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

**Step1. Identify the System:**

* Understand what your diagram is representing.
* Whether it’s a machine, a process, or any object, know what different situations or conditions it might go through.

**Step2. Identify Initial and Final States:**

* Figure out where your system starts (initial state) and where it ends (final state).
* These are like the beginning and the end points of your system’s journey.

**Step3. Identify Possible States:**

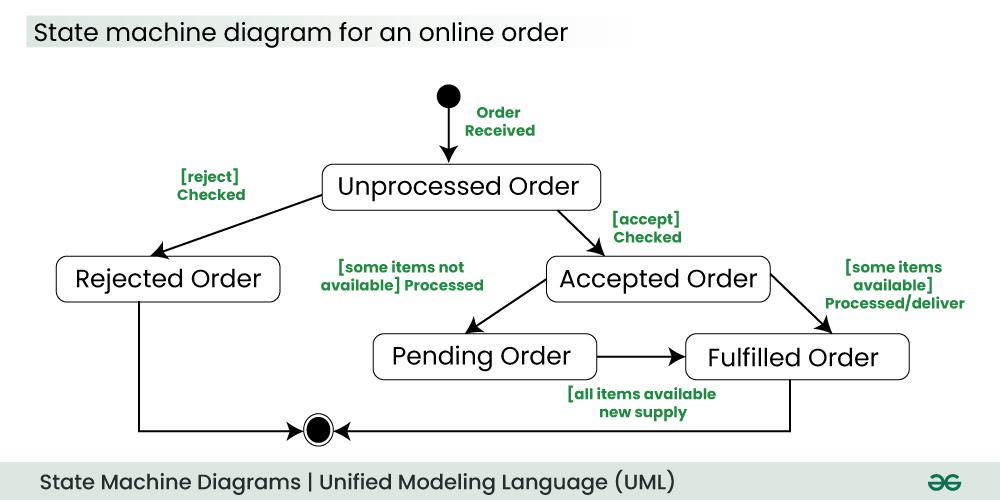
* Think about all the different situations your system can be in.
* These are like the various phases or conditions it experiences.
* Use boundary values to guide you in defining these states.

**Step4. Label Triggering Events:**

* Understand what causes your system to move from one state to another.
* These causes or conditions are the events.
* Label each transition with what makes it happen.

**Step5. Draw the Diagram with appropriate notations:**

* Now, take all this information and draw it out.
* Use rectangles for states, arrows for transitions, and circles or rounded rectangles for initial and final states.
* Be sure to connect everything in a way that makes sense.

Let’s understand State Machine diagram with the help of an example, ie for an online order :  


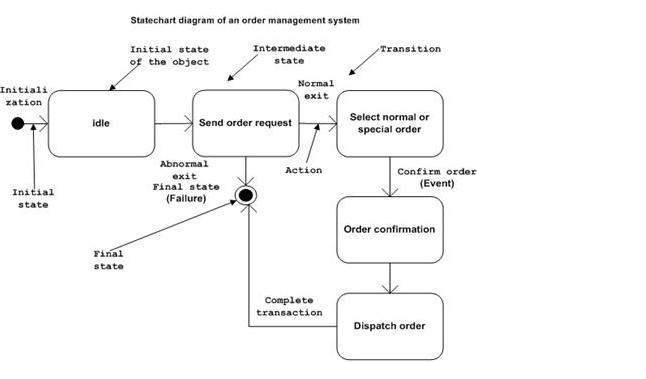
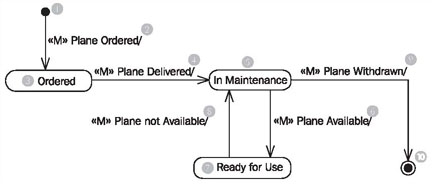
The UML diagrams we draw depend on the system we aim to represent. Here is just an example of how an online ordering system might look like :

1. On the event of an order being received, we transit from our initial state to Unprocessed order state.
2. The unprocessed order is then checked.
3. If the order is rejected, we transit to the Rejected Order state.
4. If the order is accepted and we have the items available, we transit to the fulfilled order state.
5. However, if the items are not available we transit to the Pending Order state.
6. After the order is fulfilled, we transit to the final state. In this example, we merge the two states i.e. Fulfilled order and Rejected order into one final state.

**Note:**Here we could have also treated fulfilled order and rejected order as final states separately.

**4. Use Cases of State Machine Diagram**

* State Machine Diagrams are particularly useful for modelling and visualizing the dynamic behaviour of a system.
* They showcase how a system responds to events and transitions between different states.
* We use it to state the events responsible for change in state (we do not show what processes cause those events).
* Objects go through different states during their existence, and these diagrams help in illustrating these states and the transitions between them.
* In embedded systems, where hardware interacts with software to perform tasks, State Machine Diagrams are valuable for representing the control logic and behaviour of the system.



Above diagram shows all states that the **object plane** can be in during the course of its life. Furthermore, it shows the possible transitions between the states and the events that initiate these transitions.

Each object of the class plane comes from nowhere (1) (initial state) and disappears (generally) again, into nothing (10) (final state). This usually holds true for all classes, meaning in most classes you will find an initial state (1) and a final state (10).Over the course of its life, an plane (please note: we are here talking about the object plane and not about a real airplane) can take up three states: ordered (3), in maintenance (5), and ready for use (7)

The event «M» plane ordered, leads to the occurrence, that from nowhere (1) a new plane object is created in the IT system (birth). Immediately after it has been created it is in the state ordered (3).

If the event «M» plane delivered (4) occurs, and the plane is the state ordered (3), it changes to the state in maintenance (5). If the plane is in any other state than ordered, nothing happens.

Through the events «M» plane available (6) and «M» plane not available (8), the plane changes any number of times between the states in maintenance (5) and ready for use (7).

At the end of its life, the airplane object disappears through the event «M» plane withdrawn (9) into nothing (10), meaning it will be deleted (death).