Lecture 11: Structural Software Modelling

Prof. Dr. Andreas Podelski, Dr. Bernd Westphal
Albert-Ludwigs-Universität Freiburg, Germany

- Introduction and Vocabulary
- Software Modelling
  - views and viewpoints, the 4+1 view
  - model-driven/-based software engineering
  - Unified Modelling Language (UML)
  - Modelling structure
    - (simplified) class diagrams
    - (simplified) object diagrams
    - (simplified) object constraint logic (OCL)
- Principles of Design
  - modularity
  - separation of concerns
  - information hiding and data encapsulation
  - abstract data types, object orientation
- Modelling behaviour
  - communicating finite automata
  - Uppaal query language
  - basic state-machines
  - an outlook on hierarchical state-machines

- Design Patterns

Example: Design-Models in Construction Engineering

1. Requirements
   - Shall fit on given piece of land.
   - Each room shall have a door.
   - Furniture shall fit into living room.
   - Bathroom shall have a window.
   - Cost shall be in budget.

2. Design model

3. System

Observation (1):
Floorplan abstracts from certain system properties, e.g.,
- kind, number, and placement of bricks,
- subsystem details (e.g., window style),
- water pipes/wiring, and
- wall decoration.

Notice architects can efficiently work on appropriate level of abstraction.

Observation (2):
Floorplan preserves certain system properties, e.g.,
- house and room extensions (to scale),
- presence/absence of windows and doors,
- placement of subsystems (such as windows).

Find design errors before building the system (e.g., bathroom windows).
No sharp boundaries! (would be too easy...)

Note.

...part not functionality; functionality is support...

→ employed framework... 

Example


Newer proposals...

Logical View

...structure vs. behaviour / constructive vs. reflective...

Dynamic View

...in executing the model or in translating it into executable code...

Static View

...of behaviour:

\[
\pi(S) \Rightarrow S \quad \text{or} \quad \pi(S) \Rightarrow \emptyset
\]

Scenarios...}

Physical View

...deployment view...

Process View

...process and physical view may be trivial or determined by the...

Program view:

...assertive (or reflective) constructs...}

Deployment / Physical View

...how is the system under...

Logical View

...computation paths...

Developer View

...which part of the overall software is running on which ECU?

System View

...how are things computed how...

Scenario View

...how and when...

Logical View

...which function is used when? Event triggered, time triggered, continuous, etc.?...

Process View

...how are things computed how...

Physical View

...how is the system under...

System View

...how are things computed how...

System View

...users) does it...
• **Introduction and Vocabulary**

• **Software Modelling**
  
  (i) views and viewpoints, the 4+1 view
  
  (ii) model-driven/-based software engineering
  
  (iii) Unified Modelling Language (UML)
  
  (iv) Modelling structure
    
    a) (simplified) class diagrams
    
    b) (simplified) object diagrams
    
    c) (simplified) object constraint logic (OCL)
  
  (v) Principles of Design
    
    a) modularity
    
    b) separation of concerns
    
    c) information hiding and data encapsulation
    
    d) abstract data types, object orientation
  
  (vi) Modelling behaviour
    
    a) communicating finite automata
    
    b) Uppaal query language
    
    c) basic state-machines
    
    d) an outlook on hierarchical state-machines

• **Design Patterns**
  
  VL 10...
  
  VL 11...
  
  VL 12...
  
  VL 13...

---

**Class Diagrams**

• **Concrete Syntax**

• **Abstract Syntax**

• **Class Diagrams at Work**

• **Semantics: System States**

**Object Diagrams**

• **Concrete Syntax**

• **Dangling References**

• **Partial vs. Complete**

• **Object Diagrams at Work**

**Software Modelling Cont'd**

• **An Outlook on UML**

• **Model-driven software engineering**
Once Again: Concrete vs. Abstract Syntax

Concrete Syntax: Example

Abstract Syntax: Concrete Syntax Example

Definition. Object System Signature Example

And nothing else!
A structure is a mapping of system state

System State Examples

Basic Object System Structure Example
null reference in parent

Example Diagram of Object

Example Diagram of Object
Object Diagrams for Analysis

• Software Modelling
  • views & viewpoints
  • the 4+1 view
• Class Diagrams
  • concrete syntax,
  • abstract syntax,
  • class diagrams at work,
  • semantics: system states.
• Object Diagrams
  • concrete syntax,
  • dangling references,
  • partial vs. complete,
  • object diagrams at work.

Software Modelling Cont'd
• An outlook on UML
  • model-driven software engineering
• Unified Modelling Language
  • a family of modelling languages,
  • including languages to model structure and behaviour.
• Class Diagrams
  • can be used to graphically visualise code,
  • define an object system structure $S$.
• An Object System Structure $S$ (together with a structure $D$)
  • defines a set of system states $\Sigma_{DS}$.
• A System State $\sigma \in \Sigma_{DS}$
  • can be visualised by an object diagram.

References