Systeme hoher Qualität und Sicherheit
Universität Bremen WS 2015/2016

Lecture 07 (23-11-2015)

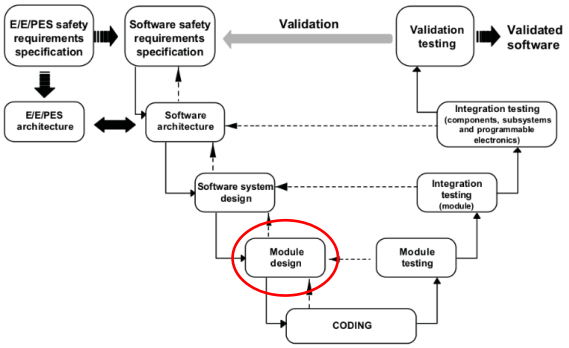
Detailed Specification with SysML

Christoph Lüth Jan Peleska Dieter Hutter

Universität Bremen

Where are we?

Detailed Specification in the Development Cycle



Why detailed Specification?

- **Detailed specification** is the specification of single modules making up our system.
- This is the „last" level both in abstraction and detail before we get down to the code – in fact, some specifications at this level can be automatically translated into code.
- Why **not** write code straight away?
  - We want to stay platform-independent.
  - We may not want to get distracted by details of our target platform.
  - At this level, we have a better chance of finding errors or proving safety properties.

Levels of Detailed Specification

- We can specify the basic modules
- By their (external) **behaviour**:
  - Which operations can be called, what are their pre/post-conditions and effects.
  - This can be modelled using OCL.
  - Alternatively, we can model the system's internal states by a **state machine**, which has states and guarded transitions between them.
- By their (internal) **structure**:
  - Modelling the control flow by flow charts aka. **activity charts.**
  - There are also a variety of **action languages** (platform-independent programming languages) for UML, but these are not standard for SysML.

State Diagrams: Basics

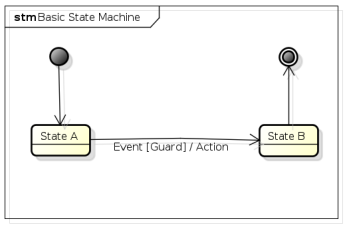
- State diagrams are a particular form of (hierarchical) finite state machines.

A **finite state machine** is given by $M = \langle \Sigma, \rightarrow \rangle$ where
- Σ is a finite set of states, and
- $\rightarrow \subseteq \Sigma \times \Sigma$ is a transition relation which is left-total.

- Example: a simple **coffee machine**.
- We will explore FSMs in detail **later**.
- In hierarchical state machines, a state may contain another FSM (with initial/final states).
- State Diagrams in SysML are taken unchanged from UML.

Basic Elements of State Diagrams

- States
  - Initial/Final
- Transitions
- Events (Triggers)
- Guards
- Actions (Effects)



What is an Event?

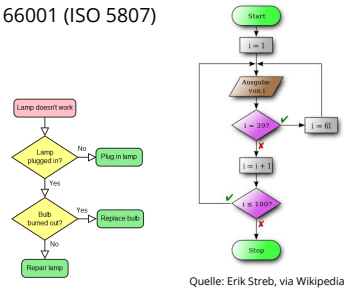
- *„The specification of a noteworthy occurence which has a location in time and space."* (UML Reference Manual)
- SysML knows:
  - Signal events `event name/`
  - Call events `operation name/`
  - Time events `after(t)/`
  - Change event `when(e)/`
  - Entry events `Entry/`
  - Exit events `Exit/`

State Diagram Elements (SysML Ref. §13.2)

- Choice pseudo state
- Composite state
- Entry point
- Exit point
- Final state
- *History pseudo states*
- Initial pseudo state
- Junction pseudo state
- Receive signal action
- Send signal action
- Action
- Region
- Simple state
- State list
- State machine
- Terminate node
- Submachine state

Activity Charts: Foundations

- The activity charts of SysML (UML) are a variation of old-fashioned **flow charts.**
  - Standardised as DIN 66001 (ISO 5807)
- Flow charts can describe programs (right example) or non-computational activities (left example)
- SysML activity charts are extensions of UML activity charts.



Quelle: Erik Streb, via Wikipedia

Quelle: Wikipedia

Basics of Activity Diagrams

- **Activities** model the sequence and conditions for low-level behaviours:
  "An activity is the specification of parameterized behaviour as the coordinated sequencing of subordinate unites whose individual elements are actions." (UML Ref. §12.3.4)
- This is performed by means of **control flow** and **object flow** models
- Control flow allows to **disable** and **enable** (sub-) activities using these two enumeration values.
- An activity execution results in the execution of a set of actions in some specific order.
- Activity executions may comprise several logical **execution threads.**

What is an Action?

- A terminating basic behaviour, such as
  - Changing variable values [UML Ref. §11.3.6]
  - Calling operations [UML Ref. §11.3.10]
  - Calling activities [UML Ref. §12.3.4]
  - Creating and destroying objects, links, associations
  - Sending or receiving signals
  - Raising exceptions .
- Actions are part of a (potentially larger, more complex) behaviour
- Inputs to actions are provided by ordered sets of **pins**
  - A pin is a typed element, associated with a multiplicity
  - **Input pins** transport typed elements to an action
  - Actions deliver outputs consisting of typed elements on **output pins**

Elements of Activity Diagrams (SysML Ref. §11.2.1)

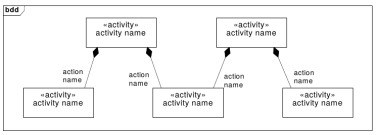
- Nodes:
  - Action nodes
  - Activities
  - Decision nodes
  - Final nodes
  - Fork nodes
  - Initial nodes
  - Local pre/post-conditions
  - Merge nodes
  - Object nodes
  - *Probabilities and rates*
- Paths (arrows):
  - Control flow
  - Object flow
  - Probability and rates
- Activities in BDDs
- Partitions
- Interruptible Regions
- Structured activities

Behavioural Semantics

- Semantics is based on **token flow** – similar to Petri Nets, see [UML Ref. pp. 326]
  - A token can be an input signal, timing condition, interrupt, object node (representing data), control command (call, enable) communicated via input pin, ...
  - An executable node (action or sub-activity) in the activity diagram begins its execution, when the required tokens are available on their input edges.
  - On termination, each executable node places tokens on certain output edges, and this may activate the next executable nodes linked to these edges.

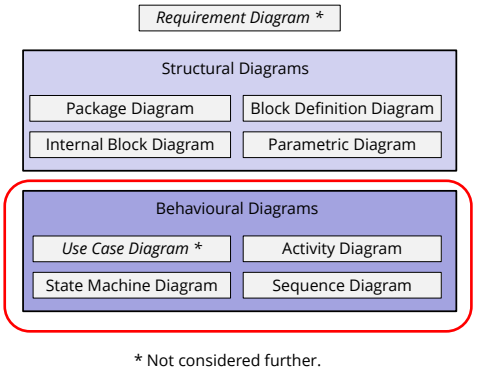
Activity Diagrams – Links With BDDs

- Block definition diagrams may show
  - Blocks representing activities



  - One activity may be composed of other activities – composition indicates parallel execution threads of the activities at the "part end"
  - One activity may contain several blocks representing **object nodes** (which represent data flowing through the activity diagram).

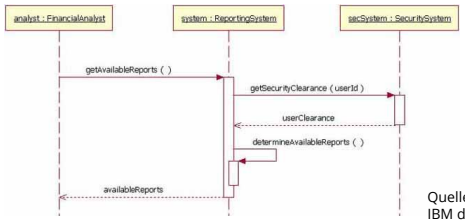
SysML Diagrams Overview



* Not considered further.

Sequence Diagrams

- Sequence Diagrams describe the flow of messages between actors.
- Extremely useful, but also extremely limited.



Quelle:
IBM developerWorks

- We may consider concurrency further later on.

Summary

- Detailed specification means we specify the internal structure of the modules in our systems.
- Detailed specification in SysML:
  - State diagrams are hierarchical finite state machines which specify states and transitions.
  - Activity charts model the control flow of the program.
- More behavioural diagrams in SysML:
  - Sequence charts model the exchange of messages between actors.
  - Use case diagrams describe particular uses of the system.