Wrapup: Motivation

Lecture 1:

Educational Objectives:
you should
• be able to explain the term model.
• know the idea (and hopes and promises) of model-driven SW development.
• be able to explain how UML fits into this general picture.
• know what we’ll do in the course, and why.
• thus be able to decide whether you want to stay with us...

How can UML help with software development?
• Where is which sub-language of UML useful?
• For what purpose? With what drawbacks?
Wrapup: Examining Motivation

- What is a model? For example?
- "A model is an image or a pre-image"—of what? Please explain!
- When is a model a good model?
- What is model-based software engineering?
- MDA? MDSE?
- What do people hope to gain from MBSE? Why? Hope justified?
- What are the fundamental prerequisites for that?
- What are the purposes of modeling guidelines?
- Could you illustrate this with examples?
- How can we establish/enforce them? Can tools or procedures help?
- What's the qualitative difference between the modeling guidelines "all association ends have a multiplicity" and "all state-machines are deterministic"?

Wrapup: "The Big Picture"

- What kinds of diagrams does UML offer?
- What is the purpose of the X diagram?
- What do the diagrams X and Y have in common?
- What is a UML model (our definition)? What does it mean?
- What is the difference between well-formedness rules and modeling guidelines?
- What is meta-modelling?
- Could you explain it on the example of UML?
- What is a class diagram in the context of meta-modelling?
- What benefits do people see in meta-modelling?
- The standard is split into the two documents "Infrastructure" and "Superstructure". What is the rationale behind that?
- In what modelling language is UML modelled?

Wrapup: Modelling Structure

- Lecture 1: Motivation and Overview
- Lecture 2: Semantic Model
- Lecture 3: Object Constraint Language (OCL)
- Lecture 4: OCL Semantics
- Lecture 5: Object Diagrams
- Lecture 6: Class Diagrams I
- Lecture 7: Type Systems and Visibility
- Lecture 8: Class Diagrams II
- Lecture 9: Class Diagrams III
- Lecture 10: Constructive Behaviour, State Machines Overview
- Lecture 11: Core State Machines I
- Lecture 12: Core State Machines II
- Lecture 13: Core State Machines III
- Lecture 14: Core State Machines IV
- Lecture 15: Core State Machines V, Rhapsody
- Lecture 16: Hierarchical State Machines I
- Lecture 17: Hierarchical State Machines II
- Lecture 18: Live Sequence Charts I
- Lecture 19: Live Sequence Charts II
- Lecture 20: Inheritance I
- Lecture 21: Meta-Modelling, Inheritance II
- Lecture 22: Wrapup & Questions
Wrapup: Modelling Structure

Lecture 7:
- Educational Objectives: Capabilities for following tasks/questions.
- Is this OCL expression well-typed or not? Why?
- How/ in what form did we define well-definedness?

Lecture 8 & 9:
- Educational Objectives: Capabilities for following tasks/questions.
- Please explain/illustrate this class diagram with associations.
- Which annotations of an association arrow are (semantically) relevant? In what sense? For what?
- What's a rolename? What's it good for?
- What's "multiplicity"? How did we treat them semantically?
- What's "reading direction", "navigability", "ownership", .. . ?
- What's the difference between "aggregation" and "composition"?

Lecture 9:
- Educational Objectives: Capabilities for following tasks/questions.
- What are purposes of modelling guidelines? (Example?)
- When is a class diagram a good class diagram?
- Discuss the style of this class diagram.

Lecture 20 & 21:
- Educational Objectives: Capabilities for following tasks/questions.
- What's the effect of inheritance on System States?
- What does the Liskov Substitution Principle mean regarding structure?
- What is the subset, what the uplink semantics of inheritance?
- What's the idea of Meta-Modelling?

Wrapup: Modelling Behaviour, Constructive

Lecture 1:
- Motivation and Overview

Lecture 2:
- Semantical Model

Lecture 3:
- Object Constraint Language (OCL)

Lecture 4:
- OCL Semantics

Lecture 5:
- Object Diagrams

Lecture 6:
- Class Diagrams I

Lecture 7:
- Type Systems and Visibility

Lecture 8:
- Class Diagrams II

Lecture 9:
- Class Diagrams III

Lecture 10:
- Constructive Behaviour, State Machines Overview

Lecture 11:
- Core State Machines I

Lecture 12:
- Core State Machines II

Lecture 13:
- Core State Machines III

Lecture 14:
- Core State Machines IV

Lecture 15:
- Core State Machines V, Rhapsody

Lecture 16:
- Hierarchical State Machines I

Lecture 17:
- Hierarchical State Machines II

Lecture 18:
- Live Sequence Charts I

Lecture 19:
- Live Sequence Charts II

Lecture 20:
- Inheritance I

Lecture 21:
- Meta-Modelling, Inheritance II

Lecture 22:
- Wrapup & Questions
• Educational Objectives: Capabilities for following tasks/questions.
• What's a kind of a state? What's a pseudo-state?
• What's a region? What's it good for?
• What is: entry, exit, do, internal transition?
• What's a completion event? What has it to do with the ether?

• Educational Objectives: Capabilities for following tasks/questions.
• What's a state configuration?
• When are two states orthogonal? When consistent?
• What's the depth of a state? Why care?
• What is the set of enabled transitions in this system configuration and this state machine?

• Educational Objectives: Capabilities for following tasks/questions.
• What's a historic state? Deep vs. shallow?
• What is: junction, choice, terminate?
• What is the idea of "deferred events"?
• What is a passive object? Why are passive reactive objects special? What did we do in that case?
• What's a behavioural feature? How can it be implemented?

• Educational Objectives: Capabilities for following tasks/questions.
• IseachLSC description of behaviour necessarily reflective?
• There exists another distinction between "inter-object" and "intra-object" behaviour. Discuss in the context of UML.
• What does this LSC mean?
• Are these UML model's state machines consistent with the interactions?
• Please provide a UML model which is consistent with this LSC.
• What is: activation (mode, condition), hot/cold condition, pre-chart, cut, hot/cold location, local invariant, legalexit, hot/cold chart etc.
Hmm...

• Open book or closed book?