Model Driven Engineering: Basic Concepts

Lesson 1
Context and Definitions

Guglielmo De Angelis
CNR - IASI / ISTI
guglielmo.deangelis@isti.cnr.it

PhD in Computer Sciences
... what they are?
... however in a very similar context ...

Leci n’est pas une pipe.
... so what's about them, now?
what are the “MODELS”? how the “MODELS” are used?

are they used in practice? are they used designing/developing software?
to help us understand complex systems
  - useful for both requirements and designs
  - minimize risks by detecting errors and omissions early in the design cycle (at low cost)
    - through analysis and experimentation
    - investigate and compare alternative solutions
  - to communicate understanding
    - Stakeholders: Clients, users, implementers, testers, documenters, etc.

- to drive implementation
  - the model as a blueprint for construction
an intuitive example ...

- building plans are a kind of engineering models:
ok, fine. but ... ... 

1. WHAT'S ABOUT SOFTWARE ?!?!?!?
2. WHAT'S ABOUT ENGINEERING SOFTWARE ?!?!?!?
model-driven engineering

• design of software based on use of:
  – structural models
  – behavioral models
  – QoS specifications
    • quantitative and qualitative analysis techniques and computer simulation

• advantages:
  – early detection of design flaws and inadequacies
  – increased productivity
  – higher reliability (simplification due to use of models)
  – platform independence
sw && models : to represent
sw && models : to synthesize
sw && modes : to analyze
sw && modes : automation
what is a model (in MDE)

• A simplification of a system built with an intended goal in mind. The model should be able to answer questions in place of the actual system. [Bézivin et al 2001]

• A description or specification of that system and its environment for some certain purpose. A model is often presented as a combination of drawings and text. [MDA Guide 2003]

• A coherent set of formal elements describing something built for some purpose that is amenable to a particular form of analysis. [Mellor et al 2003]
what is a model (in MDE)

- A **simplification of a system** built with an intended goal in mind. The model should be able to answer questions **in place of** the actual system. [Bézivin et al 2001]

- A **description or specification of that system** and its environment **for some certain purpose**. A model is often presented as a **combination of drawings and text**. [MDA Guide 2003]

- A **coherent set of formal elements** describing something **built for some purpose** that is amenable to a particular form of analysis. [Mellor et al 2003]
modeling languages -1-

- set of modeling languages for specific purposes
  - flow charts
  - state machines
  - petri net
  - ER diagrams
modeling languages -2-

- set of modeling languages for specific purposes
modeling languages -3-

- set of modeling languages for specific purposes:
  - but also in some sense:

  Java
  C++
  Perl

  SQL
  PHP

are kinds of imperative/declarative modeling languages that programmers use in order to “model” (i.e. implement) software systems
which kind of model?

• the choice can be driven by:
  – the problem that has to be solved
  – the granularity of the problem
  – the granularity of a potential solution
  – acceptance of the notation in the considered context
  – tools support
  – maintenance and evolution of the artifacts
  – … … many many others criteria
characteristics of useful models

- abstract
  - emphasize important aspects while removing irrelevant ones
- understandable
  - expressed in a form that is readily understood by observers
- accurate
  - faithfully represents the modeled system
- predictive
  - can be used to answer questions about the modeled system
- inexpensive
  - much cheaper to construct and study than the modeled system

*To be useful, engineering models must satisfy all of these characteristics!*
just only one point of view ?!?!
going back to the house example ...
going back to the house example ...
the multiple aspects of a system

- a (software) system manifests several aspects
  - functional
    - relative to requirements from the users
    - relative to requirements from the implementation
    - relative to requirements from laws
  - extra-functional
    - relative to the offered QoS
    - relative to constraints on the available resources
  - organizational
    - relative on business aspects
    - relative to the specific context
- each aspect has to be properly addressed
• how to manipulate models and software artifacts?
In a “similar” domain … …

- As natural languages, programming languages are defined by means of grammars
- Often formal grammars are expressed by means of another (formal) language
metamodelling

- in MDE models are specified according to “grammars” called metamodels
- metamodels are defined by means of other languages called: meta-metamodels
A problem in the real world
a concrete example:
• how to manipulate models and software artifacts?
  – metamodeling
  – ……
• how to manipulate models and software artifacts?
  – metamodeling
  – automatic transformations
automatic transformations
[“A Taxonomy of Model Transformation” Mens & Van Gorp 2006]

• what needs to be transformed into what?
• number of source and target models
• technical space
• endogenous and exogenous transformations
• horizontal and vertical transformations
automatic transformations
[“A Taxonomy of Model Transformation” Mens & Van Gorp 2006]

• what needs to be transformed into what?
  – easy answer: MODELS!
    • representations of data, systems, artifacts
    • programs
  – model to model transformation
  – model to text (or model to code)

• number of source and target models
• technical space
• endogenous and exogenous transformations
• horizontal and vertical transformations
automatic transformations
[“A Taxonomy of Model Transformation” Mens & Van Gorp 2006]

• what needs to be transformed into what?
• number of source and target models
  – multiple source models and/or multiple target models
  – model merging: combine or merge multiple source models that have been developed in parallel into one resulting target model

• technical space
• endogenous and exogenous transformations
• horizontal and vertical transformations
automatic transformations
[“A Taxonomy of Model Transformation” Mens & Van Gorp 2006]

- what needs to be transformed into what?
- number of source and target models
- technical space
  - is a model management framework containing concepts, tools, mechanisms, techniques, languages and formalisms associated to a particular technology
    - e.g. the XML technical space: XML, XML Schema, XSTL
- endogenous and exogenous transformations
- horizontal and vertical transformations
automatic transformations

["A Taxonomy of Model Transformation" Mens & Van Gorp 2006]

- what needs to be transformed into what?
- number of source and target models
- technical space
  - is a model management framework containing concepts, tools, mechanisms, techniques, languages, and formalisms associated to a particular technology
  - e.g., the XML technical space: XML, XML Schema, XSLT
- endogenous and exogenous transformations
- horizontal and vertical transformations
automatic transformations
[“A Taxonomy of Model Transformation” Mens & Van Gorp 2006]

- what needs to be transformed into what?
- number of source and target models
- technical space

- endogenous and exogenous transformations
  - **endogenous**: transformations between models expressed in the same language
    - optimization, refactoring, normalization
  - **exogenous**: transformations between models expressed using different languages
    - synthesis, reverse engineering, migration

- horizontal and vertical transformations
• endogenous and exogenous transformations
  
  - **endogenous**: transformations between models expressed in the same language
    - optimization, refactoring, normalization
  
  - **exogenous**: transformations between models expressed using different languages
    - synthesis, reverse engineering, migration

• horizontal and vertical transformations

---

**automatic transformations**

[“A Taxonomy of Model Transformation” Mens & Van Gorp 2006]
automatic transformations

[“A Taxonomy of Model Transformation” Mens & Van Gorp 2006]

• what needs to be transformed into what?
• number of source and target models
• technical space
• endogenous and exogenous transformations
• horizontal and vertical transformations
  – **horizontal**: same abstraction level
    • O.O. → ER
  – **vertical**: different abstraction levels
    • O.O. → Java
    • ER → SQL
automatic transformations

[“A Taxonomy of Model Transformation” Mens & Van Gorp 2006]

- what needs to be transformed into what?
- number of source and target models
- technical space
- endogenous and exogenous transformations
- horizontal and vertical transformations
  - horizontal: same abstraction level
    - O.O. → ER
  - vertical: different abstraction levels
    - O.O. → Java
    - ER → SQL
recap!!

Diagram:
- Source Metamodel
- Transformation Language
- Target Metamodle
- Transformation Rules
- Transformation Engine
- Source Model
- Target Model
- MMM

Relationships:
- conformsTo
- from
- to
- exec
- source
- target
- MMM

The diagram illustrates the metamodeling and transformation process, where the source model is transformed to the target model through a transformation engine and rules, constrained by the source and target metamodels.
**recap!!**

**M2M**

```
Source Metamodel -- conformsto --> MMM
          |                         | conformedTo
          v                         v
        Transformation Language
          |                         | conformedTo
          v                         v
    Transformation Rules
          |                         | conformedTo
          v                         v
Source Model -- from => Transformation Engine
          |                         | exec
          v                         v
Target Model -- to => Target Metamodel
```
MDE in practice

- **Model Driven Architecture (MDA)**
  - MDA is an effort defined by OMG around industrial standards, such as MOF, UML, QVT, etc.

- **Eclipse Modeling Framework (EMF)**
  - EMF project is a modeling framework and code generation facility for building tools and other applications based on a structured data model.

- **Software Factories Initiative (SFI)**
  - A Software Factory is a software product line that configures extensible development tools DSLs for building specific kinds of applications.
MDE in practice

- what does actually exist in practice behind these “interesting” definitions?

<table>
<thead>
<tr>
<th></th>
<th>Metamodeling</th>
<th>M2M</th>
<th>M2T</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDA</td>
<td>MOF/UML</td>
<td>QVT</td>
<td>Model-to-text RFP</td>
</tr>
<tr>
<td>EMF</td>
<td>Ecore</td>
<td>ATL, QVTc, QVTo, Epsilon</td>
<td>JET, Acceleo, Xpand</td>
</tr>
<tr>
<td>MS-SF</td>
<td>DSL Tools</td>
<td>X</td>
<td>Text Templates</td>
</tr>
</tbody>
</table>
MDE in practice

- what does actually exist in practice behind these “interesting” definitions?

<table>
<thead>
<tr>
<th></th>
<th>Metamodeling</th>
<th>M2M</th>
<th>M2T</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDA</td>
<td>MOF/UML</td>
<td>QVT</td>
<td>Model-to-text RFP</td>
</tr>
<tr>
<td>EMF</td>
<td>Ecore</td>
<td>ATL, QVTo, Epsilon</td>
<td>JET, Acceleo, Xpand</td>
</tr>
<tr>
<td>MS-SF</td>
<td>DSL Tools</td>
<td>X</td>
<td>Text Templates</td>
</tr>
</tbody>
</table>
some of the contents of these slides are based on the presentations:

- “An Introduction to MDE” by A. Pierantonio
- “Introduction to Model-Driven Engineering” by J.M. Jézéquel
- “Model-Driven Development in The Embedded Environment with OMG Standards” by B. Selic
- “On the Basic Principles of Model Engineering” by J.Bézivin


