Advanced state modeling

Marlon Dumas
Institute of Computer Science
How does a washing machine work?

- On/off (power) button.
- Start button (no stop button!)
- Light indicates current stage
  - soaking, rinsing, draining, drying
- Three washing plans that can be changes using a “mode” button:
  - Regular
  - Delicate (no soaking)
  - Super delicate (no soaking, no drying)
- Off can be pushed only:
  - before starting
  - or after finishing
Statechart for the washing machine

- **off**:
  - power

- **idle**:
  - mode[plan=delicate]/plan = super delicate
  - mode[plan=regular]/plan = delicate

- **soak**:
  - start[plan=delicate or plan=super delicate]
  - do: light(soak)
  - do: pump(in)
  - after (30)

- **rinse**:
  - after (30 min)
  - do: light(rinse)
  - do: stir()

- **drain**:
  - after (5 min)
  - [plan<>super delicate]/light(off)
  - do: light(drain)
  - do: pump(out)

- **dry**:
  - after (20 min)
  - [plan<>super delicate]/light(off)
  - do: light(dry)
  - do: stir()
State explosion and history

What if the “power” button can be clicked at anytime?

What if we want to come back to the same state we left?
State explosion

- If we have “n” classes with “m” (boolean) attributes each
  (let’s assume that all classes have the same number of attributes)
- Possible states of the whole system = $2^{nm}$
Abstraction in Statecharts

- Factor out common behavior
- Remember history
- Segregate independent behavior

- Composite States
- History pseudo-states
- Orthogonal/Parallel States
Composite states

The transition can be fired from any internal state

start[plan=regular] after (30 min)

start[plan<>super delicate] after (5 min)
start[plan=super delicate] after (20 min) light(off)

start[plan=delicate or plan=super delicate] after (30)

soak
do: light(soak)
do: pump(in)

soak
after (5 min)[plan=super delicate]/light(off)

soak
after (30)

rinsedo: light(rinse)
do: stir()

drain
do: light(drain)
do: pump(out)

drain
after (5 min)[plan<>super delicate]

dry
do: light(dry)
do: stir()
Exercise 1

- Group “FlashOn” and “FlashOff” states into a composite state “Flashing”
Exercise 2

- Fix and simplify this statechart
Exercise 3

- Simplify this statechart
History pseudo-state

- Return to a previously visited hierarchical state
- Shallow history: just the current level
- Deep history: includes all nested states
- Sometimes it is useful to clear history:
  - clear-history(state)  clh(state)
  - clear-history(state*)  clh(state*)
Back to the washing machine...

State modeling -- Marlon Dumas
Washing machine with “history”

```
off

power

on

idle
  do: selectMode()

start[plan=delicate or plan=super delicate]

soak
  do: light(soak)
  do: pump(in)

after (30)

start[plan=regular]

rinse
  do: light(rinse)
  do: stir()

after (30 min)

[plan<>super delicate]/light(off)

after (20 min)/light(off)

dry
  do: light(dry)
  do: stir()

after (5 min)[plan<>super delicate]

drain
  do: light(drain)
  do: pump(out)

dry
  do: light(dry)
  do: stir()

H

start[plan=delicate or plan=super delicate]/light(off)
```
Shallow vs. Deep history
Revise the follow statechart such that:

- There is no flash button. The only way to make the lamp flash is to push the “on” button when we are in the LampOn state
- To stop flashing, we must push on the “on” button or the “off” button
- If we push “off” while the lamp is flashing, and then we push “on”, the lamp re-starts flashing
Note on transition precedence

- Two or more transitions may have the same event trigger
  - inner transition takes precedence
  - if no transition is triggered, event is discarded

![Diagram showing transition precedence]
Order of activities in nested models

- Same approach as for the simple case

Execution sequence:

exS11 \rightarrow exS1 \rightarrow actE \rightarrow enS2 \rightarrow initS2 \rightarrow enS21
Independent behavior

- Multiple simultaneous perspectives on the same entity
Parallelism:
States with orthogonal regions

- Combine multiple simultaneous descriptions
Parallelism:
States with orthogonal regions

- All mutually orthogonal regions detect the same events and respond to them “simultaneously”
- usually reduces to interleaving of some kind
If an object O is composed of multiple other objects O1, O2, etc., we can use orthogonal regions to model O’s behavior as a composition of the state machines of O1, O2, etc.
“Flat” vs. Parallel State Machines

- Every parallel machine can be transformed into a sequential machine:

With Orthogonal Regions

Without Orthogonal Regions
Exercise 5:
Rewrite this without parallel regions
Orthogonal regions/states can be synchronized via conditions of the form “region in a state”

**Note:** In Yakindu the syntax for checking if region A.C is in state M is as follows: [active(A.C.M)]
Readings & Resources

- Last week: Blaha & Rumbaugh, Chapter 5
- **This week:** Blaha & Rumbaugh, Chapter 6