# Object Oriented Analysis and Design using Java

**Prof: Mahitha G**

Department of Computer Science and Engineering

Leveraging information from slides of  Dr H L Phalachandra

# Object Oriented Analysis and Design using Java

## Behavioral Models

### (State Diagram)

**Mahitha G**

Department of Computer Science and Engineering

Object Oriented Analysis and Design using Java

## Unit-03 : Dynamic Models, Diagrams and Architecture design and principles

# Class-26 & 27 Behavior Modelling: UML State Machine Diagrams and Models

## Generics of State diagram

- We now discuss the dynamic behavior of objects over time by modelling the lifecycles of objects of each class over the objects life span represented as the **state diagram**
- In the state diagram we consider
  - Each object treated as an isolated entity that communicates with the rest of the world by detecting events and responding to them.
  - Events representing the different changes that objects can detect... anything that can affect an object can be characterized as an event
- The state of an object is a condition or a situation during the life of an object during which it satisfies some condition, performs some activity or waits for an event
- When an event occurs some activity will take place based on the current state of the object
- Activity which is an ongoing non atomic execution within the state machine, results in some action which could be made of some atomic computation that results in the change in the state of the model or a return of a value

## State diagram

State diagram can be visualized as representation of potential states of the objects and the transitions among the those states

Few points which are inherently true for these would be

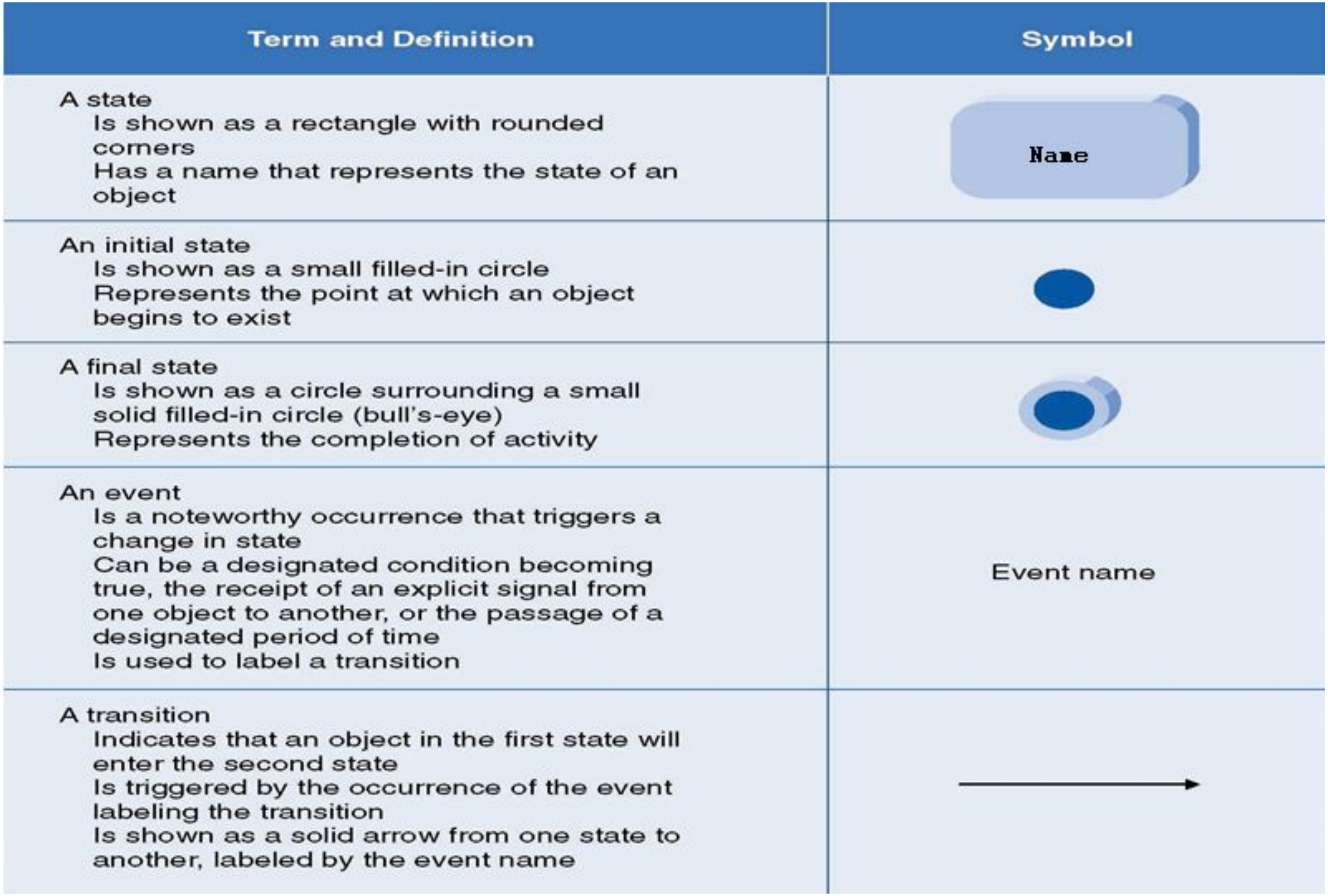
An object must be in some specific state at any given time during its lifecycle.

An object transitions from one state to another as a result of some event that affects it.

There can be only one start state in a state diagram, but there may be many intermediate and final states

We take the approach of looking at the different terms and concepts which are part of this and look at state diagrams

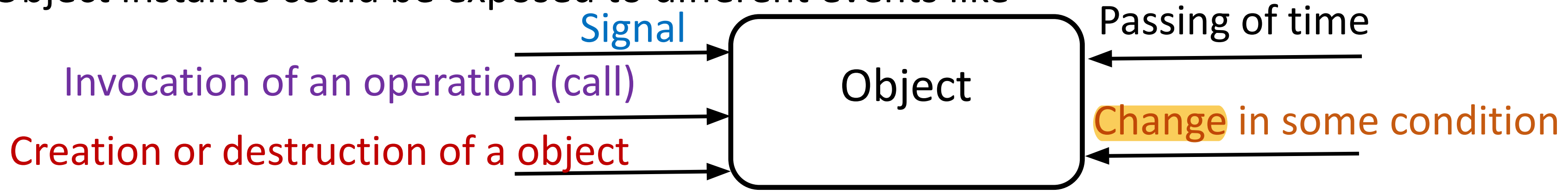
## State diagram – UML Symbols



| Term and Definition | Symbol |
|---|---|
| A state<br>Is shown as a rectangle with rounded corners<br>Has a name that represents the state of an object | Name |
| An initial state<br>Is shown as a small filled-in circle<br>Represents the point at which an object begins to exist | ● |
| A final state<br>Is shown as a circle surrounding a small solid filled-in circle (bull's-eye)<br>Represents the completion of activity | ◉ |
| An event<br>Is a noteworthy occurrence that triggers a change in state<br>Can be a designated condition becoming true, the receipt of an explicit signal from one object to another, or the passage of a designated period of time<br>Is used to label a transition | Event name |
| A transition<br>Indicates that an object in the first state will enter the second state<br>Is triggered by the occurrence of the event labeling the transition<br>Is shown as a solid arrow from one state to another, labeled by the event name | → |

## State Diagram Terms and Concepts : Events

- Object instance could be exposed to different events like



Signal

Invocation of an operation (call)

Creation or destruction of a object

Object

Passing of time

Change in some condition

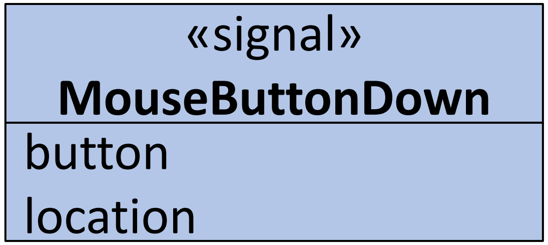
- An Event is an occurrence at a given point in time. It often corresponds to a verb in past sense or on the onset of a condition other E.g. User depresses a button, customer reports a problem, shipment arrives, phone taken off hook, paper try becomes empty, temperature falls below zero
- A event represents a named object that is dispatched (thrown) asynchronously by one object and then received (caught) by another
- Located in time and space, considered as Instantaneous occurrence, Duration unimportant – the fact that it has occurred is important
- These events could be
  - Normal as well as error conditions
  - External or internal (e.g. page loaded)
- Concurrent and Related Events
  - Related events if one event logically or causally follows another
  - If the events are causally unrelated they are concurrent events

## State Diagram Terms and Concepts : Events types – Signal Event

- Signal – One way transmission of information/message from one object to another
- An object sending a signal to another object may expect a reply, but the reply is a separate signal under the control of the second object, which may or may not choose to send it.
- ***Signal Event*** is the event of sending or receiving a signal
- Contrasting signal and signal event, signal is a message between objects, whereas the signal event is an occurrence in time

## State Diagram Terms and Concepts : Events types – Signal

- Signal occurrences are unique, but are grouped to ***signal classes*** with a class name to indicate common structure and behavior. They also have attributes indicating values they convey
- They are represented in UML as <<signal>> on top of class name

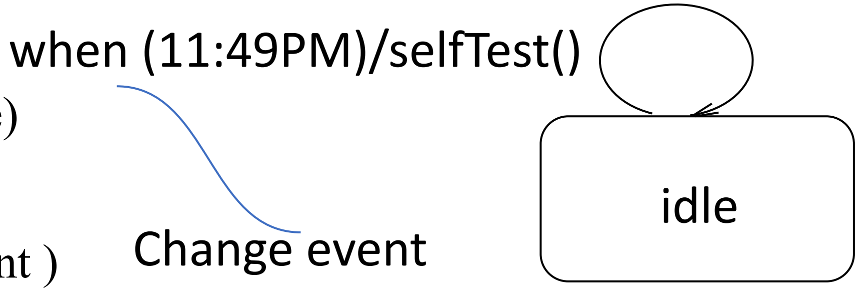


«signal»
**MouseButtonDown**

button
location

## State Diagram Terms and Concepts : Events types – Change Event

- Caused by satisfaction of Boolean expression
- Modeled using
  - when (expression)
- Expression below is continuously (practically) tested & whenever the expression evaluates to true the event happens
- Examples
  - when (inventory < reorder point)
  - when (storage used > maximum allowable)
  - when (altitude < 10000)
  - when (room temperature < heating set point )
  - when (room temperature > cooling set point )
  - when (battery power < lower limit )
  - when (tire pressure < minimum pressure )



when (11:49PM)/selfTest()

Change event

idle

- Generally detected by polling or other application logic

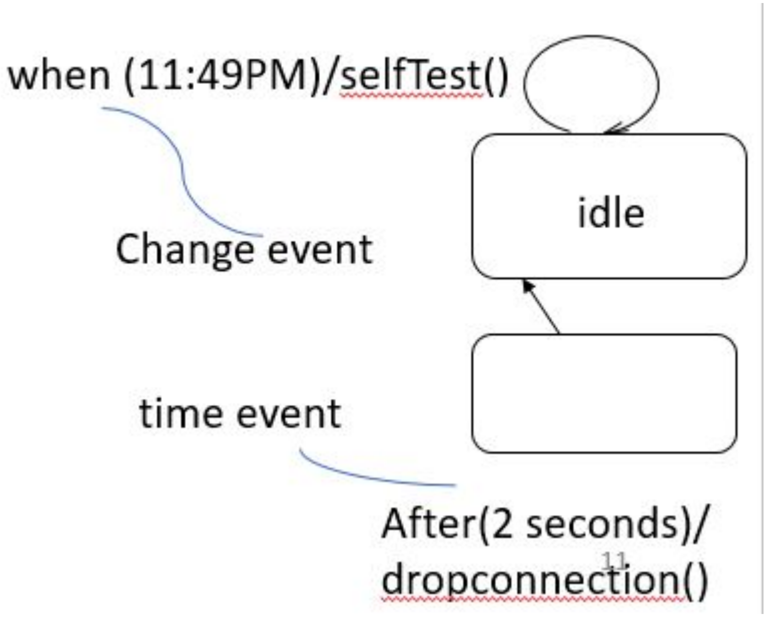
## State Diagram Terms and Concepts : Events types – Time Event

- Represents elapse of time or absolute time
- Modeled using
  - when (expression)
  - after (expression)
- UML notation for an absolute time is the keyword when followed by a parenthesized expression involving time.

  Eg:

  -- when (date = November 1, 2011)

  -- after (10 seconds)



when (11:49PM)/selfTest()

idle

Change event

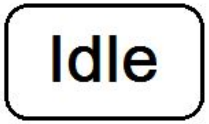
time event

After(2 seconds)/
dropconnection()

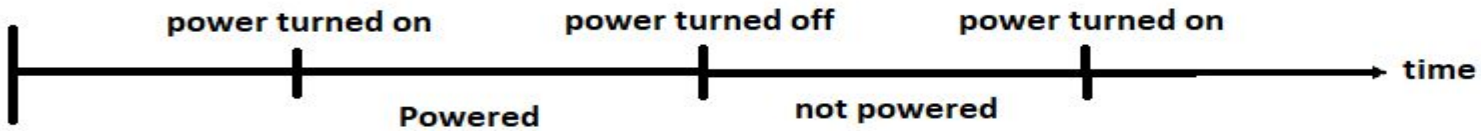
## State Diagram Terms and Concepts : State

- State is an abstraction of values and links of an object (some internal condition). Sets of values and links are grouped together into a state according to the gross behaviour of objects.

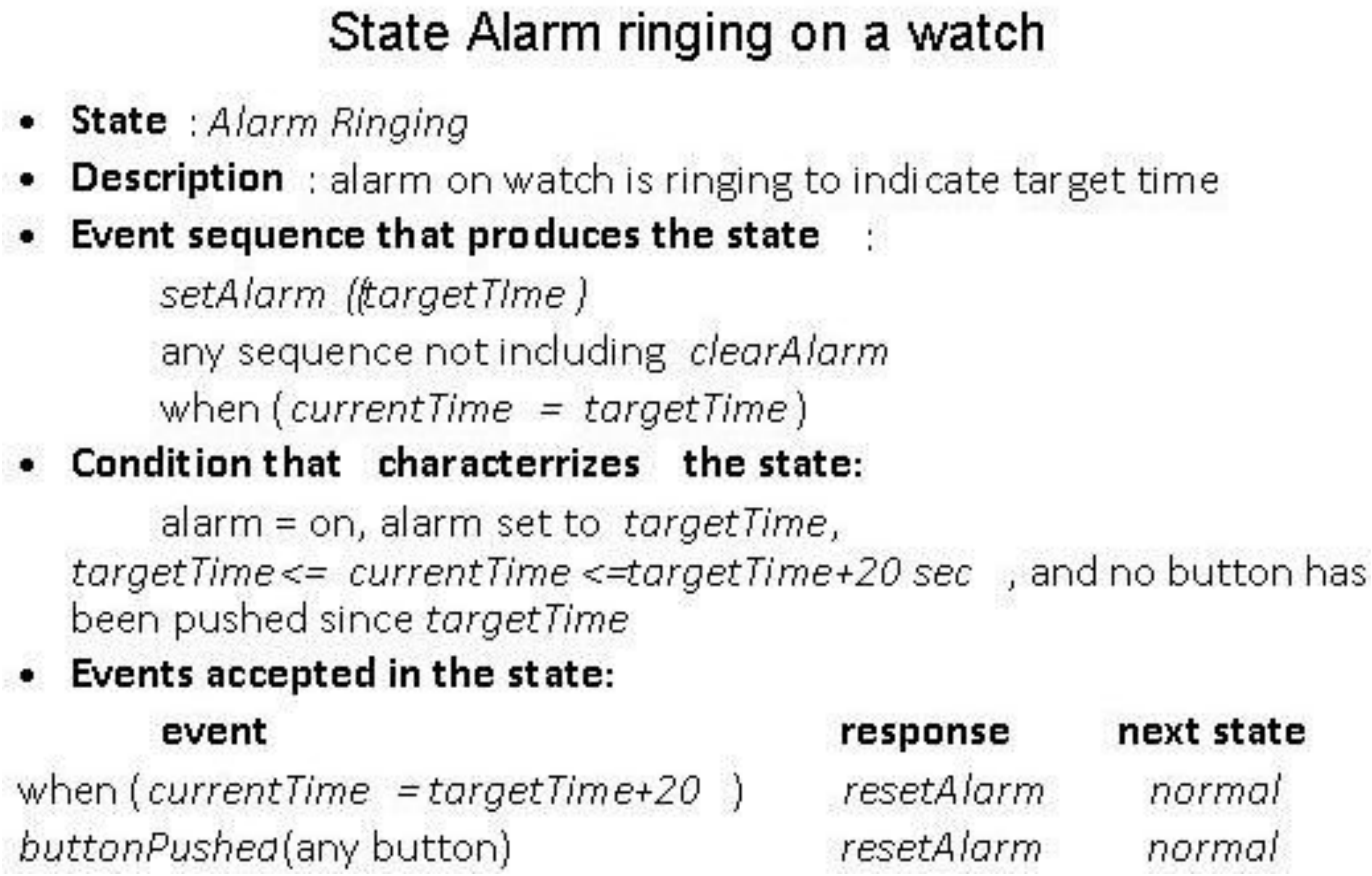
State is represented as



- States correspond to verbs with a suffix of “ing” (Waiting, Dialing) or the duration of some condition (Powered, BelowFreezing)
- Objects of a class have a finite number of possible states. Each object can be in one state at a time.
- State determines response of the object to input events. All inputs other than ones which are defined for are ignored
- State of the object depends on the past events which in most cases are eventually hidden by subsequent events
- Events corresponds to a specific points in time and states represent intervals of time between two events received by the object



## State Diagram Terms and Concepts : Characterizing States



### State Alarm ringing on a watch

- **State** : *Alarm Ringing*
- **Description** : alarm on watch is ringing to indicate target time
- **Event sequence that produces the state** :

  *setAlarm (targetTIme )*

  any sequence not including *clearAlarm*

  when (*currentTime = targetTime*)
- **Condition that characterrizes the state:**

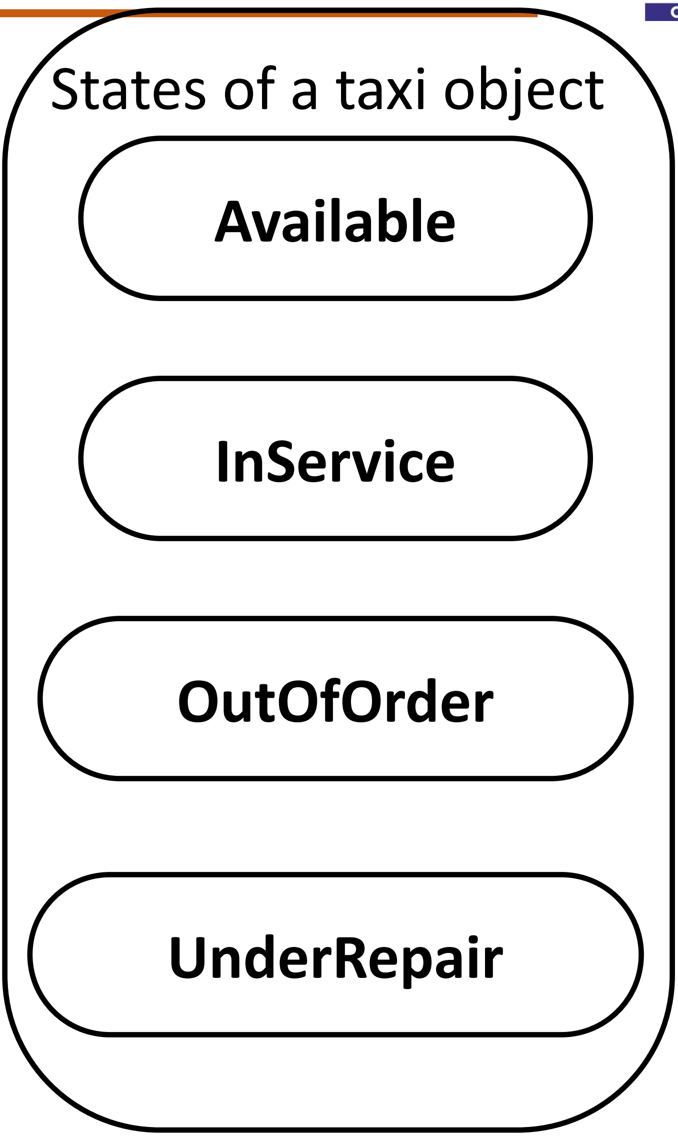
  alarm = on, alarm set to *targetTime*, *targetTime <= currentTime <=targetTime+20 sec* , and no button has been pushed since *targetTime*
- **Events accepted in the state:**

| event | response | next state |
|---|---|---|
| when (*currentTime =targetTime+20* ) | *resetAlarm* | *normal* |
| *buttonPushed*(any button) | *resetAlarm* | *normal* |

# State Diagram Terms and Concepts : Characterizing States

## States of a Taxi Object for Example

- Name
- Description
- Entry/Exit effects
    - Actions executed on entering and exiting the state
- Events accepted, not accepted, deferred for a state
    - Apply it to the example states here
- Sub-states
    - Apply it to the example states here



## State Diagram Terms and Concepts : Transition and Conditions

**Transition**: an instantaneous change in state from one state to another

- triggered by an event
- Transition is said to fire upon the change from source to target state
- Original and target state of a transition depends on the original state and could be different or the same.

e.g. when a phone line is answered, the phone line transitions from the ***Ringing*** state to the ***Connected*** state.

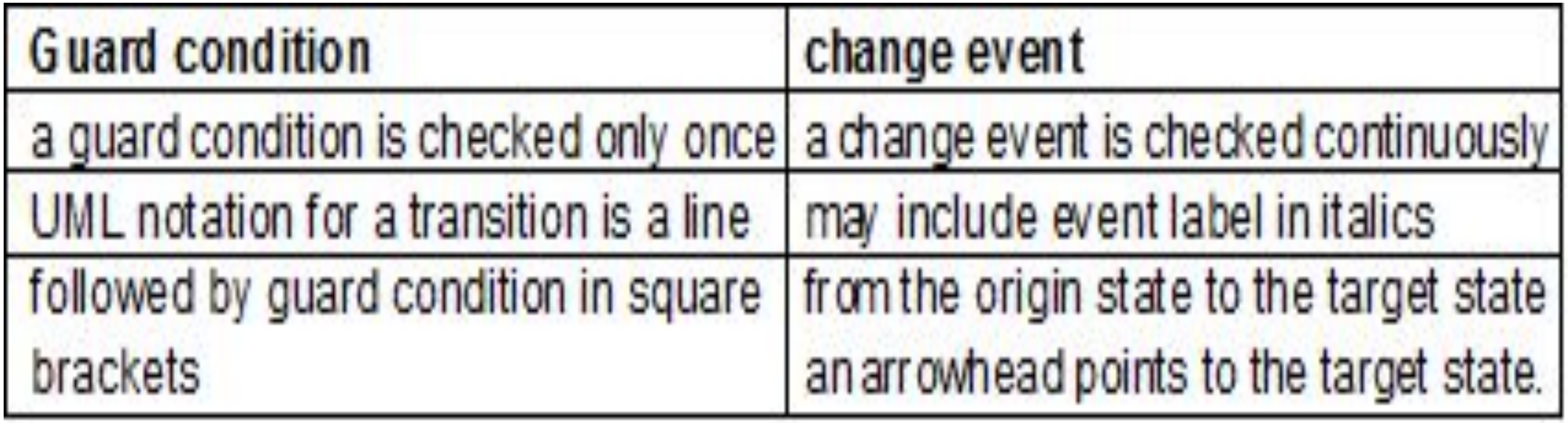
**Guard Condition**:

- boolean expression that must be true for transition to occur
- checked only once, at the time event occurs; transition fires if true

e.g. when you go out in the morning ***(event)***, if the temperature is below freezing ***(condition)***, then put on your gloves ***(next state)***.

**Guard Condition Example – Turnstile with coin**

# State Diagram : Guard Condition vs Change Event



| Guard condition | change event |
|---|---|
| a guard condition is checked only once | a change event is checked continuously |
| UML notation for a transition is a line | may include event label in italics |
| followed by guard condition in square brackets | from the origin state to the target state an arrowhead points to the target state. |

## State Diagram :

- State Diagrams show the sequences of states an object goes through during its life cycle in response to stimuli, together with its responses and actions;
- A state diagram is a graph whose nodes are states and whose directed arcs are transitions between states.
- It represents
  - State sequences caused by event sequences
  - State names must be unique within the scope of a state diagram
  - Common behavior of all instances of the class

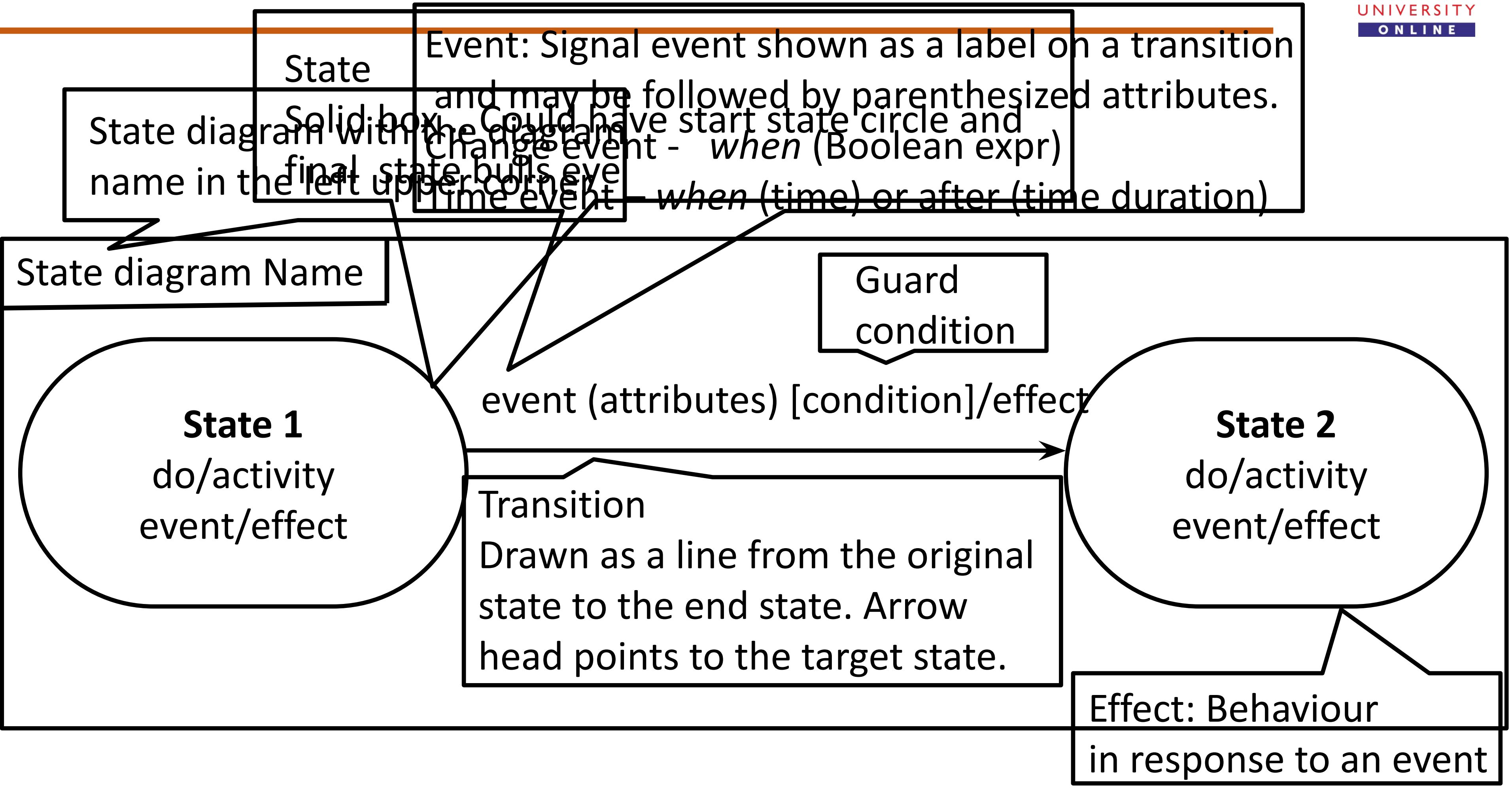
## State Model :

- A state model consists of multiple state diagrams one state diagram for each class with important temporal behavior. The individual state diagrams interact by passing events and through the side effects of guard conditions.

- UML notation for a state diagram is a rectangle with its name in small pentagonal tag in the upper left corner. The constituent states and transitions lie within the rectangle.

- States do not totally define all values of an object.

## State Diagram :



State diagram with the diagram name in the left upper corner

State

Solid box . Could have start state circle and final state bulls eye

Event: Signal event shown as a label on a transition and may be followed by parenthesized attributes.

Change event - *when* (Boolean expr)

Time event - *when* (time) or after (time duration)

Transition

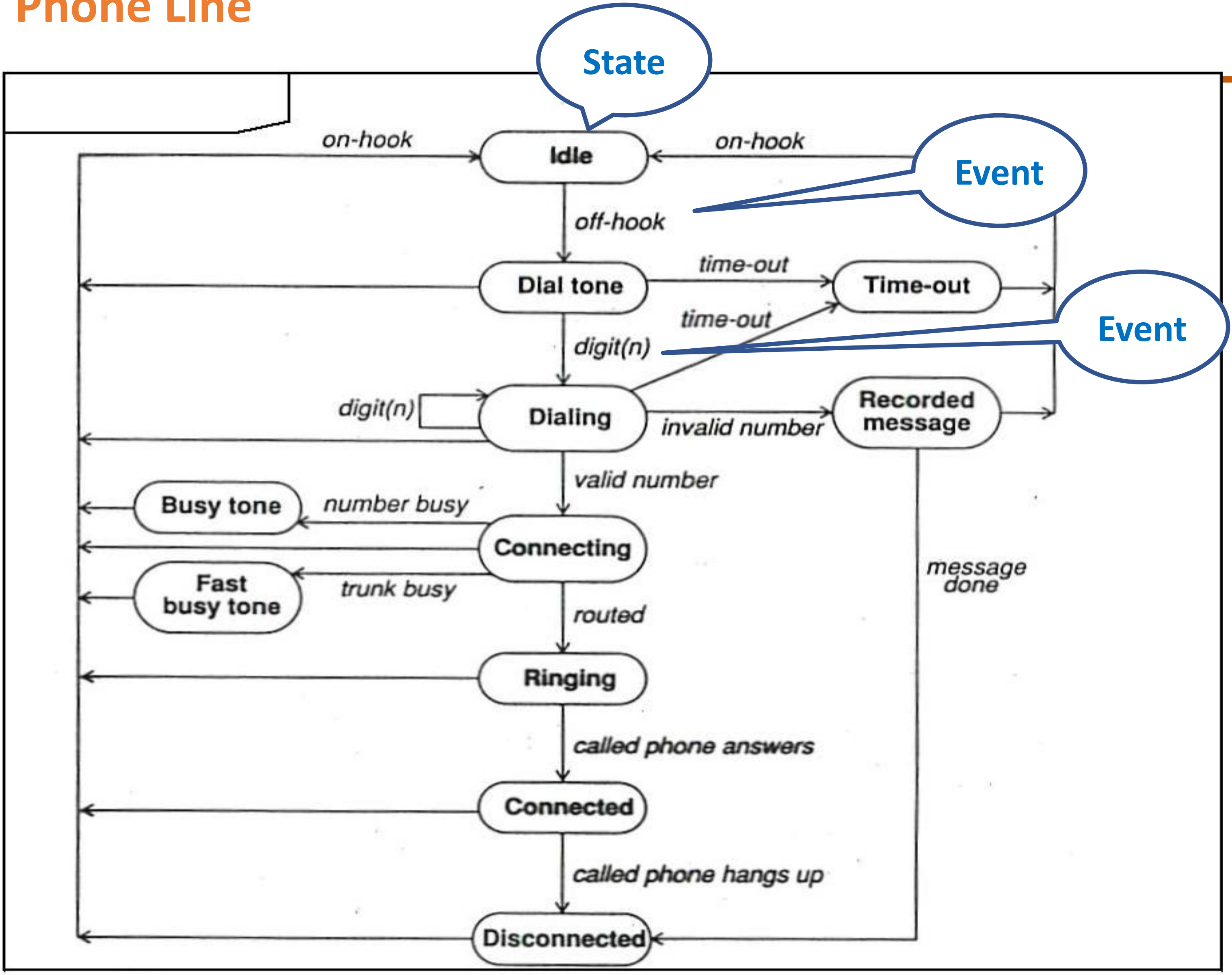
Drawn as a line from the original state to the end state. Arrow head points to the target state.

Effect: Behaviour in response to an event

Event can be entry, exit, etc.

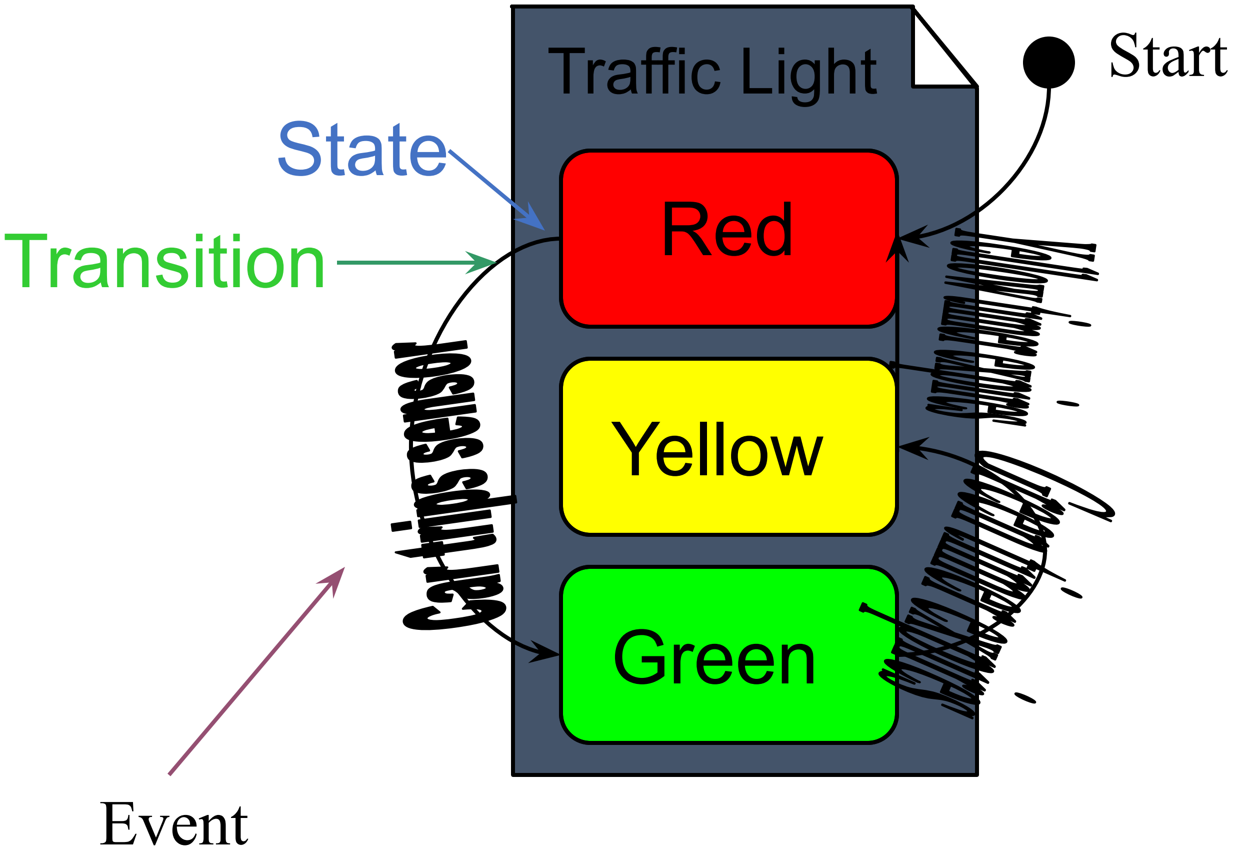
Do activity continues as long as the object is in that state

## Phone Line



State diagram for phone line

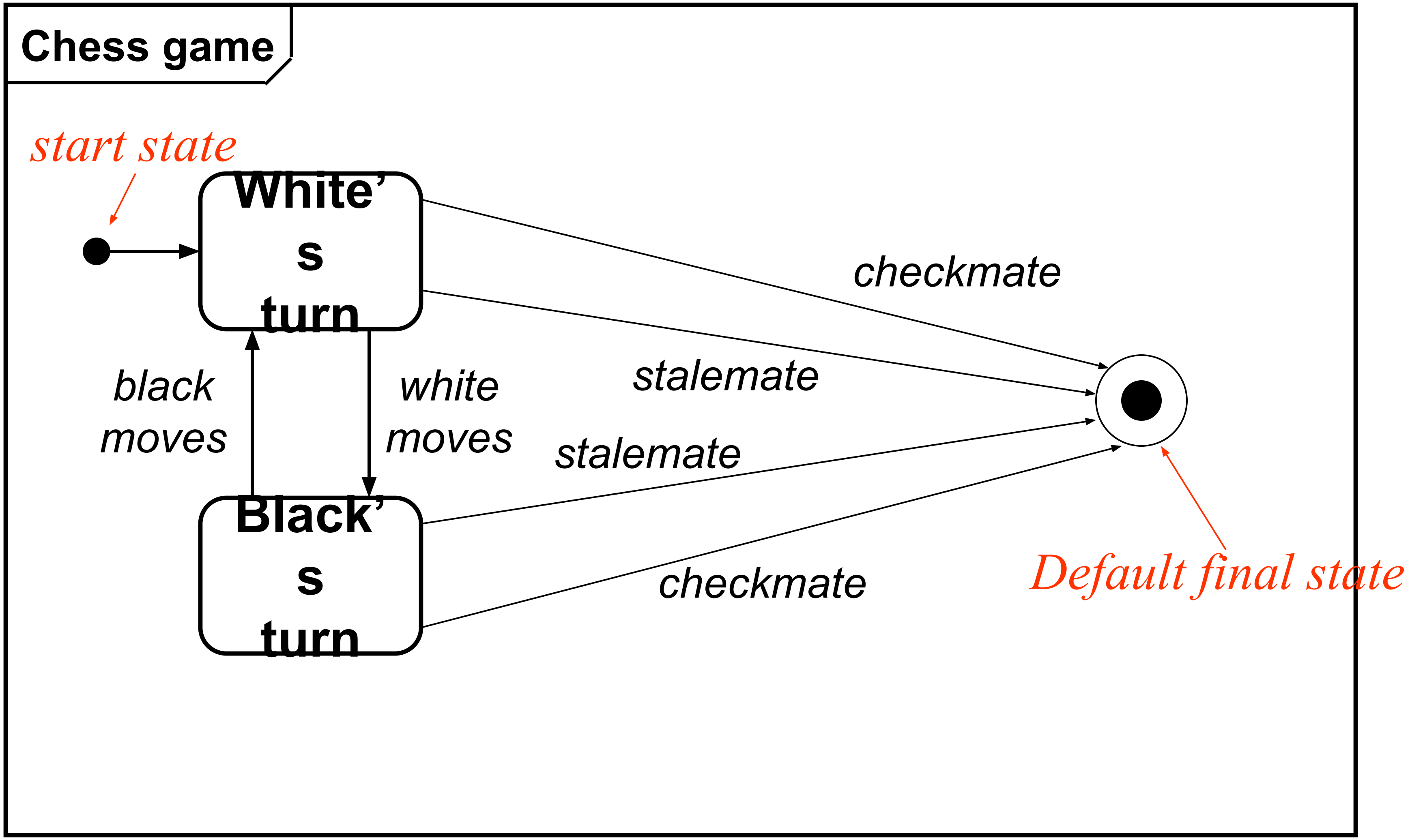
## **State Diagrams** (Traffic light example)



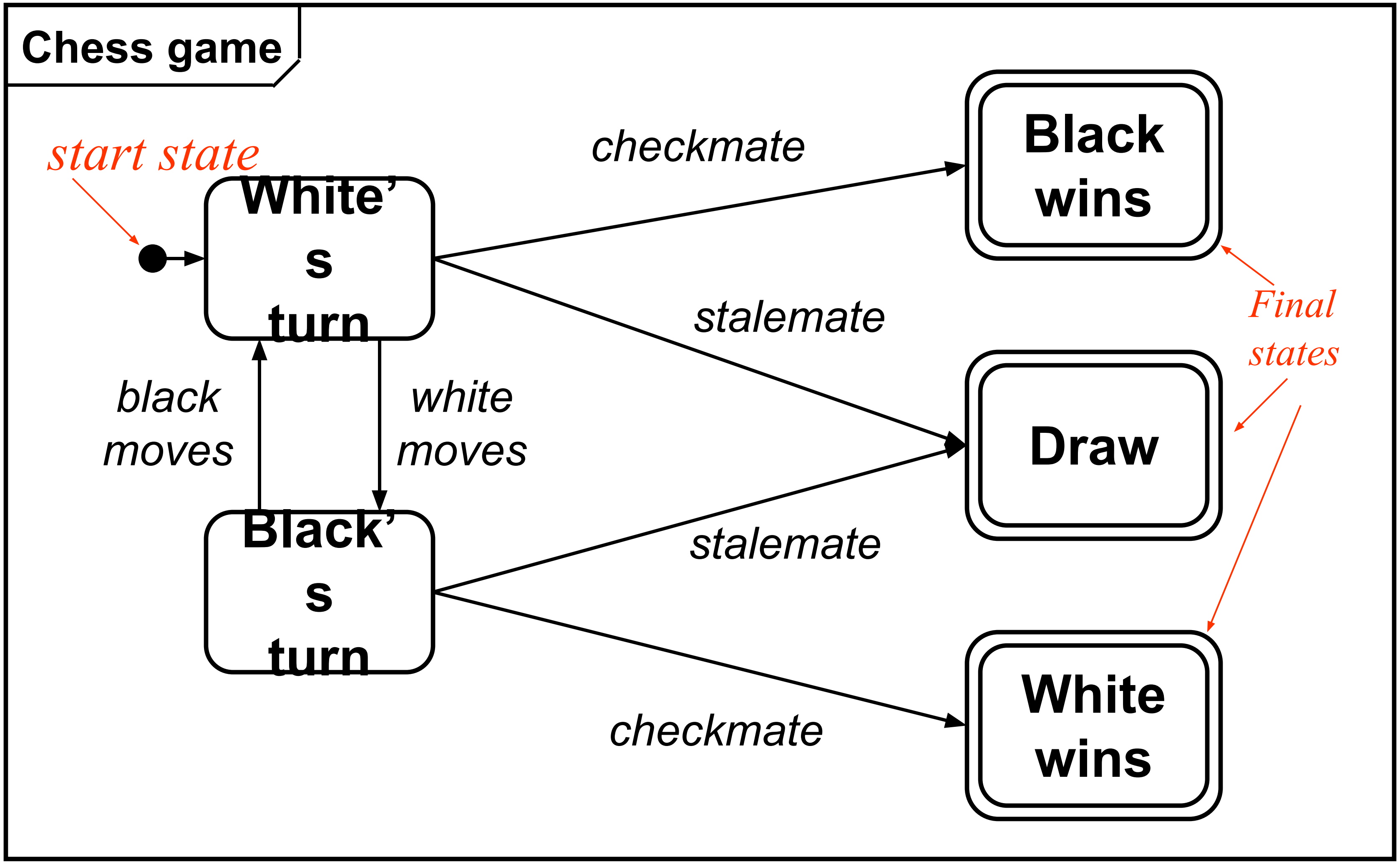
## ONE SHOT STATE DIAGRAMS

- State diagrams can represent continuous loops or one-shot life cycles. Eg: Diagram for the phone line is a continuous loop
- One – shot state diagrams represent objects with finite levels and have initial and final states. The initial state is entered on creation of an object ;entry of the final state implies destruction of the object.
- Object created in a state and change of the state deletes the object
- As an alternate notation, you can indicate initial and final states via **entry** and **exit** points. Entry points (hollow circles) and exit points (circles enclosing an "x") appear on the state diagram's perimeter and may be named.

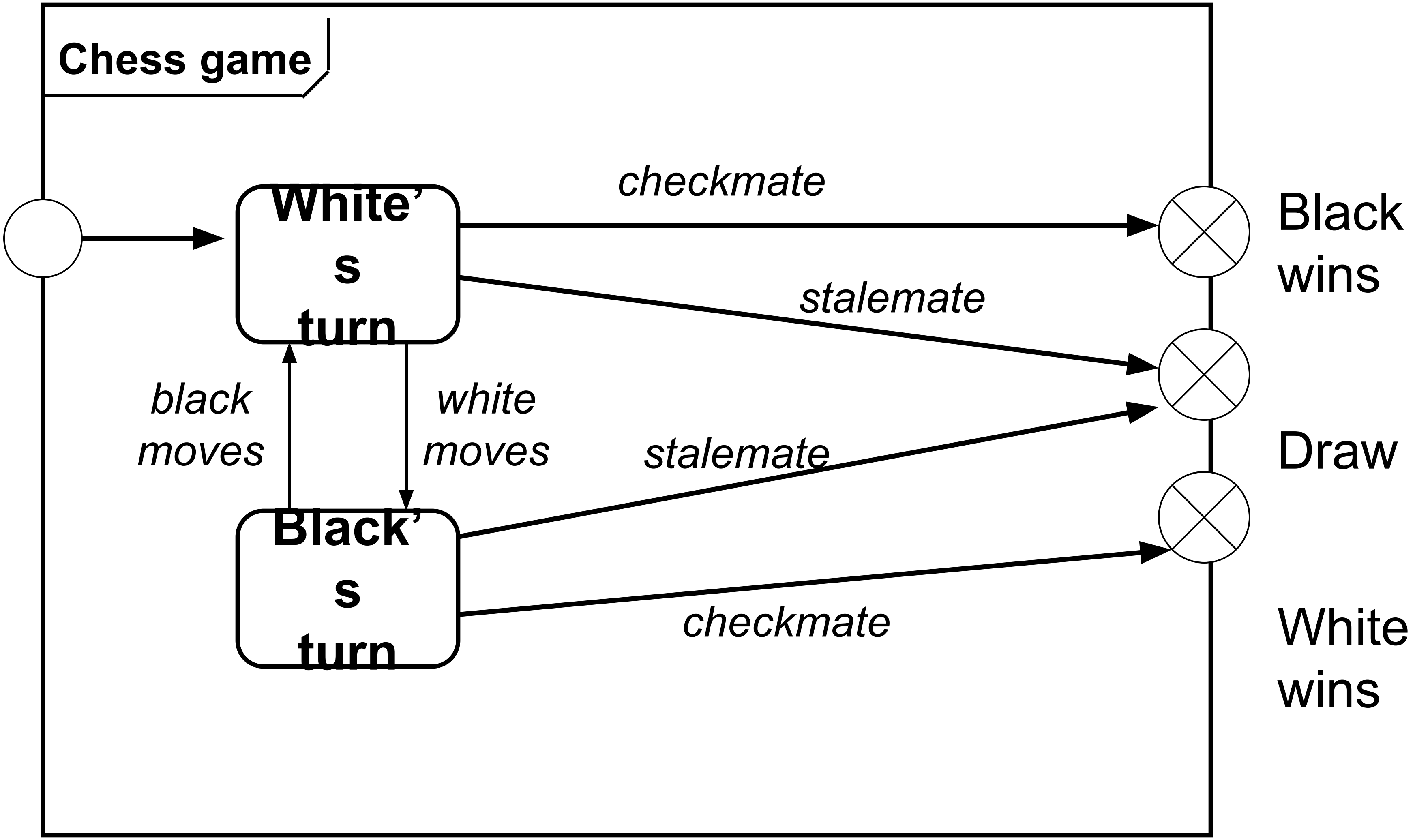
## Example


Chess game
start state
White's turn
checkmate
stalemate
black moves
white moves
stalemate
Black's turn
checkmate
Default final state

## Example


Chess game
start state
White's turn
Black's turn
black moves
white moves
checkmate
stalemate
stalemate
checkmate
Black wins
Draw
White wins
Final states

## Example - entry and exit points



# State Diagram : Activity Effects - representing Behavior of object

- This references to the behavior of the object with state change or the activity (an action of an activity) that is invoked in response to an event. An activity is the actual behavior that can be invoked by any number of effects.

  Eg: disconnectPhoneLine might be an activity that is executed in response to an onHook event.

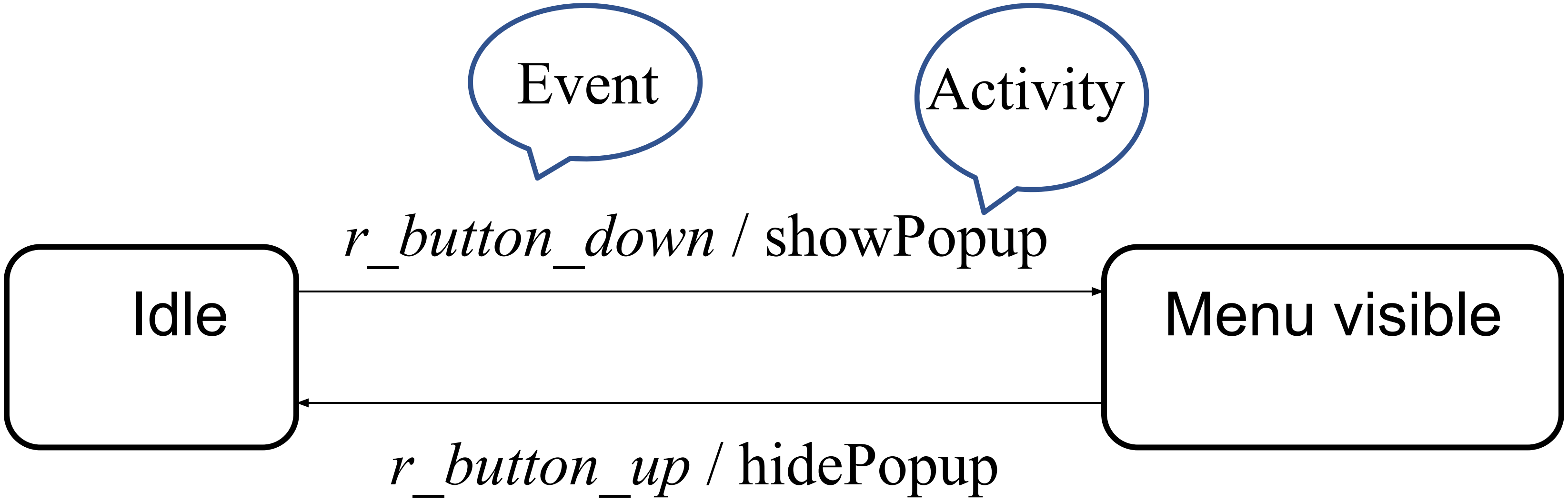
- An activity may be performed upon
  - a transition
  - the entry to or exit from a state, or
  - some other event within a state.

## Activities

Often useful to specify an *activity* that is performed within a given state

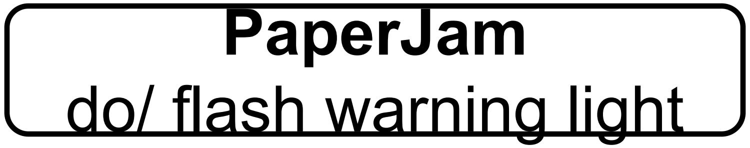
- E.g., while in **PaperJam** state, the warning light should be flashing
- E.g., on entry into the **Opening** state, the motor should be switched on
- E.g., upon exit of the **Opening** state, the motor should be switched off

## Activity effects



## Do-Activities

- continue for an extended time
- can occur only within a state
- can not be attached to a transition
- include
  - continuous operations, such as displaying a picture on a television screen
  - Sequential operations that terminate by themselves after an interval of time
- may be performed for all or part of time that an object is in a state
- may be interrupted by event received during execution; event may or may not cause state transition

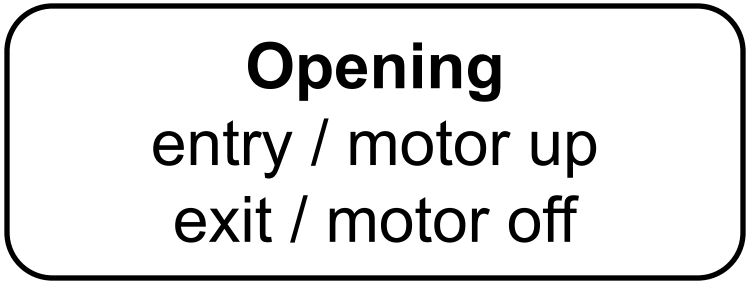


PaperJam

do/ flash warning light

## Entry and Exit Activities

- can bind activities to entry to/ exit from a state
- All transitions into a state perform the same activity, in which case it is more concise to attach the activity to the state



**Opening**
entry / motor up
exit / motor off

# Order of activities

1. activities on incoming transition
2. entry activities
3. do-activities
4. exit activities
5. activities on outgoing transition

Events that cause transitions out of the state can interrupt do-activities. If a do-activity is interrupted, the exit activity is still performed

## EXAMPLE

The control of a garage door opener

- ❑ The user generates depress events with a pushbutton to open and close the door.
- ❑ Each event reverses the direction of the door.
- ❑ The control generates motor up and motor down activities for the motor.

# FIG: THE CONTROL OF A GARAGE DOOR OPENER

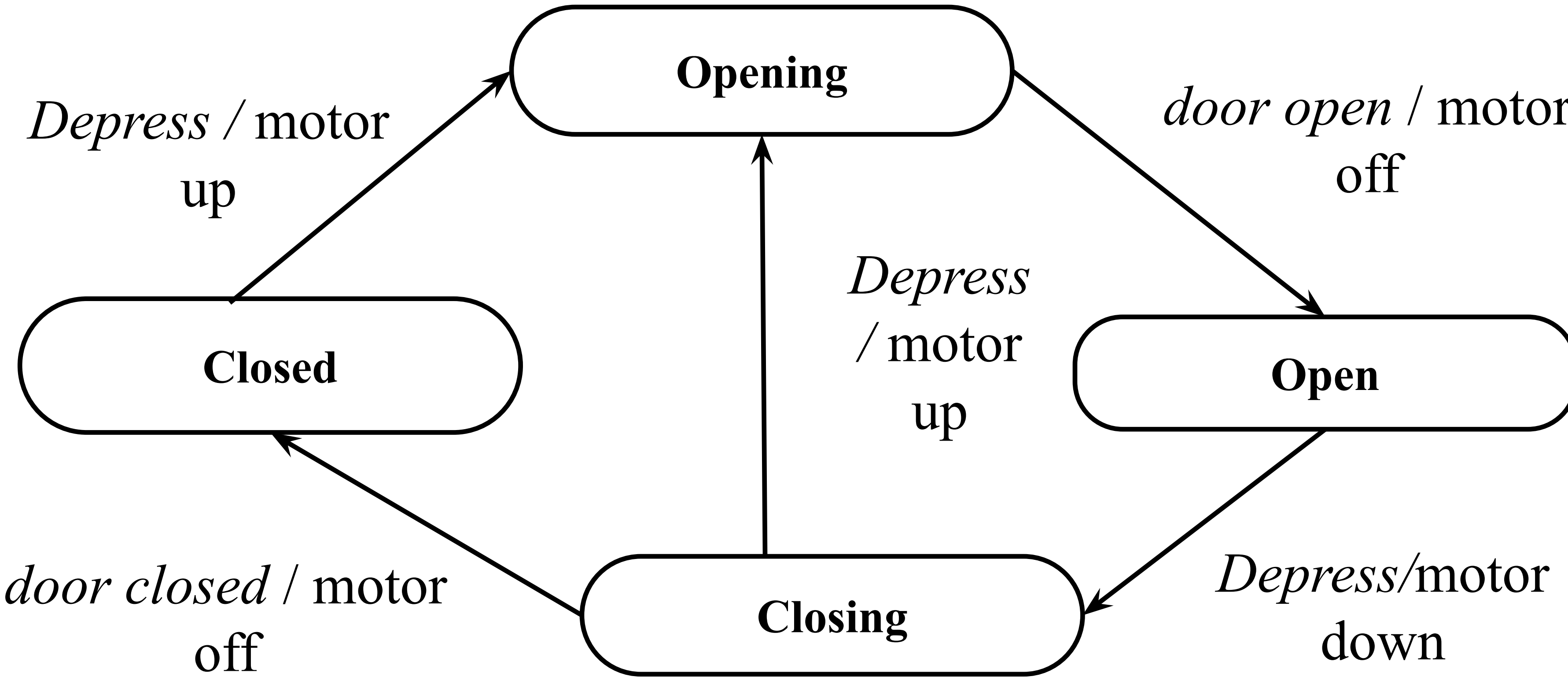


Figure: Activities on transitions. An activity may be bound to an event that causes a transition

# FIG: THE CONTROL OF A GARAGE DOOR OPENER

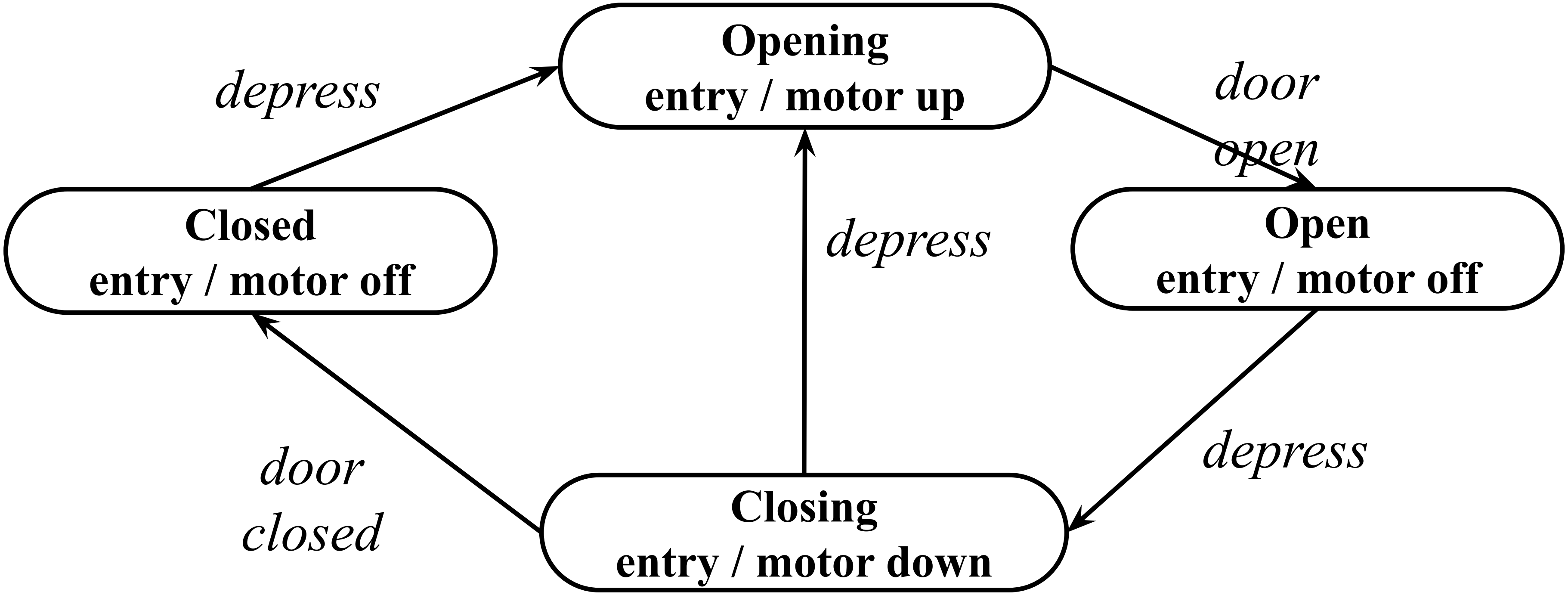


Figure: Activities on transitions. An activity may also be bound to an event that occurs within a state

# References

Text Book: Object Oriented Modeling and Design with UML, by Michael R Blaha and James Rumbaugh.

# THANK YOU

**Mahitha G**
Department of Computer Science and Engineering
**mahithag@pes.edu**