## UNIT-V

## BEHAVIORAL MODELING

# Use Case Diagrams

### Terms and Concepts

A *use case diagram* is a diagram that shows a set of use cases and actors and their relationships.

### Common Properties

A use case diagram is just a special kind of diagram and shares the same common properties as do all other diagrams• a name and graphical contents that are a projection into a model. What distinguishes a use case diagram from all other kinds of diagrams is its particular content.

### Contents

Use case diagrams commonly contain

- Use cases
- Actors
- Dependency, generalization, and association relationships

Use case diagrams may also contain packages, which are used to group elements of your model into larger chunks. Occasionally, you'll want to place instances of use cases in your diagrams, as well, especially when you want to visualize a specific executing system.

### Common Uses

1. To model the context of a system

Modeling the context of a system involves drawing a line around the whole system and asserting which actors lie outside the system and interact with it.Here, you'll apply use case diagrams to specify the actors and the meaning of their roles

2. To model the requirements of a system

Modeling the requirements of a system involves specifying what that system should do (from a point of view of outside the system), independent of how that system should do it. Here, you'll apply use case diagrams to specify the desired behavior of the system. In this manner, a use case diagram lets you view the whole system as a black box; you can see

what's outside the system and you can see how that system reacts to the things outside, but you can't see how that system works on the inside.

## Common Modeling Techniques

### Modeling the Context of a System

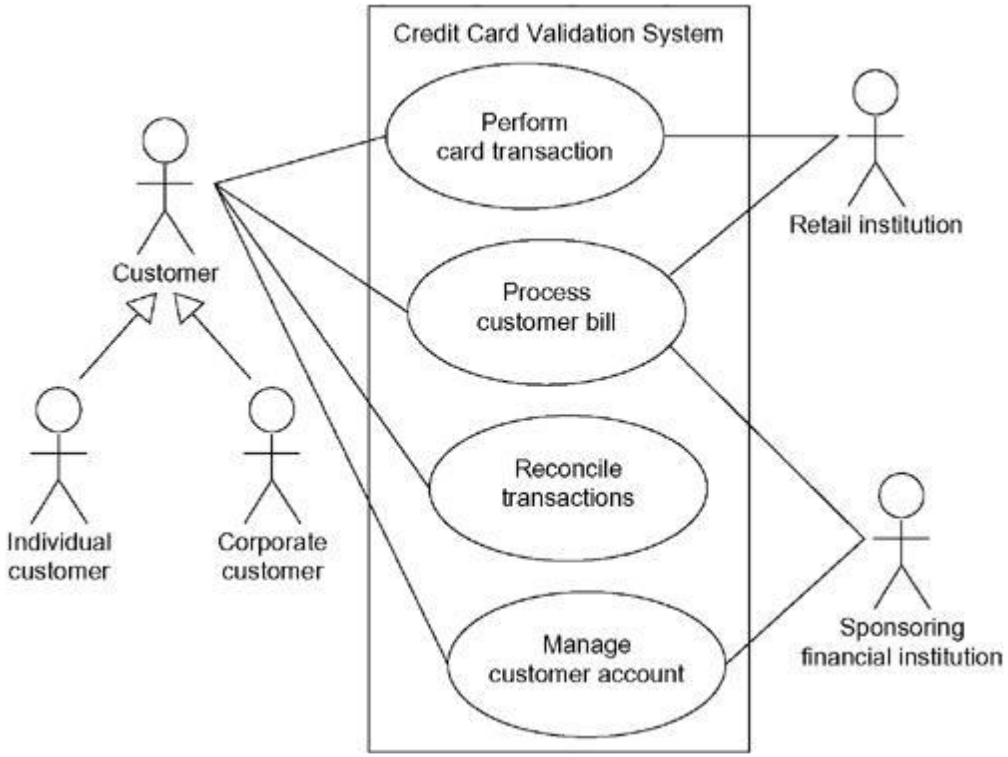
To model the context of a system,

· Identify the actors that surround the system by considering which groups require help from the system to perform their tasks; which groups are needed to execute the system's functions; which groups interact with external hardware or other software systems; and which groups perform secondary functions for administration and maintenance.

· Organize actors that are similar to one another in a generalization/specialization hierarchy.

· Where it aids understandability, provide a stereotype for each such actor.

· Populate a use case diagram with these actors and specify the paths of communication from each actor to the system's use cases.




**Modeling the Requirements of a System**

To model the requirements of a system,

· Establish the context of the system by identifying the actors that surround it.

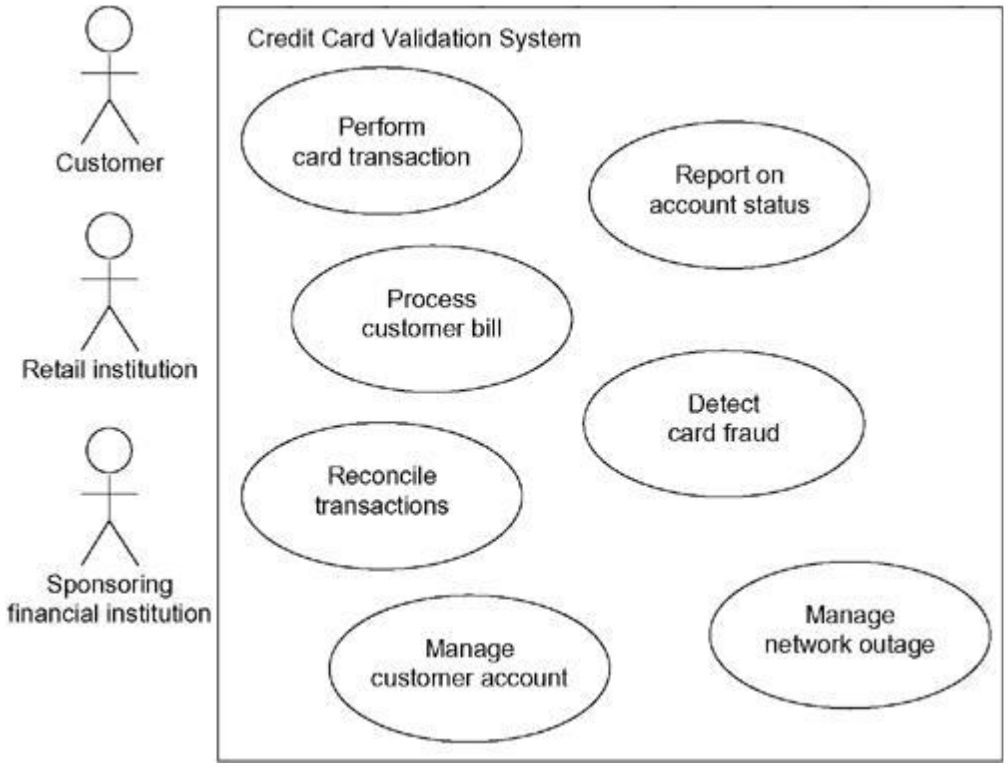
· For each actor, consider the behavior that each expects or requires the system to provide.

· Name these common behaviors as use cases.

· Factor common behavior into new use cases that are used by others; factor variant behavior into new use cases that extend more main line flows.

· Model these use cases, actors, and their relationships in a use case diagram.

· Adorn these use cases with notes that assert nonfunctional requirements; you may have to attach some of these to the whole system.




## Interaction Diagrams

### Terms and Concepts

An *interaction diagram* shows an interaction, consisting of a set of objects and their relationships, including the messages that may be dispatched among them. A *sequence diagram* is an interaction diagram that emphasizes the time ordering of messages. Graphically, a sequence diagram is a table that shows objects arranged along the X axis and messages, ordered in increasing time, along the Y axis. A *collaboration diagram* is an

interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Graphically, a collaboration diagram is a collection of vertices and arcs.

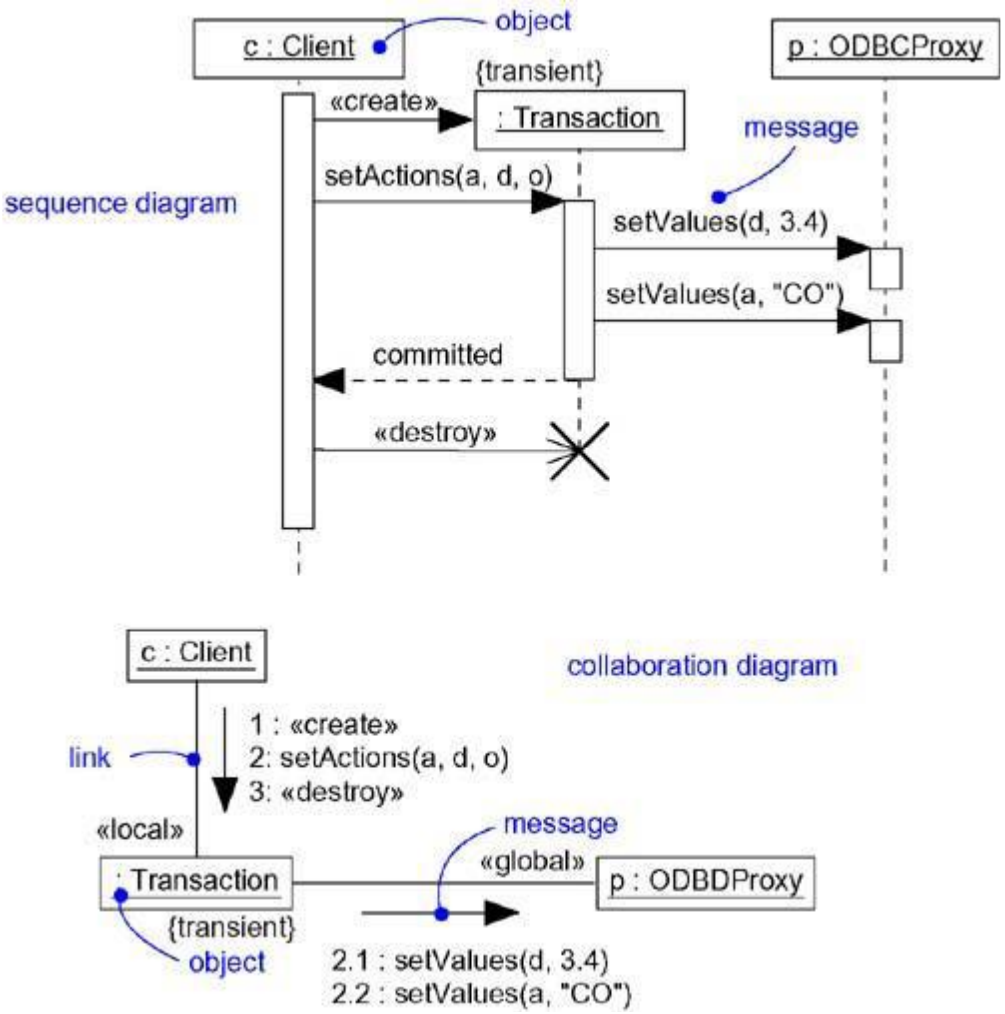
## Common Properties

An interaction diagram is just a special kind of diagram and shares the same common properties as do all other diagrams• a name and graphical contents that are a projection into a model. What distinguishes an interaction diagram from all other kinds of diagrams is its particular content.

## Contents

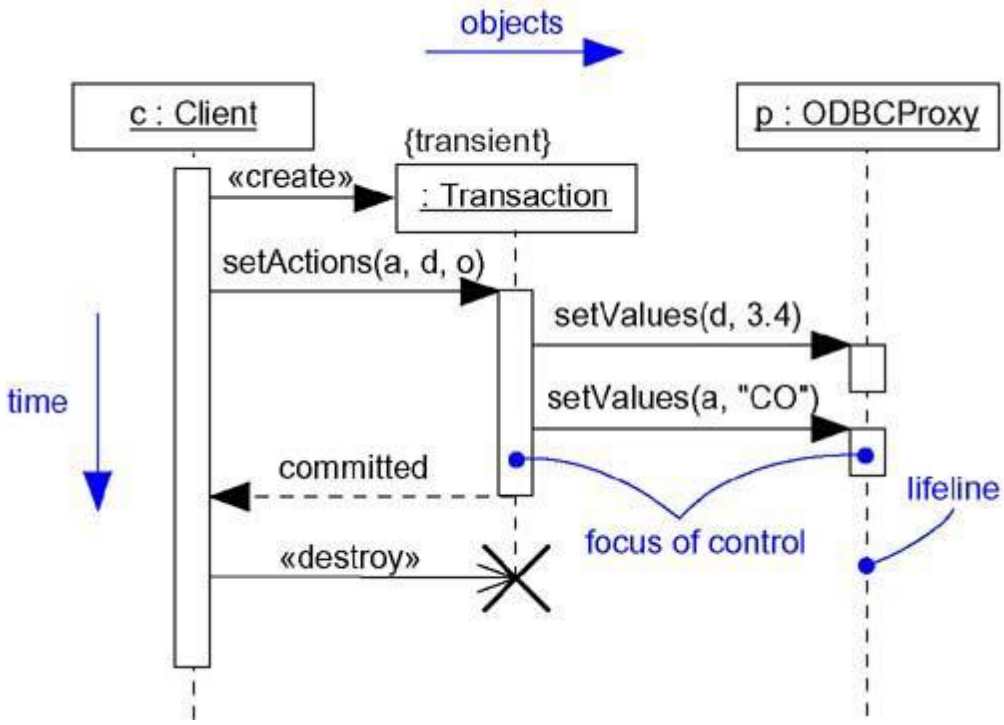
Interaction diagrams commonly contain

- Objects
- Links
- Messages




**Sequence Diagrams**

A sequence diagram emphasizes the time ordering of messages. As Figure 18-2 shows, you form a sequence diagram by first placing the objects that participate in the interaction at the top of your diagram, across the X axis. Typically, you place the object that initiates the interaction at the left, and increasingly more subordinate objects to the right. Next, you place the messages that these objects send and receive along the Y axis, in order of increasing time from top to bottom. This gives the reader a clear visual cue to the flow of control over time.




## Activity Diagrams

### Terms and Concepts

An *activity diagram* shows the flow from activity to activity. An is an ongoing nonatomic execution within a state machine. Activities ultimately result in some *action*, which is made up of executable atomic computations that result in a change in state of the system or the return of a value. Actions encompass calling another operation, sending a signal, creating or destroying an object, or some pure computation, such as evaluating an expression. Graphically, an activity diagram is a collection of vertices and arcs.

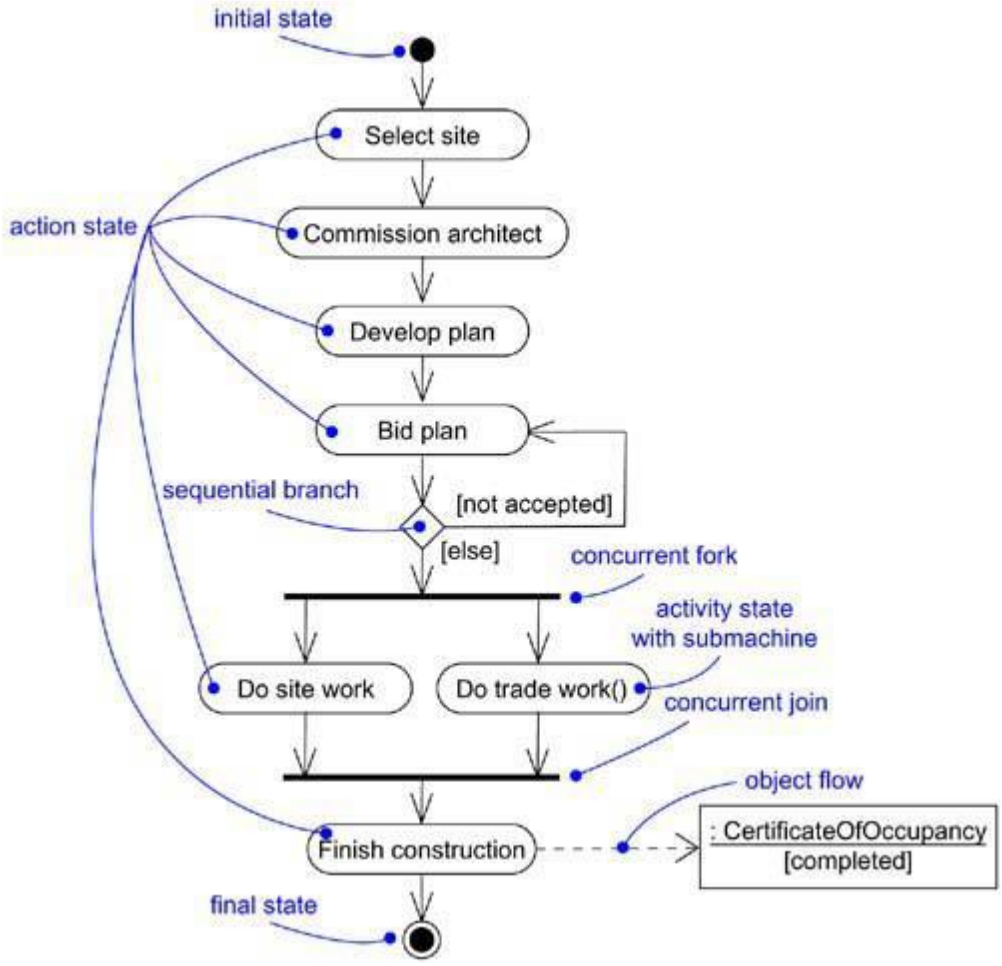
### Common Properties

An activity diagram is just a special kind of diagram and shares the same common properties as do all other diagrams• a name and graphical contents that are a projection into a model. What distinguishes an interaction diagram from all other kinds of diagrams is its content.

### Contents

Activity diagrams commonly contain

- Activity states and action states
- Transitions
- Objects

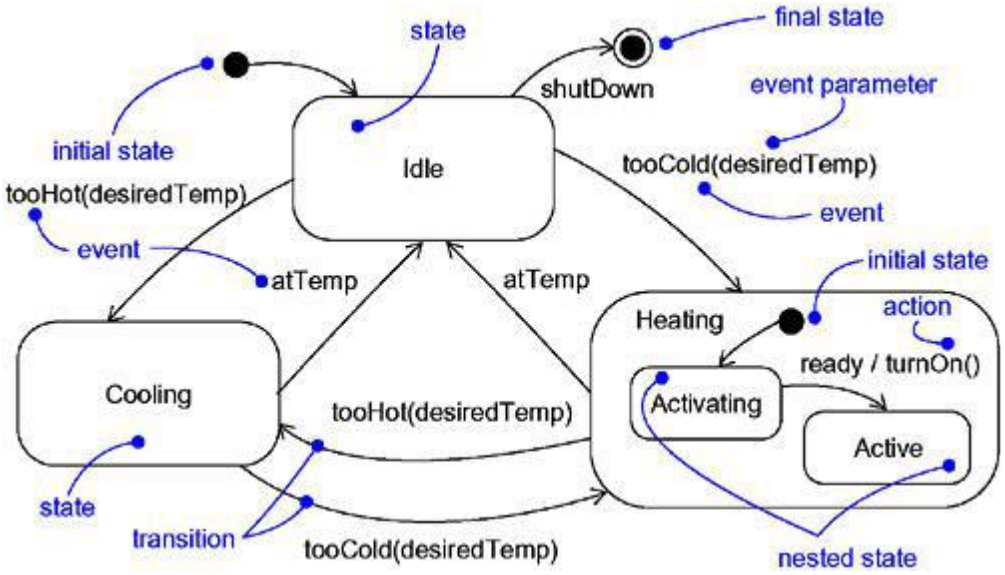



## State Machines

### Terms and Concepts

A *state machine* is a behavior that specifies the sequences of states an object goes through during its lifetime in response to events, together with its responses to those events. A *state* is a condition or situation during the life of an object during which it satisfies some condition, performs some activity, or waits for some event. An *event* is the specification

of a significant occurrence that has a location in time and space. In the context of state machines, an event is an occurrence of a stimulus that can trigger a state transition. A *transition* is a relationship between two states indicating that an object in the first state will perform certain actions and enter the second state when a specified event occurs and specified conditions are satisfied. An *activity* is ongoing nonatomic execution within a state machine. An *action* is an executable atomic computation that results in a change in state of the model or the return of a value. Graphically, a state is rendered as a rectangle with rounded corners. A transition is rendered as a solid directed line.

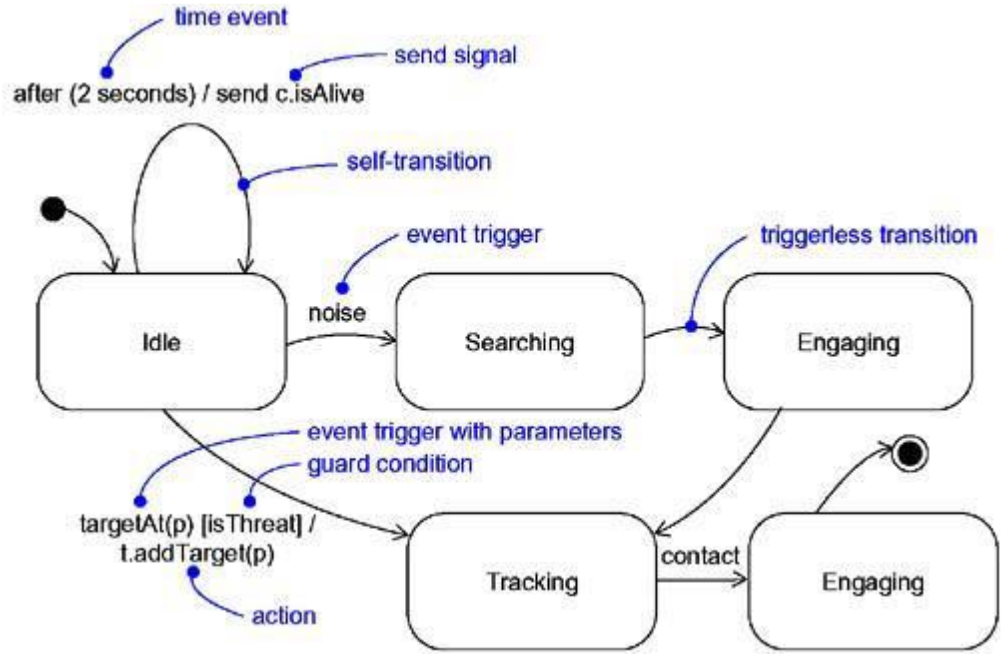



**States**

A state is a condition or situation during the life of an object during which it satisfies some condition, performs some activity, or waits for some event. An object remains in a state for a finite amount of time. For example, a Heater in a home might be in any of four states: Idle (waiting for a command to start heating the house), Activating (its gas is on, but it's waiting to come up to temperature), Active (its gas and blower are both on), and ShuttingDown (its gas is off but its blower is on, flushing residual heat from the system).

## Transitions

A transition is a relationship between two states indicating that an object in the first state will perform certain actions and enter the second state when a specified event occurs and specified conditions are satisfied. On such a change of state, the transition is said to fire. Until the transition fires, the object is said to be in the source state; after it fires, it is said to be in the target state. For example, a Heater might transition from the Idle to the Activating state when an event such as tooCold (with the parameter desiredTemp) occurs.




### Event Trigger

An event is the specification of a significant occurrence that has a location in time and space. In the context of state machines, an event is an occurrence of a stimulus that can trigger a state transition. As shown in the previous figure, events may include signals, calls, the passing of time, or a change in state. A signal or a call may have parameters whose values are available to the transition, including expressions for the guard condition and action.