Finite State Machines and Statecharts

Hassan Gomaa
Dept of Information & Software Engineering
George Mason University

Reference: H. Gomaa, "Chapter 10 - Designing Concurrent, Distributed, and Real-Time Applications with UML", Addison Wesley Object Technology Series, July, 2000

Object-Oriented Software Life Cycle

Requirements & Analysis Modeling

- Requirements Modeling
  - Use Case Modeling
    - – Define software functional requirements in terms of use cases and actors
- Analysis Modeling
  - Static Modeling
    - – Define structural relationships between classes
    - – Depict classes and their relationships on class diagrams
  - Dynamic Modeling
    - – Define statecharts for state dependent objects
    - – Defines how objects participate in use cases using collaboration diagrams or sequence diagrams

Finite State Machines

Finite number of states

Only in one state at a time

Transition

Change of state

Caused by event

Transition to same or different state

Action may result from state transition

Notation

State transition diagram
State transition table
Statechart
Examples of statecharts (Figures 10.1 - 10.3)

Finite State Machines and Statecharts

- Statechart
  - Graphical representation of finite state machine
  - States are rounded boxes
  - Transitions are arcs
- Statechart relates events and states
- Event
  - Causes change of state
    - Referred to as state transition
- State
  - A recognizable situation
  - Exists over an interval of time
  - Represents an interval between successive events
- Examples of statecharts (Figures 10.1 - 10.3)

Events

- Event
  - A discrete signal that happens at a point in time
  - Also known as a stimulus
  - Has no duration
- Two events
  - May logically depend on each other
  - E.g, ATM Card inserted before Pin # entered
- Two events
  - May be independent of each other
  - E.g., ATM card read at Alexandria ATM
  - ATM Card read at Fairfax ATM

Events and Conditions

- State transition label
  - Event [Condition]
- Condition is a Boolean function
  - Conditions are optional on statecharts
  - Condition is true for finite period of time
- When event occurs, condition must be *true* for state transition to occur.
- If condition is *false*, state transition does not occur
- Condition may be used to indicate that event has occurred
  - E.g., Closedown Was Requested
- **Figure 10.4 Partial statechart**
- **Figure 10.5 Relationship between events and conditions**
- **Figure 10.6 Use of events and conditions in statechart**
- **Figure 10.7 Example of events and conditions**

Actions

- State transition label
  - Event / action(s)
  - Event [condition] / action(s)
- Action
  - Executed as a result of state transition
  - Executes instantaneously at state transition
  - Terminates itself
  - Is optional
- **Figure 10.8 Example of actions**
- **Figure 10.9 Detailed Cruise Control statechart with actions and conditions**
- Activity
- Entry/exit actions

Activities

- Activity
  - Executes for duration of state
    - Enable Activity on entry to state
    - Disable Activity on exit from state
  - Alternatively
    - do / Activity in state
- Examples of activities
  - Increase Speed
    - Executes for duration of Accelerating state
  - Maintain Speed
    - Executes for duration of Cruising state
  - Resume Cruising
    - Executes for duration of Resuming state
- **Figure 10.10 Cruise Control statechart with activities**

Entry and Exit Actions

- Entry action
  - Action executed on entry into state
    - Entry / action
  - E.g., Display System Down
  - E.g., Display Welcome
  - **Figure 10.11 Example of entry actions**
- Exit action
  - Action executed on exit from state
    - Exit / action
  - E.g, Select Desired Speed
  - **Figure 10.12 Example of exit action**

Hierarchical Statecharts

- Disadvantages of State Transition Diagrams and Flat Statecharts
  - Complex State Transition Diagrams get very cluttered
  - Limited capability for managing complexity
- Hierarchical Statecharts
  - Based on Harel Statecharts
  - Notation for hierarchical decomposition of state transition diagrams
    - Superstate decomposed into substates
    - Default entry states
    - Transition out of superstate corresponds to transition out of every substate

Hierarchical Statecharts

- **OR** decomposition
  - When object is in superstate
    - It is in one and only one of substates
  - Transition into superstate
    - Must be to one and only one of substates
- Aggregation of state transitions
  - If same event causes transition out of every substate
  - Then aggregate into transition out of superstate
- Examples: Fig. 10.14, 19.20-19.23

Hierarchical Statecharts

- Concurrent statecharts
  - State of an object described by more than one statechart
  - Show different aspects of object, may not be concurrent
- Orthogonal statechart
  - Used to depict states of different aspects of object
- **AND** decomposition
  - Object is in one substate on each lower level statechart
  - Object's state is union of all substates
- Same event
  - May cause transitions on more than one statechart
- Output event on one statechart
  - May be input event on other statechart
- Substate on one statechart
  - May be condition on other statechart
- Example: Fig. 10.15

Guidelines on Statecharts

- State name must be passive not active
  - Represents time period when something
    - is happenING, e.g., Elevator Moving
    - Identifiable situation, e.g., Elevator Idle, Initial
- State names must be unique
- Must be able to exit from every state
- Flat statechart
  - Statechart is only in one state at a time
- Hierarchical statechart
  - **or** decomposition
    - Statechart is only in one substate at a time
  - **and** decomposition
    - Statechart is in one substate on each lower level concurrent statechart

Guidelines on Statecharts

- Event is the cause of the state transition
  - Event happens at a moment in time
  - Event name indicates something has just happened
    - e.g, Up Request, Door Closed
- Action is the result of the state transition
  - Action is a command, e.g., Stop, Close Door
  - Action executes instantaneously
  - Activity executes throughout a given state
- More than one action possible with a state transition
  - No sequential dependency between actions
- Condition is a Boolean value
  - Event [Condition]
  - State transition only occurs if
    - Event happens & Condition is True
  - Condition is True over some interval of time
- Actions, Activities and Conditions are optional

Developing Statechart from Use Case

- Develop state dependent use case
- Start with scenario (one path through use case)
  - Consider sequence of interactions between actor and system
- Consider sequence of external events
  - Input event from external environment
  - Causes state transition to new state
  - Action may result from state transition
  - Activity may be enabled / disabled
- Initially develop flat statechart

Example of Developing Statechart from Use Case

- Cruise Control System
- Control Speed use case
  - Scenario of external events
  - Initial state: INITIAL
    a) Driver engages cruise control lever in ACCEL position
    b) Driver releases lever (CRUISE)
    c) Driver presses brake
    d) Driver engages lever in RESUME position
- Example: Fig. 10.16, 10.17

Example of Developing Statechart from Use Case

Control Speed Use Case
Actor: Driver

Summary: This use case describes the automated cruise control of the car, given the driver inputs via the cruise control lever, brake, and engine external input devices.

Precondition: Driver has switched on the engine and is operating the car manually.

Description:

This use case is described in terms of a typical scenario consisting of the following sequence of external events:

1. Driver moves the cruise control lever to the ACCEL position and holds the lever in this position. The system initiates automated acceleration so that the car automatically accelerates.
2. Driver releases the cruise control lever in order to cruise at a constant speed. The system stops automatic acceleration and starts maintaining the speed of the car at the cruising speed. The cruising speed is stored for future reference.
3. Driver presses the brake to disable cruise control. The system disables cruise control so that the car is once more under manual operation.
4. Driver moves the cruise control lever to the RESUME position in order to resume cruising. The system initiates acceleration (or deceleration) toward the previously stored cruising speed.
5. When the system detects that the cruising speed has been reached, it stops automatic acceleration (or deceleration) and starts maintaining the speed of the car at the cruising speed.
6. Driver moves the cruise control lever to the OFF position. The system disables cruise control, so that the car is once more under manual operation.
7. The driver stops the car and switches off the engine.

Developing Statechart from Use Case
(continued)

- Consider alternative external events
  - Could result in additional states
  - Could result in additional state transitions
  - Example: Fig. 10.10
- Develop hierarchical statechart
  - States that can be aggregated to form superstate
  - Event causing transition from several states
    - Create superstate with one transition out of superstate
    - Instead of many transitions out of substates
  - Example: Fig. 10.18-10.19
- Develop orthogonal statechart
  - Model different aspects of state dependent object
  - Example: Fig. 10.20

Example of Developing Statechart from Use Case

(Control Speed use case - continued)

Alternatives:

The driver actor interacts with the system, using three external input devices: the cruise control lever, the brake, and the engine. Following are the complete set of input events initiated by the driver actor using these external devices, and the reaction of the system to them:

- The Accel, Cruise, Resume, and Off external events from the cruise control lever. The Accel event causes automated acceleration, providing the brake is not pressed. The Cruise event may only follow an Accel event. The Resume event may only occur after cruising has been disabled and the desired cruising speed has been stored. The Off event always disables cruise control.
- The Brake Pressed and Brake Released external events from the brake. The Brake Pressed event always disables cruise control. Automated vehicle control is not possible as long as the brake is pressed. After the brake is released, automated vehicle control may be enabled.
- The Engine On and Engine Off external events from the engine. The Engine Off event disables any activity in the system.

Postcondition: The car is stationary, with the engine switched off.