Object Behaviours
UML Statecharts

Software Engineering Design
Lecture 5

Based on Ch 5 Lecture Notes by Bruegge & Dutoit

Familiar notation?

Ex 1. Interaction with my PC (1)

Interaction with my PC (2)

UML Statechart Diagram Notation

Statechart Diagrams
- Graphs whose nodes are **states** and whose directed arcs are transitions labelled by **events**.
- States capture conditions which hold for a period of time
  - e.g. light is on, light is off
- Events change the state
  - e.g. turning the light on, turning the light off
- A statechart diagram relates events and states for one class
  - An object model with a set of objects has a set of state diagrams
State
- An abstraction of the attribute of a class
  - State is the aggregation of several attributes of a class
- Basically an equivalence class of all those attribute values and links that do not need to be distinguished as far as the control structure of the system is concerned
  - Example: State of a user interface screen
    - idle, login, active, logout
    - NB active is an abstraction of all the user’s logged in activity
- State has duration

Event
- An activity which changes the state
  - e.g. the operations of an object

Example 2: vending machine
- coins_in(amount) / set balance
- Collect Money
  - coins_in(amount) / add to balance
- cancel / refund coins
- [item empty] / [select(item)] / [change<0]
- do: test item and compute change
  - [change=0] / [change>0]
- do: dispense item
- do: make change

Another example
- Incident states of the Incident object of FRIEND

Problem Statement:
Direction Control for a Toy Car
- Power is turned on
  - Car moves forward and car headlight shines
- Power is turned off
  - Car stops and headlight goes out.
- Power is turned on
  - Headlight shines
- Power is turned off
  - Headlight goes out.
- Power is turned on
  - Car runs backward with its headlight shining.

Toy Car: Dynamic Model
- Headlight
  - Off
  - On
- Wheel
  - Forward
  - Stationary
- Stationary
Practical Tips for Dynamic Modeling

- Construct dynamic models only for classes with significant dynamic behavior
  - Avoid "analysis paralysis"
- Consider only relevant attributes
  - Use abstraction if necessary
- Look at the granularity of the application when deciding on actions and activities
- Reduce notational clutter

Exercise

- Consider the operation of the CITS2220 discussion page:

Model the dynamic behavior of a controller object for someone reading and posting to the forum.