An Analogy: Construction Engineering

Floorplans as Formal Specification Language

The Notion of Model

"Floorplans" for Software

Goals, Content and Non-Content of the Course

The UML Standard Documents

The Map

A Brief History of UML

UML Models

Course Organisation

Lectures

Tutorials

Exam

Recall: Model

Definition.

[Folk]

A model is an abstract, formal, mathematical representation or description of structure or behaviour of a (software) system.

Definition.

([425], 425)

A model is a concrete or mental image (Abbild) of something or a concrete or mental archetype (Vorbild) for something.

Three properties are constituent:

(i) the image attribute (Abbildungsmerkmal), i.e. there is an entity (called original) whose image or archetype the model is,

(ii) the reduction attribute (Verkürzungsmerkmal), i.e. only those attributes of the original that are relevant in the modelling context are represented,

(iii) the pragmatic attribute, i.e. the model is built in a specific context for a specific purpose.
Goal: A Common, Precise Understanding of UML Models

(i) We need to know how the words of the language look like: Syntax.
(UML example: is this a proper UML state machine diagram?)

(ii) We need to know what a word of the language means: Semantics.
→ Then we can formally analyse the model, e.g., prove that the design satisfies the requirements, simulate the model, automatically generate test cases, automatically generate equivalent code, etc.
(UML example: can sending event E and then G kill the object?)

• UML is sometimes (neutrally, or as offence) called "semi-formal": the UML standard is strong on (i), but weak(er) on (ii).
("the diagram is self-explanatory" , "everybody understands the diagram" — No.)

• In the lecture: study the (!) syntax, define one (!) semantics.
A D C ∈ \mathbb{R}_0^+ \Rightarrow \emptyset S Q UML : ChoicePanel & 50
• Entry Action:itsChoicePanel Outlook: Concrete vs. Abstract Syntax
A Brief History of the Unified Modelling Language (UML)

- Boxes/lines and finite automata are used to visualise software for ages.
  Modelling languages: Flowcharts, Nassi-Shneiderman, Entity-Relation Diagrams
- Mid 1980's: Statecharts (?), StateMate™ (?)
- Early 1990's, advent of Object-Oriented-Analysis/Design/Programming—Inflation of notations and methods, most prominent:
  - Object-Modeling Technique (OMT) (?)
  - Booch Method and Notation (?)
  - Object-Oriented Software Engineering (OOSE) (?)

Each "persuasion" selling books, tools, seminars...
As needed.

Domain Specific Languages or define your own UML semantics for your own context/purpose,
you."

So when someone else’s view of the UML seems rather different to yours, it may be because they use a different mode,

“... people differ about what should be in the UML because there are differing fundamental views about what the UML

should be.

After the course, you should...: Side Effects

If you can detail the UML enough, and provide semantics for everything you need in software, you can make the UML be your...

This promise is true. I don’t believe that graphical programming will succeed just because it’s graphical.

In forward engineering the idea is that blueprints are developed by a designer whose job is to build a detailed...

In this UmlMode developers use the UML to help communicate some aspects of a system. [...]

Sketches are also useful in rather than complete specification. Hence my sound-bite “comprehensiveness is the enemy of comprehensibility.”

With UML it’s the same: http://martinfowler.com/bliki

... diagrams and back it up with a repository to hold the information. [...]

In Object-Oriented Software Engineering (OOSE), the solution was a suite of notations for analysis, design and programming, with a unique style for each.

Programming Language Blueprint Sketch

Program: Blueprint: Sketch

Floorplan and UML Modes!

A Brief History of the Unified Modelling Language (UML)

The standards are published by the Object Management Group (OMG), an international, open membership, non-profit consortium for technology and business standards in the computer industry.

UML 0.x: joint effort of “the three amigos” yielded late 1980’s: Object-Oriented Software Engineering (OOSE) - Analysis/Design/Programming

UML 1.x: mid 1990’s - Inflation of notations and methods, most prominent:

- Statecharts (Jens Ullrich, UCSD), StateMate (Ullrich), Statecharts’ s:

- OMT - Object-Oriented Modeling Technique (Grady Booch, Ivar Jacobson, Jim Rumbaugh)

- Flowcharts, Nassi-Shneiderman, Entity-Relation Diagrams

- Software Crisis

- Idea: learn from engineering disciplines to handle growing complexity.

Since 1990’s - advent of UML - Aims: unified models; formal analysis - to allow avoid misunderstandings - to find errors early - to be consistent with (informal semantics in)

Floorplan and UML Modes!
An Analogy: Construction Engineering

Floorplans as Formal Specification Language

The Notion of Model

“Floorplans” for Software

Goals, Content and Non-Content of the Course

The UML Standard Documents

The Map

A Brief History of UML

UML Modes

Course Organisation

Lectures

Tutorials

Exam

Formalia: Lectures

Lecturer: Dr. Bernd Westphal

Support: Claus Schätzle

Homepage: http://swt.informatik.uni-freiburg.de/teaching/WS2016-17/sdmauml

Time/Location: Tuesday, Thursday, 8:00 – 10:00 / here (building 51, room 03-026)

Course language: English (slides/writing, presentation, questions/discussions)

Presentation: half slides/half on-screen hand-writing — for reasons

Script/Media: slides with annotations on homepage, typically soon after the lecture recording on ILIAS with max. 1 week delay (links on homepage)

Break: We’ll have a 10 min. break in the middle of each event from now on, unless a majority objects now.

Formalia: Exercises and Tutorials

You should work in groups of approx. 3, clearly give names on submission.

Please submit via ILIAS (cf. homepage); paper submissions are tolerated.

Schedule:

Week N, Thursday, 8–10 Lecture A1 (exercise sheet A online)

Week N + 1, Tuesday 8–10 Lecture A2

Thursday 8–10 Lecture A3

Week N + 2, Monday, 12:00 (exercises A early submission)

Tuesday, 8:00 (exercises A late submission)

8–10 Tutorial A

8–10 Lecture B1 (exercise sheet B online)

. . .

Rating system: “most complicated rating system ever

Admission points (good-will rating, upper bound)

Exam-like points (evil rating, lower bound)

10% bonus for early submission.

Tutorial: Plenary, not recorded.

Together develop one good solution based on selection of early submissions (anonymous) — there is no “Musterlösung” for modelling tasks.

Formalia: Exam

Exam Admission: Achieving 50% of the regular admission points in total is sufficient for admission to exam.

Typically, 20 regular admission points per exercise sheet; some exercise sheets have bonus tasks.

Exam Form:

oral for BSc and on special demand (Erasmus),

written for everybody else (if sufficiently many candidates remain).

Scores from the exercises do not contribute to the final grade.

Exam Date: Please remind me in early December that we need to agree on an exam date.

Formalia: User’s Guide

Approach: The lectures is supposed to work as a lecture: spoken word + slides + discussion. It is not our goal to make any of the three work in isolation.

Interaction: Absence often moaned but it takes two: please ask/comment immediately.

Exercise submissions: Each task is a tiny little scientific work:

(i) Briefly rephrase the task in your own words.

(ii) State your claimed solution.

(iii) Convince your reader that your proposal is a solution (proofs are very convincing).
• Approach: The lectures is supposed to work as a lecture: spoken word + slides + discussion. It is not our goal to make any of the three work in isolation.

• Interaction: Absence often moaned but it takes two: please ask/comment immediately.

• Exercise submissions: Each task is a tiny little scientific work:
  1. Briefly rephrase the task in your own words.
  2. State your claimed solution.
  3. Convince your reader that your proposal is a solution (proofs are very convincing).

Example:

Task: Given a square with side length $a = 19.1$. What is the length of the longest straight line fully inside the square?

Submission A:

Submission B:

The length of the longest straight line fully inside the square with side length $a = 19.1$ is 27.01 (rounded). The longest straight line inside the square is the diagonal. By Pythagoras, its length is $\sqrt{a^2 + a^2}$. Inserting $a = 19.1$ yields 27.01 (rounded).