Foundations of the UML

Joost-Pieter Katoen

Lehrstuhl für Informatik 2
Software Modeling and Verification Group

http://moves.rwth-aachen.de/i2/370

19. Oktober 2009

Outline

1 Lecture 1: Introduction

Presentation outline

1. Lecture 1: Introduction

Foundations of the UML

Lecture 1: Introduction

Joost-Pieter Katoen

Lehrstuhl für Informatik 2
Software Modeling and Verification Group

http://moves.rwth-aachen.de/i2/370

19. Oktober 2009

Target audience

You are studying:

- Diplom Programme Informatik, or
- Master Computer Science, or
- Master Systems Software Engineering, or
- Bachelor Computer Science

Target audience

You are studying:

- Diplom Programme Informatik, or
- Master Computer Science, or
- Master Systems Software Engineering, or
- Bachelor Computer Science

Usage as:

- elective course Theoretical Computer Science
- not a Wahlpflicht course for bachelor students
- specialization MOVES (Modeling and Verification of Software)
- complementary to Model-based Software Development (Rumpe)

Target audience (contd.)

In general:

- interest in system software engineering
- interest in formal methods for software
- interest in semantics and verification
- application of mathematical reasoning

Target audience (contd.)

In general:

- interest in system software engineering
- interest in formal methods for software
- interest in semantics and verification
- application of mathematical reasoning

Prerequisites:

- mathematical logic
- formal language and automata theory
- algorithms and data structures
- computability and complexity theory

Organization

Schedule:

Type	Day	Time	Lecture hall
Lecture	Mon	15:00 - 16:30	5052
	Tue	10:00 - 11:30	5056
Exercises	Fri	10:00 - 11:30	AH II

about 19 lectures in total; Keep track of website for precise dates!

Organization

Schedule:

Type	Day	Time	Lecture hall
Lecture	Mon	15:00 - 16:30	5052
	Tue	10:00 - 11:30	5056
Exercises	Fri	10:00 - 11:30	AH II

about 19 lectures in total; Keep track of website for precise dates!

People involved:

Type	Lecturer	EMail
Lecture	Joost-Pieter Katoen	`katoen@cs.rwth-aachen.de`
Exercises	Tingting Han	`tingting.han@cs.rwth-aachen.de`
	Alexandru Mereacre	`mereacre@cs.rwth-aachen.de`

Organization (contd.)

Assignments:

- (almost) weekly assignments
- first assignment: available from course web-site Friday October 23
- hand in solution at next exercise class
- groups of maximally two students

Organization (contd.)

Examination: (6 ECTS credit points)

- written/oral exam (depending on the number of students)
- proposal: Friday February 5, 2010

Organization (contd.)

Examination: (6 ECTS credit points)

- written/oral exam (depending on the number of students)
- proposal: Friday February 5, 2010

Admission:

- at least 50% of exercise points

Motivation

Scope:

- Goal: formal description + analysis of (concurr.) software systems
- Focus: the Unified Modeling Language

Motivation

Scope:

- Goal: formal description + analysis of (concurr.) software systems
- Focus: the Unified Modeling Language

More specifically:

- Sequence Diagrams (used for requirements analysis), PDL
- Hierarchical State Machines (behavioral description of systems)
- The Object Constraint Language (OCL) (property specification of UML diagrams)

Motivation

Scope:

- Goal: formal description + analysis of (concurr.) software systems
- Focus: the Unified Modeling Language

More specifically:

- Sequence Diagrams (used for requirements analysis), PDL
- Hierarchical State Machines (behavioral description of systems)
- The Object Constraint Language (OCL) (property specification of UML diagrams)

Aims:

- clarify and make precise the semantics of treated UML fragments
- formal reasoning about basic properties of UML models
- algorithms to verify such properties

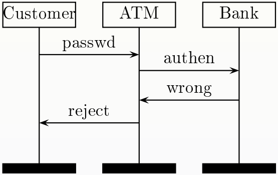
What this course is **NOT** about:

What is it **not** about?

- the use of the UML in the software development cycle
    - see the complementary course by Prof. Rumpe
- other notations of the UML (e.g., class diagrams, activity diagrams)
- what is precisely in the UML, and what is not
    - liberal interpretation of which constructs belong to the UML
- applying the UML to concrete SW development case studies
- empirical results on the usage of UML
- drawing pictures
- . . .

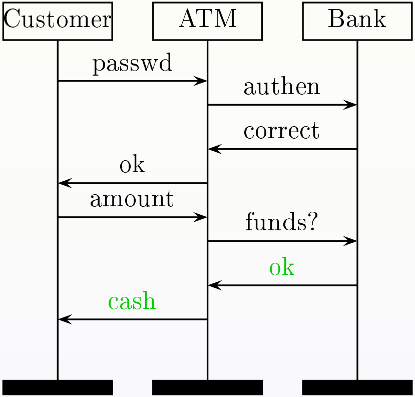
Sequence Diagrams

- origin: telecommunications: "Message Sequence Charts" (MSCs)
- describe interactions between processes (or objects)
- attractive visual formalism

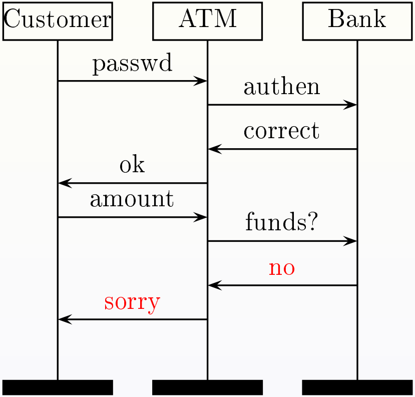


- describes a possible scenario
- standardized by the ITU (Z. 120)
- adopted by the OMG for UML

Another example MSC



Another example MSC

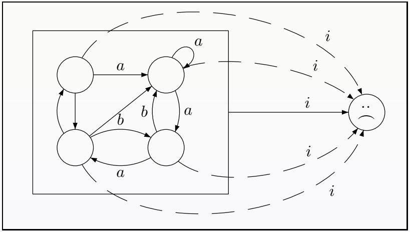


Message Sequence Charts

- MSCs
  (syntax, semantics, linearizations, races)
- Message sequence graphs
  (composition, expressiveness, compositional MSCs)
- Realizability
  (communicating finite-state machines, reachability in CFSMs, MSCs vs. CFSMs, boundedness)
- Regularity
  (regular MSCs and MSGs, realizability)
- Verification
  (positive + negative model checking, complexity results, basic properties: `MSCan`)
- PDL
  (Propositional Dynamic Logic for checking MSC properties)

Hierarchical State Machines

- finite state machines
  - no strategy for top-down or bottom-up development ("states have <u>no</u> structure")
  - no natural notion of hierarchy
  - uneconomical concerning transitions (e.g., high-level interrupt)

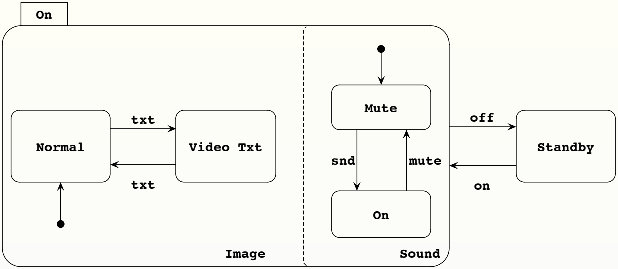


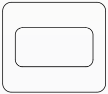
  - uneconomical wrt. parallel composition (exponential growth in # states)

Statecharts

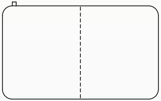
Statecharts = Mealy machines
+ depth
+ orthogonality [Harel'86]
+ broadcast
+ data

Statecharts (contd.)

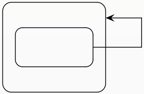




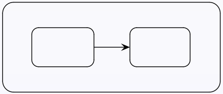
hierarchical state



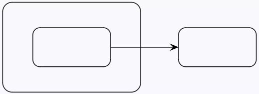
AND state



intra-level transition



inter-level transition



intra-level transition

Statecharts (contd.)

- Harel's Statecharts
  (basic features, syntax, state hierarchy, orthogonality, intra- and inter-level transitions)
- Semantics
  (main issues, formal semantics, flattening, succinctness)
- Verification
  (expressiveness, reachability, LTL model checking)

Object Constraint Language

- allows specification of basic properties on objects:

Example

context Room invariant

guest→size ≤ numOfBeds

context Hotel::checkIn (g:Guest)

pre not guests→includes(g)

post guests→size = (guests@pre→size)+1

and guests→includes(g)

- not related to particular diagram of UML
- often: annotations to different types of UML diagrams (e.g. class diagrams, activity diagrams, statecharts, ...)

Object Constraint Language (contd.)

Topics:

- OCL basics
  (types, operations, navigation, class diagrams)
- semantics of the OCL
  (operational model, logic, types + values)
- Embedding into temporal logic
  (LTL/CTL, from OCL to temporal)