UML Diagrams

State Machine Diagram

Linda Marshall

Department of Computer Science
University of Pretoria

7 September 2021

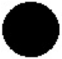
A state is a condition in which an object can be at some point during its lifetime, for some finite period of time. A state diagram is:

- used to model critical states within a system and the events that may trigger a change in state (mostly attribute values).
- good for describing the behaviour of one object over time.

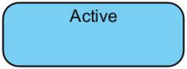
A UML State diagram consists of nodes and edges. *Nodes* are classified as State nodes or Control nodes. *Edges*, more commonly referred to as *transitions*, are labelled arrows connecting the nodes.

There are three kinds of *state* nodes:

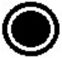
- Initial node: The starting point of a state diagram. There may be at most one on a diagram.



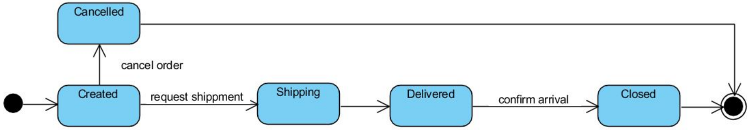
- State node: Used to indicate a state. It should be named. It may contain another state diagram.



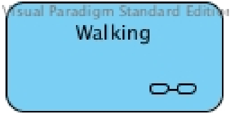
- End node: Indicates where the system terminates. Many final nodes may be present on a diagram.



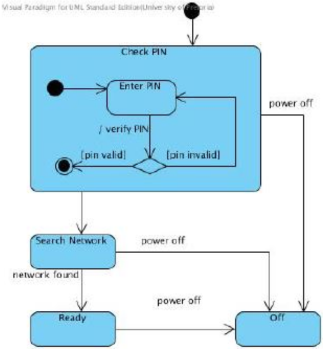
Example

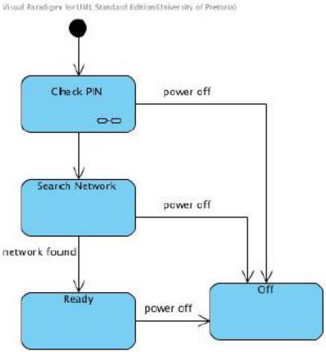

Cancelled
cancel order
Created
request shippment
Shipping
Delivered
confirm arrival
Closed

A composite state node is a special type of state node containing another state diagram. composite state nodes may be named, or be left anonymous. If they are anonymous, detailed flow needs to be modelled.



Examples of composite states

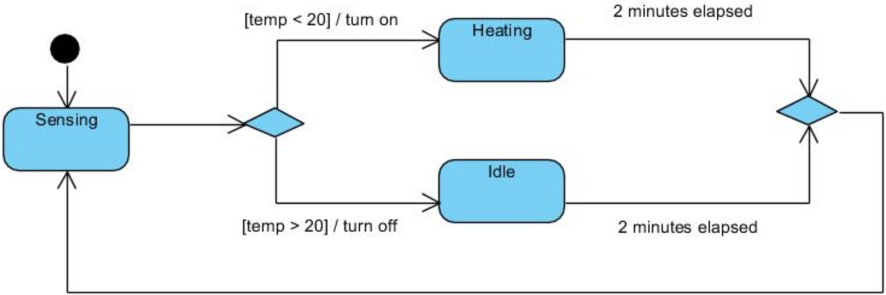




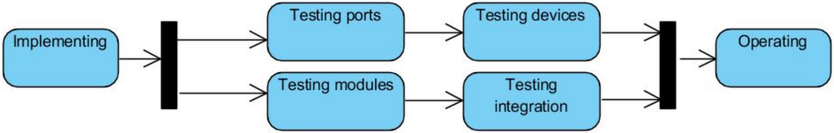
There are two kinds of control nodes:

- *Decision-and-Merge* nodes to model alternate flows. Guard conditions are used to indicate when an alternative should be chosen. A diamond is used to model a decision at the beginning of the alternate flows and at the end of the alternate flows when they merge again.
- *Fork-and-Join* nodes model parallel flows. The flows are executed at the same time. Only when all flows reach the join (synchronisation) point, does the flow in the state diagram move on. Fork and join nodes are indicated by heavy vertical or horizontal lines

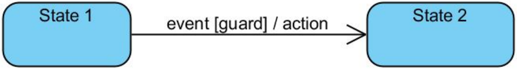
Example: Heating device with alternate flows



Example: Stages of a system with parallel flows



Transitions (edges) indicate transition between states.



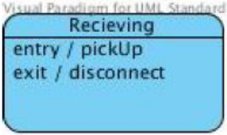
The transition between states:

- Should indicate the event that triggers the transition (text)
- May have a guard condition that is a prerequisite for transition (text in [])
- May have an action that is executed during transition (name of operation to be called after /)

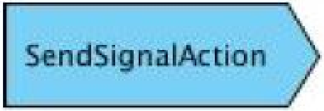
- If a transition is shown without an event it triggers immediately.
- If an event happens for which there is no transition, nothing happens. The object remains in the current state.

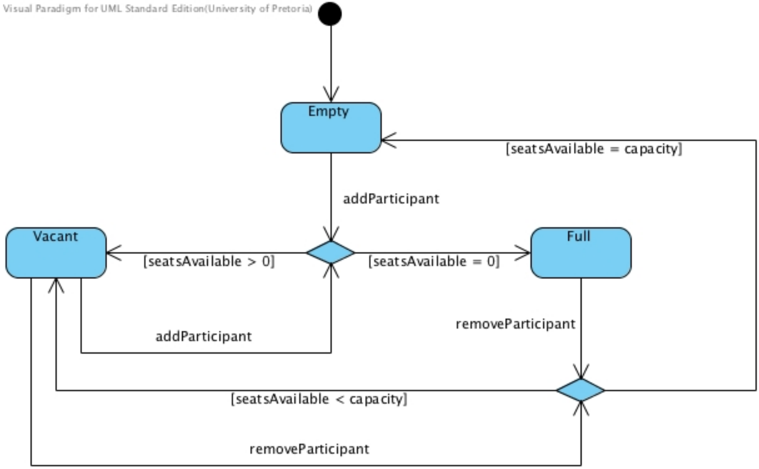
Note: Guard conditions are required for Alternate Flows.

Actions are generally executed during transition. It is possible to indicate actions that are executed while the system is in a state. Such actions are triggered on entering a state or when leaving a state.



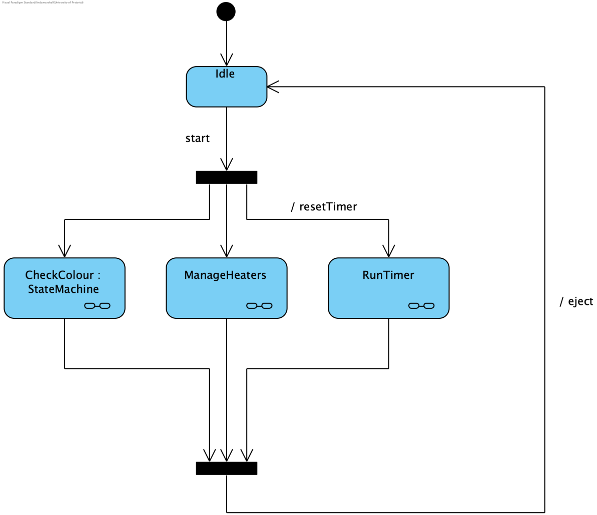
Events generated while actions are executed by the system are referred to as signals and are modelled using a signal node.




Visual Paradigm for UML Standard Edition(University of Pretoria)
Empty
[seatsAvailable = capacity]
addParticipant
Vacant
[seatsAvailable > 0]
[seatsAvailable = 0]
Full
addParticipant
removeParticipant
[seatsAvailable < capacity]
removeParticipant

Draw a UML State diagram to model the states of an intelligent toaster. The toaster is in the idle state until it is put to use. After use, it returns to idle. When being used, three processes run in parallel:

1. the colour of the toast is checked using two sensors, an absolute colour check and a sensor to check the rate of change in the colour
2. the elements are switched on and off to ensure the temperature is kept constant
3. a timer is set to indicate the duration of toasting


Idle
start
/ resetTimer
CheckColour : StateMachine
ManageHeaters
RunTimer
/ eject