Modeling for Software Design and UML

CS20006: Software Engineering

Prof. Partha Pratim Das

Source: ftp://ftp.ifs.uni-linz.ac.at/pub/wieland/uml.ppt

भारतीय प्रौद्योगिकी संस्थान खड़गपुर
Indian Institute of Technology
Kharagpur

March 2021

UML - Outline

- Introduction to Modeling
- Requirements Specification
- Analysis
- Design
- Pitfalls and Workarounds
- Roadmap to UML 2.0
- References

} UML Syntax and Semantics

Software Engineering Model

WHAT IS A MODEL?

WHY WE NEED MODELS?

Motivation for Analysis and Design

Why do we model?

"When it comes down to it,
the real point of software development is cutting code"

"Diagrams are, after all, just pretty pictures"

"No user is going to thank you for pretty pictures; what a user wants is software that executes"

[M. Fowler, "UML Distilled", Addison Wesley, 1997]

Models we may have seen …

- Physics
  - Time-Distance Equation
- Chemistry
  - Valency-Bond Structures
- Mathematics
  - All about models …
- Geography
  - Maps
  - Projections
- Electrical Circuits
  - Kirchoff's Loop Equations
  - Time Series Signals & FFT
  - Transistor Models
- Building & Bridges
  - Drawings
  - Finite Element Models
- Machine Design
  - Differential Equations

Why do we need models?

- Solutions Method: Real Systems may not be available, accessible, affordable …
  - Represent the System as a Model
  - Solve problems on the Model
  - Map the solutions back to the System
- Human Cognition Mechanism is limited
  - Short Term & Long Term Memory
  - Ability to track only up to 7 entities
- Models are Abstractions that help track complexity

Implication in Program Development

- A small program having just a few variables:
  - Is within easy grasp of an individual.
- As the number of independent variables in the program increases:
  - It quickly exceeds the grasping power of an individual:
    - Requires an unduly large effort to master the problem.

Implication in Program Development

- Instead of a human, if a machine could be writing (generating) a program,
  - The slope of the curve would be linear.
- But, why does the effort-size curve become almost linear when software engineering principles are deployed?
  - Software engineering principles extensively use techniques specifically to overcome the human cognitive limitations.

How to overcome Human Cognitive Limitations?

- Two important principles are profusely used in SE:
  - Abstraction
  - Decomposition

Abstraction

- Simplify a problem by omitting unnecessary details.
  - Focus attention on only one aspect of the problem and ignore irrelevant details.
  - Also called **Model Building**

Abstraction

- Suppose you are asked to develop an overall understanding of some country.
  - Would you:
    - Meet all the citizens of the country, visit every house, and examine every tree of the country?
  - You would possibly only refer to various types of maps for that country.

Abstraction

- A map is:
  - An abstract representation.

Abstraction

- Several abstractions of the same problem possible:
  - Focus on some specific aspect and ignore the rest.
  - Different types of models help understand different aspects of the problem.

Abstraction - Views

- Maps
  - Physical
  - Political
  - Road Network
  - Rivers
- Electrical Design
  - Schematic
  - Layout
  - Timing
- Building Design
  - Physical
    - Plan
    - Elevation
  - Water Supply
  - Electrical Distribution
  - Sewage

Abstraction

- For complex problems:
  - A single level of abstraction is inadequate.
  - A hierarchy of abstractions needs to be constructed.
- Hierarchy of models:
  - A model in one layer is an abstraction of the lower layer model.
  - An implementation of the model at the higher layer.

Decomposition

- Decompose a problem into many small independent parts.
  - The small parts are then taken up one by one and solved separately.
  - The idea is that each small part would be easy to grasp and can be easily solved.
  - The full problem is solved when all the parts are solved.

Decomposition

- A popular use of decomposition principle:
  - Try to break a bunch of sticks tied together versus breaking them individually.
- Any arbitrary decomposition of a problem may not help.
  - The decomposed parts must be more or less independent of each other.

Decomposition Example

- Example use of decomposition principle:
  - You understand a book better when the contents are organized into independent chapters.
  - Compared to when everything is mixed up.

Decomposition is Hierarchical

- Decompose the WHOLE into PARTs
- Decompose each PART into SUB-PARTs
- Decompose each SUB-PART into SUB-SUB-PARTs

Decomposition Hierarchy Examples

- Books
  - Chapters
    - Sections
      - Paragraphs
        - Sentences
        - …
- Cars
  - Engine
    - Piston
    - Cylinders
  - Wheels
    - Tyre
    - Tube
  - Steering
  - Brakes
  - AC
  - Seats
  - …

Modelling Relations for Hierarchies

- Abstraction Hierarchy
  - IS-A
- Decomposition Hierarchy
  - HAS-A

Motivation for Analysis and Design

The Model as an Abstraction of the Reality

- **We are not able to comprehend a complex system in its entirety -** a single model is not enough
  - **different perspectives** are required, which in turn require **different models** being independent from each other
  - each model must exist on **different levels of granularity**

- **Good models are necessary ...**
  - for making complex systems more understandable
  - for visualizing the essential aspects of a system
  - for communication among project members and with the customer
  - for ensuring architectural soundness