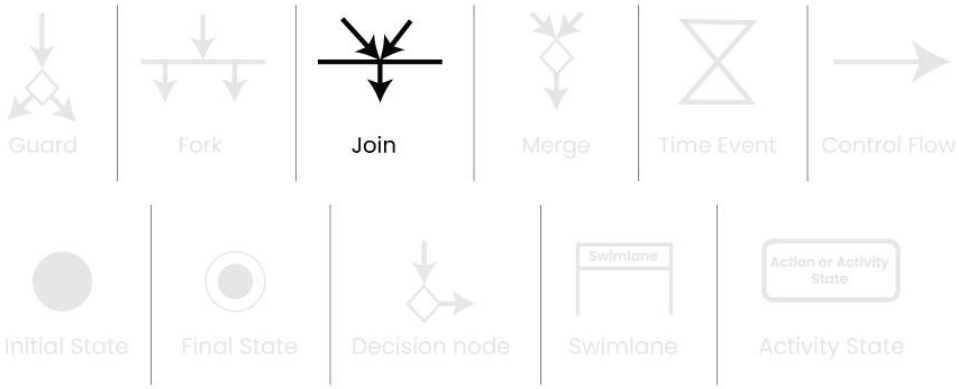
A Diagram using Fork Notation



Unified Modeling Language (UML) | Activity Diagrams

2.7. 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.

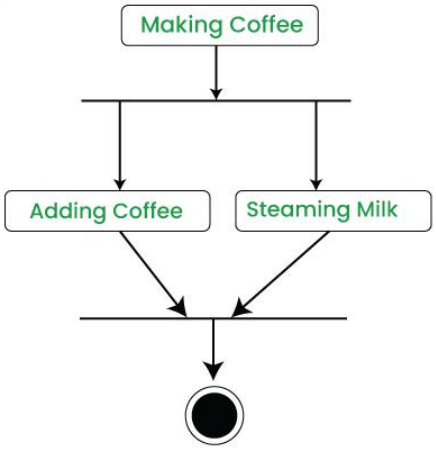


Join

For example:

When both activities i.e. steaming the milk and adding coffee get completed, we converge them into one final activity.

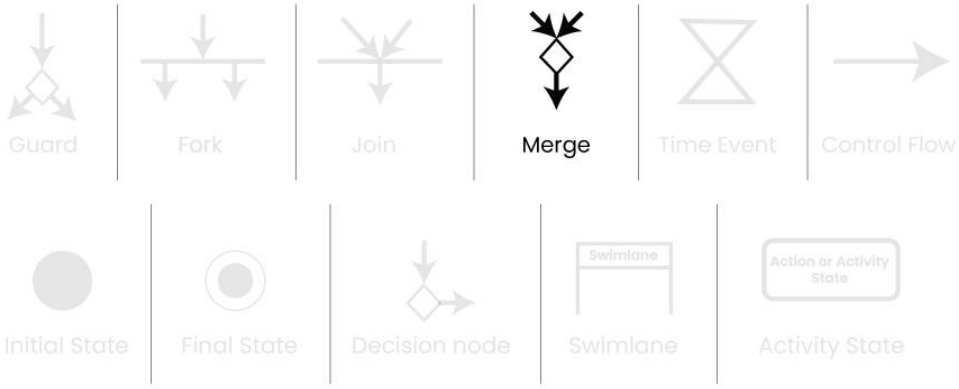
A Diagram using Join Notation



Unified Modeling Language (UML) | Activity Diagrams

2.8. 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.

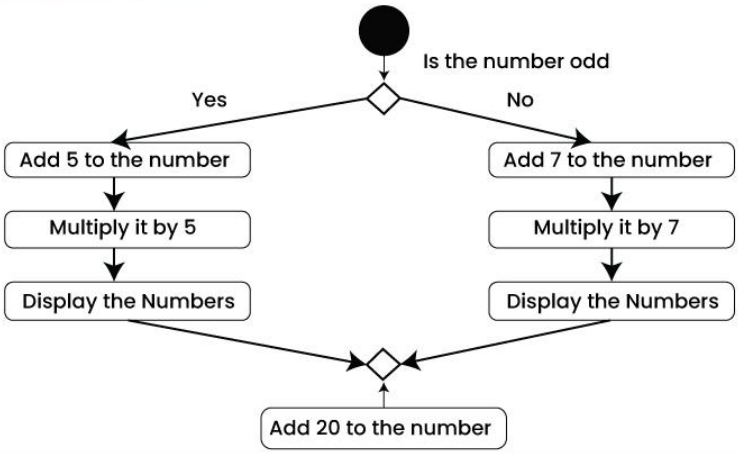


Merge

For example:

> *In the diagram below: we can't have both sides executing concurrently, but they finally merge into one. A number can't be both odd and even at the same time.*

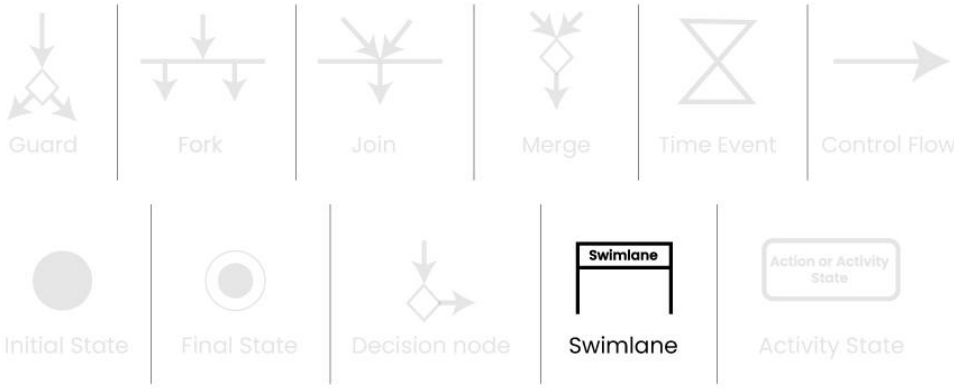
An Activity Diagram using Merge Notation



Unified Modeling Language (UML) | Activity Diagrams

2.9. 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.



Swimlane

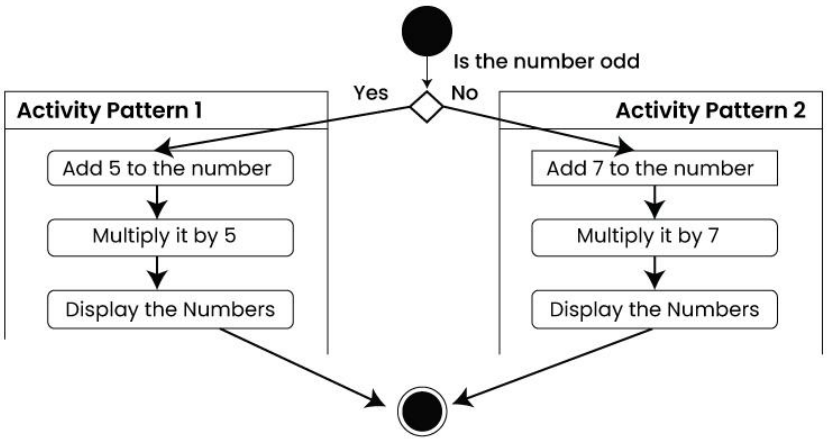
We use a rectangular column to represent a swimlane as shown in the figure above.

For example:

Here different set of activities are executed based on if the number is

odd or even. These activities are grouped into a swimlane.

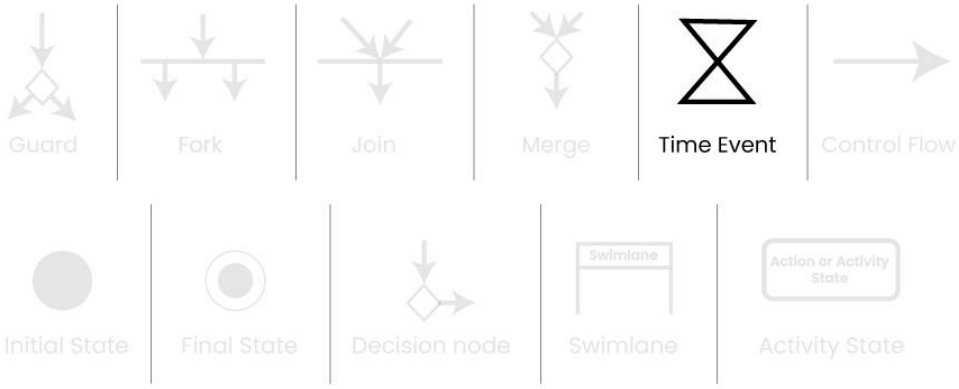
An Activity Diagram making use of Swimlanes



Unified Modeling Language (UML) | Activity Diagrams

2.10. Time Event

This refers to an event that stops the flow for a time; an hourglass depicts it. We can have a scenario where an event takes some time to completed.

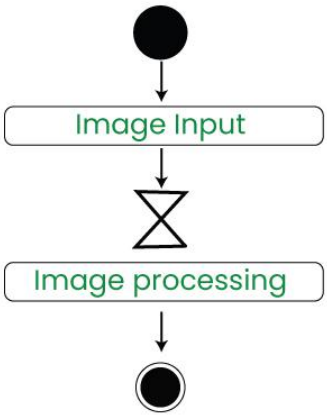


Time Event

For example:

Let us assume that the processing of an image takes a lot of time. Then it can be represented as shown below.

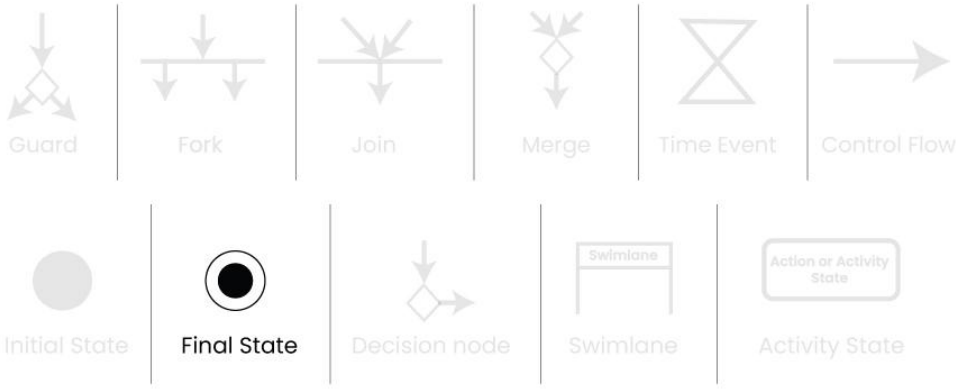
An Activity Diagram using Time Event Notation



Unified Modeling Language (UML) | Activity Diagrams

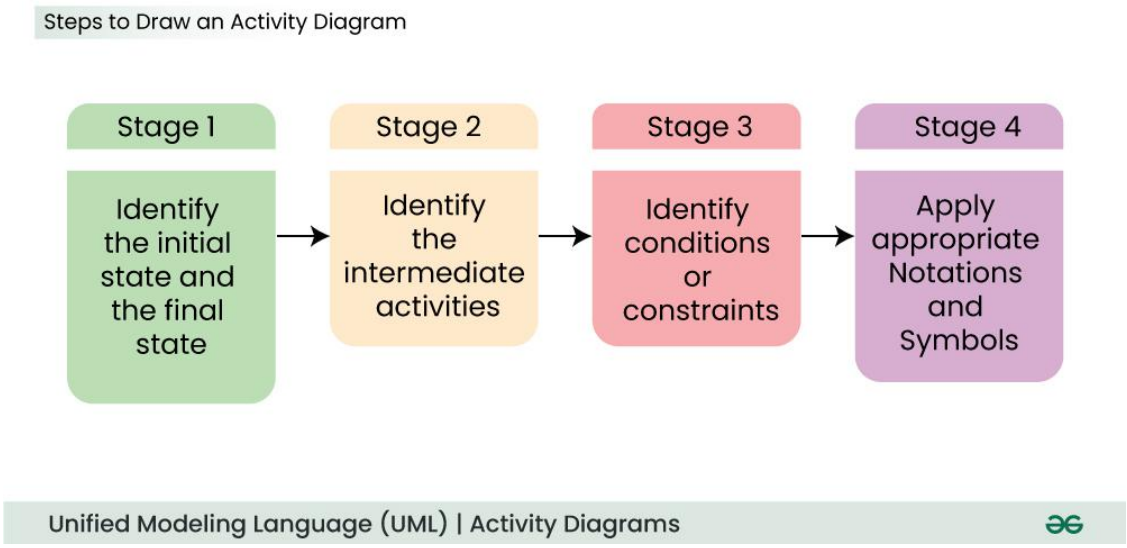
2.11. Final State or End State

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.



Final State

3. How to Draw an Activity Diagram in UML?



Below are the steps of how to draw the Activity Diagram in UML:

Step 1. Identify the Initial State and Final States:

- This is like setting the starting point and ending point of a journey.
- Identify where your process begins (initial state) and where it concludes (final states).
- For example, if you are modelling a process for making a cup of tea, the initial state could be “No tea prepared,” and the final state could be “Tea ready.”

Step 2. Identify the Intermediate Activities Needed:

- Think of the steps or actions required to go from the starting point to the ending point.
- These are the activities or tasks that need to be performed.
- Continuing with the tea-making , intermediate activities could include “Boil water,” “Pour tea into a cup”.

Step 3. Identify the Conditions or Constraints:

- Consider the conditions or circumstances that might influence the flow of your process.
- These are the factors that determine when you move from one activity to another.
- Using the tea-making scenario, a condition could be “Water is boiled,” which triggers the transition to the next activity.

Step 4. Draw the Diagram with Appropriate Notations:

- Now, represent the identified states, activities, and conditions visually using the appropriate symbols and notations in an activity diagram. This diagram serves as a visual map of your process, showing the flow from one state to another.

4. What are Activity Diagrams used for?

Activity diagrams are used in software development and system design to model and visualize the dynamic aspects of a system. Here are some common uses of activity diagrams:

- 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.
- During the requirements analysis phase, activity diagrams assist in capturing and documenting the dynamic aspects of user interactions.

5. What are the Differences between an Activity diagram and a Flowchart?

An activity diagram is very similar to a flowchart. So let us understand if activity diagrams or flowcharts are any different.

What is a Flow Chart?

An algorithm is like a set of clear instructions to solve a problem, and a flowchart is a picture that shows those instructions.

- When we're writing computer programs, a flowchart helps us map out the steps of the algorithm to solve the problem.
- Non programmers use Flow charts to model workflows.
- We can call a flowchart a primitive version of an activity diagram.
- Business processes where decision making is involved is expressed using a flow chart.

Example:

A manufacturer uses a flow chart to explain and illustrate how a particular product is manufactured.

What are the differences?

Activity Diagram	Flow Chart
An activity diagram is associated with the UML(Unified Modelling Language)	A Flow Chart is associated with the programming.
An activity diagram is used to model the dynamic aspects of a system and also illustrates the workflow of activities within a use case or business process.	Depicts a diagrammatic representation illustrating a solution model to a given problem and a flow chart converges into being an activity diagram if complex decisions are being made.
Commonly used in software engineering within the UML for modeling and designing software systems on high level.	Widely used in software engineering for representing algorithms, decision structures, and program flows.

Do we need to use both the diagrams and the textual documentation?

Let's understand this with the help of an example:

- Different individuals have different preferences in which they understand something.
- To understand a concept, some people might prefer a written tutorial with images while others would prefer a video lecture.
- So we generally use both the diagram and the textual documentation to make our system description as clear as possible.

6. Conclusion

In conclusion, Activity Diagrams serve as invaluable tools in system design and analysis, offering a visual representation of dynamic processes within organizations. They are widely utilized to model business processes, illustrate user interactions, and guide software system design. By providing a clear and concise overview of activities, decision points, and interactions, activity diagrams enhance communication among project stakeholders and contribute to effective documentation.

Last Updated : 15 Jan, 2024

State Machine Diagrams | Unified Modeling Language (UML)

Sequence Diagrams | Unified Modeling Language (UML)

Object Diagrams | Unified Modeling Language (UML)

Use Case Diagrams | Unified Modeling Language (UML)

Structural Diagrams | Unified Modeling Language(UML)

Interaction Overview Diagrams | Unified Modeling Language (UML)

Unified Modeling Language (UML) Diagrams

Collaboration Diagrams | Unified Modeling Language(UML)

GeeksforGeeks

Article Tags : UML , Design Pattern , System Design

A-143, 9th Floor, Sovereign Corporate Tower, Sector-136, Noida, Uttar Pradesh - 201305