State Diagrams

PB007 Software Engineering I

Bruno Rossi

7. 11. 2016

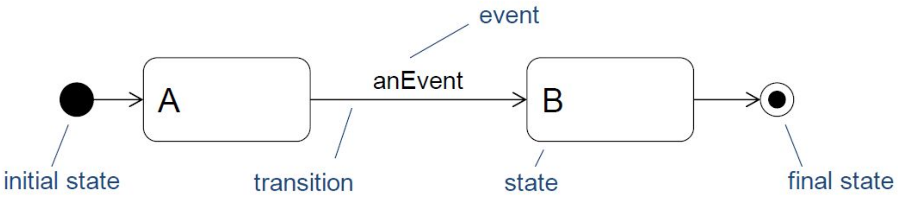
State Diagrams

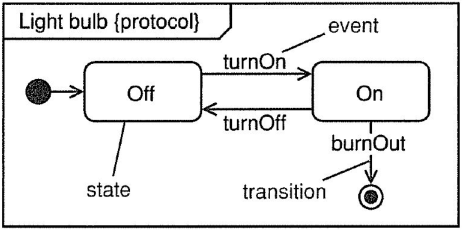
The **State Diagram** models the dynamic behavior (life cycle) of one *reactive object* (class, use case, system, subsystem).

Its essential components are:

- States
- Transitions
- Events

The components of a state diagram



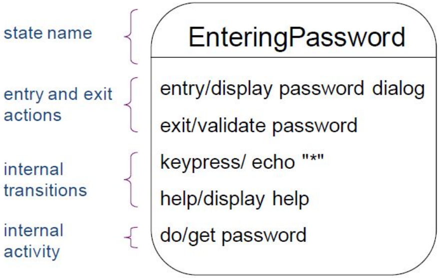


State

The **State** represents a semantically important situation in which the object can be. It is generally determined by the value of attributes, relations with other objects, and activities performed.

Conditions may include:

- Entry/Exit actions
- Internal transitions/actions
- Internal activities



state name

EnteringPassword

entry and exit actions

entry/display password dialog

exit/validate password

internal transitions

keypress/ echo "*"

help/display help

internal activity

do/get password

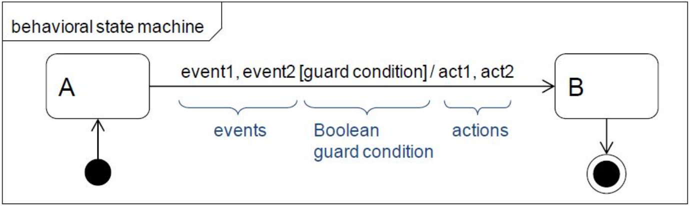
Transitions

Transitions define how you will get from one state to another.

The transition elements:

- Events
- Conditions
- Actions

Syntax: *event[guard condition]/action*
Semantics: At the occurence of the *event*, and with *condition* satisfied, perform *action* and go to the new state.



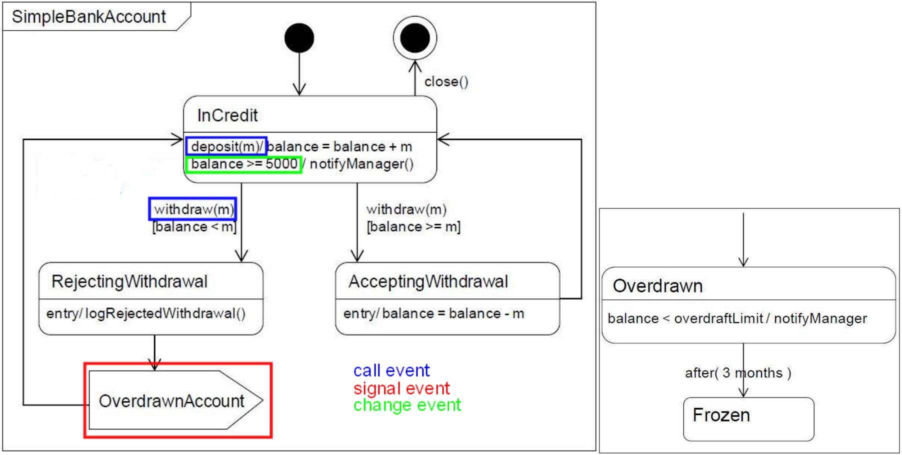
Events

Events are the stimulus to which the subject may react by changing the status or performing some kind of operation. .

We distinguish 4 types of events:

- **Call event** - calling the operations of an object.
- **Signal event** - asynchronous emission and reception of signals from one object to another.
- **Change event** - truth expression. The event occurs after you change its value from false to true.
- **Time event** - events occur at a certain time (*when()*) or after a certain time (*after()*)

Events - examples



call event
signal event
change event

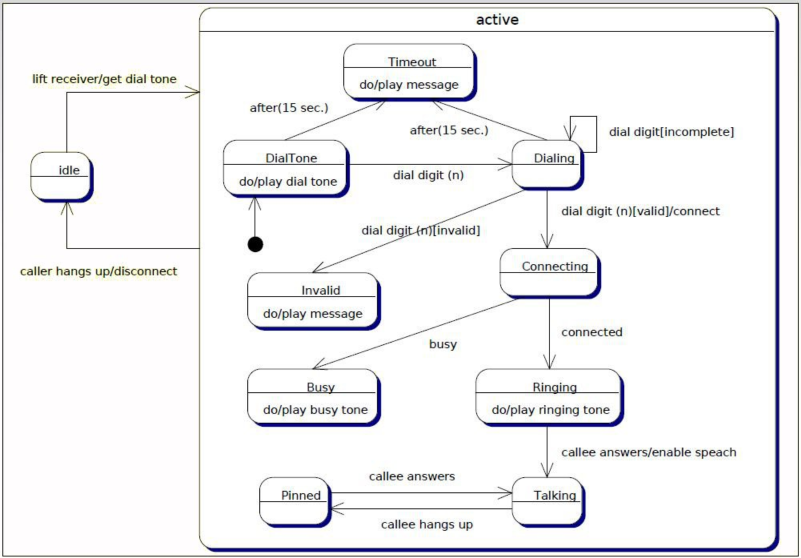
Composite state

Composite states are states that have some nested conditions.

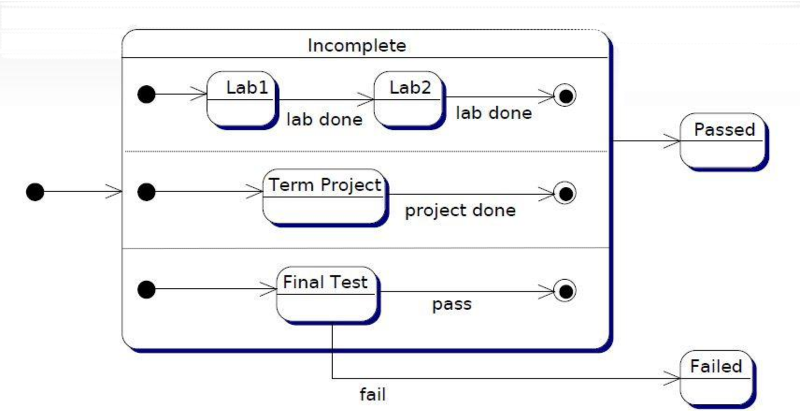
We distinguish two types of composite states:

- **Simple composite states** - that are just in one area - they are suitable to capture the inheritance aspects between different states.
- **Orthogonal composite states** - that cover two or more areas. Each of them contains a nested state diagram. Their execution is carried out in parallel.

Composite state - Example



Composite state - Example II



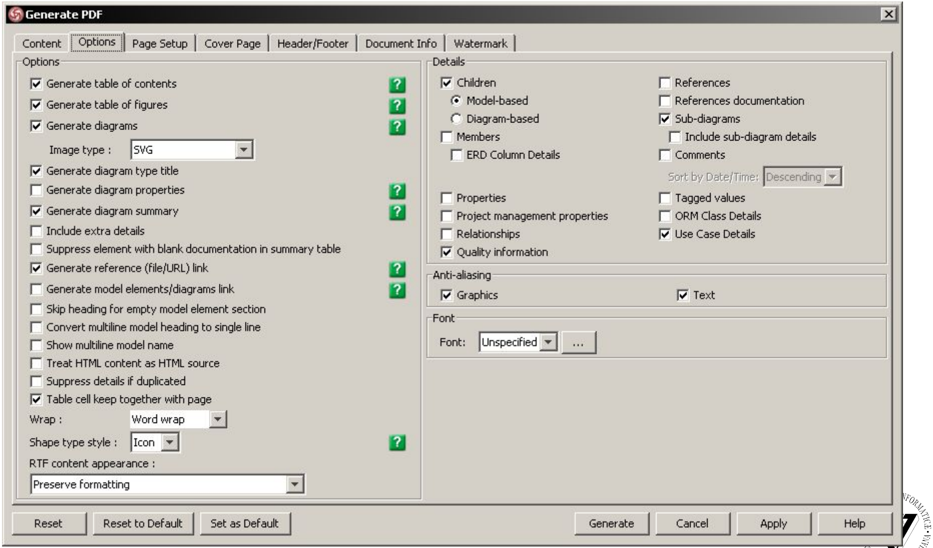
Additional Resources

- http://www.uml-diagrams.org/state-machine-diagrams.html
- http://mpavus.wz.cz/uml/uml-b-state-3-2-2.php
- http://www.agilemodeling.com/artifacts/stateMachineDiagram.htm
- http://www.sparxsystems.com.au/resources/uml2_tutorial/uml2_statediagram.html

Tasks

- Select one reactive object in your system that you will model with a state diagram.
- First, draw up a list of all the states that can be found in the object. Select then those that are really interesting to discriminate with respect to the features of the system.
- Create a list of events (external stimuli) that cause interaction with the object. From the list, select those that affect/alter the state of the object.
- On the basis of the lists of conditions and events, draw the state diagram. Try to take advantage of inheritance between states and thus simplifying the resulting model.
- Upload the **PDF report** into the folder for (**Week 07**).
  Deadline: Friday, 11.11.16 23:59

Customization of PDF Export



Generate PDF

Content | Options | Page Setup | Cover Page | Header/Footer | Document Info | Watermark

Options

- ☑ Generate table of contents
- ☑ Generate table of figures
- ☑ Generate diagrams
  - Image type : SVG
- ☑ Generate diagram type title
- □ Generate diagram properties
- ☑ Generate diagram summary
- □ Include extra details
- □ Suppress element with blank documentation in summary table
- ☑ Generate reference (file/URL) link
- □ Generate model elements/diagrams link
- □ Skip heading for empty model element section
- □ Convert multiline model heading to single line
- □ Show multiline model name
- □ Treat HTML content as HTML source
- □ Suppress details if duplicated
- ☑ Table cell keep together with page

Wrap : Word wrap

Shape type style : Icon

RTF content appearance :

Preserve formatting

Details

- ☑ Children
  - ● Model-based
  - ○ Diagram-based
- □ Members
  - □ ERD Column Details
- □ Properties
- □ Project management properties
- □ Relationships
- ☑ Quality information
- □ References
- □ References documentation
- ☑ Sub-diagrams
  - □ Include sub-diagram details
- □ Comments
  - Sort by Date/Time: Descending
- □ Tagged values
- □ ORM Class Details
- ☑ Use Case Details

Anti-aliasing

- ☑ Graphics
- ☑ Text

Font

Font: Unspecified ...