STATE CHART DIAGRAMS

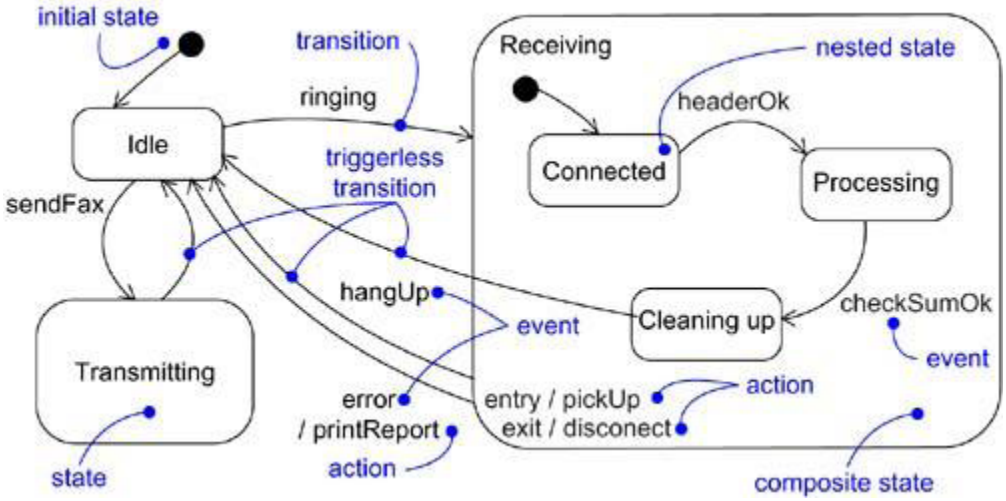

Figure 1: Statechart Diagram

Statechart diagram is simply a presentation of a state machine which shows the flow of control from state to state.

Statechart diagrams are important for constructing executable systems through forward and reverse engineering.

statechart diagrams are useful in modeling the lifetime of an object

Statechart diagrams commonly contain – Simple states and composite states, Transitions- including events and actions

It is one of the five diagrams in UML for modeling the dynamic aspects of systems.

Graphically, a statechart diagram is a collection of vertices and arcs.

A state is a condition or situation in the life of an object during which it satisfies some condition, performs some activity, or waits for some event.
An event in the context of state machines 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.

A reactive or event-driven object is one whose behavior is best characterized by its response to events dispatched from outside its context

Modeling Reactive Objects

To model a reactive object,

· Choose the context for the state machine, whether it is a class, a use case, or the system as a whole.

· Choose the initial and final states for the object. To guide the rest of your model, possibly state the pre- and postconditions of the initial and final states, respectively.

· Decide on the stable states of the object by considering the conditions in which the object may exist for some identifiable period of time. Start with the high-level states of the object and only then consider its possible substates.

· Decide on the meaningful partial ordering of stable states over the lifetime of the object.

· Decide on the events that may trigger a transition from state to state. Model these events as triggers to transitions that move from one legal ordering of states to another.

· Attach actions to these transitions (as in a Mealy machine) and/or to these states (as in a Moore machine).

· Consider ways to simplify your machine by using substates, branches, forks, joins, and history states.

· Check that all states are reachable under some combination of events.

· Check that no state is a dead end from which no combination of events will transition the object out of that state.

· Trace through the state machine, either manually or by using tools, to check it against expected sequences of events and their responses.

For example, Figure 2 shows the statechart diagram for parsing a simple context-free language

message : ‘<‘string ‘>’string ‘;’

The first string represents a tag; the second string represents the body of the message.

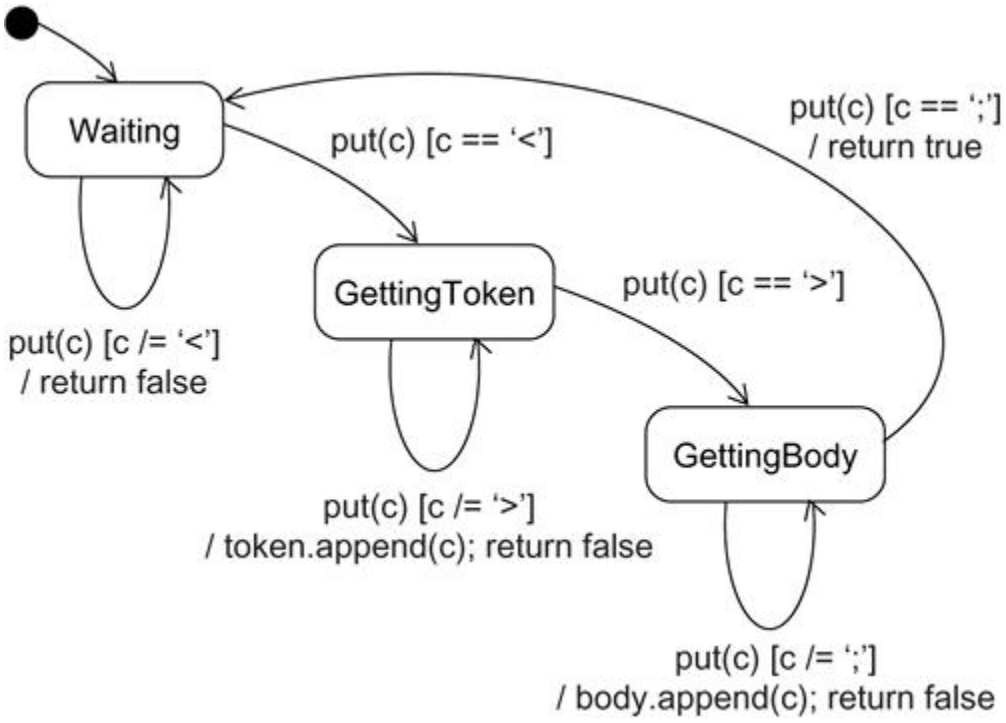

Figure 2: Modeling Reactive Objects

Source : http://praveenthomasln.wordpress.com/2012/04/07/statechart-diagrams-s8-cs/