MODEL-DRIVEN DEVELOPMENT: PRINCIPLES, PRACTICES AND PROMISES

Mehrdad Saadatmand
XDIN AB, Stockholm
mehrdad.saadatmand@xdin.com
What we are NOT talking about!
In the simplest terms, a model is a (partial) representation of a system.

Features:
- Mapping: a model is based on an original
- Reduction: reflects parts of the original
- Pragmatic: should be usable in place of an original

What we see is actually a model of the real world.

* Model-Driven Software Engineering In Practice; Brambilla, Cabot, Wimmer; 2012.
Use of models in other engineering fields; e.g., in civil engineering

Traditional role of models in software engineering
Complexity of (embedded) systems today:
- Growing demands and expectations on the services
- Growing complexity of the services provided (e.g., due to interaction with environment)
- Complexity of the infrastructure and platforms

In a typical automobile, computer systems are not only taking control of the core functionality, but also create potentials for new features
- e.g., Mercedes-Benz *Magic Body Control*
To cope with the ever-increasing complexity, there is a need to apply appropriate methods and tools

Model-Driven Development
  - Increase the abstraction level
  - Enable analysis at earlier phases of development
  - Code generation
Business Objects (employee, account, customer)

Components

Libraries (GUI, lists)

High-level languages & Procedural constructs

Assembler Languages (mnemonics)

Assembly & Binary values (0 & 1)

Closer to the problem domain

Abstraction

* Testing Commercial-off-the-Shelf Components and Systems; Sami Beydeda, Gruhn Volker.
Model-Driven Software Engineering In Practice; Brambilla, Cabot, Wimmer; 2012.

- Static Analysis
- Rapid Prototyping
- Code Generation
- Automated Testing
- Documentation
- Refactoring / Transformation
MDA, MDD, MDE and MBE:

* Model-Driven Software Engineering In Practice; Brambilla, Cabot, Wimmer; 2012.
MODELING LANGUAGES

- How to create a model?
  - modeling language

- 3 components of a modeling language
  - Abstract Syntax: language concepts and their combinations
  - Concrete Syntax: representation of language concepts
    - Textual
    - Graphical
  - Semantics: meaning
What is the problem here?
Modeling language

- Domain Specific Modeling Language (DSML/DSL)
- General Purpose Modeling Language (GPML/GPL)
A model conforms to a metamodel

A metamodel models a language

Metamodel represents the abstract syntax of the modeling language

Metamodel defines all valid models of a language (e.g., Java grammar)

Similarly a meta model conforms to a meta-meta model
<table>
<thead>
<tr>
<th>Language Engineering</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>M3</strong> Meta-Metamodel</td>
<td>MOF, Ecore</td>
</tr>
<tr>
<td><strong>M2</strong> Metamodel</td>
<td>UML, ER, …</td>
</tr>
<tr>
<td><strong>M1</strong> Model</td>
<td>UniSystem, …</td>
</tr>
<tr>
<td><strong>M0</strong> Model Instance</td>
<td>A UniSystem Snapshot</td>
</tr>
</tbody>
</table>

*Model-Driven Software Engineering In Practice; Brambilla, Cabot, Wimmer; 2012.*
Model Transformation

- Model transformation is the main operation in MDD
- Models are manipulated using transformations
- Using transformations, a mapping between a source and target can be established
- “The automatic manipulation of models, that conforms to a specification and has a specific intent” [Syriani et al. 2013]
- Model-to-Model (M2M) Transformation
- Model-to-Text (M2T) Transformation
  - Code generation
  - Documentation
  - ...
- Traces
- Bidirectional transformation
High-Order Transformations (HOT) : input and output of the transformation are transformations → “Everything is a Model!”
Model-Driven Software Engineering In Practice; Brambilla, Cabot, Wimmer; 2012.

Revisited

Static Analysis

Rapid Prototyping

Code Generation

Automated Testing

Documentation

Refactoring / Transformation
Object Management Group (OMG)’s suggested way of applying MDD techniques

Based on other OMG specifications and products: UML, MOF, OMG UML Profiles, QVT

Based on the well-known idea of Separation of Concerns
MDA

- Specify a system (independent of its platform)
- Specify platforms
- Transform the system specification and solution logic into a technical solution realized on a particular platform
Three types of models:
- Computation Independent Model (CIM)
- Platform Independent Model (PIM)
- Platform Specific Model (PSM)
Unified Modeling Language: a set of notations/suite of languages
Extension mechanisms:
- Constraints
- Tagged Values
- Stereotypes

UML Profile: collection of extensions for a specific domain (e.g., system modeling, automotive domain, real-time embedded systems)
- SysML: System Modeling Language
- UML profile for MARTE: Modeling and Analysis of Real-Time and Embedded Systems
- EAST-ADL: Electronics Architecture and Software Technology - Architecture Description Language (Automotive domain)
MARTE
APPLICATIONS

NFR Profile
Analysis using model transformation

* Toward Model-Based Trade-off Analysis of Non-Functional Requirements; Mehrdad Saadatmand, Antonio Cicchetti, Mikael Sjödin; EuroMicro SEAA; 2012.
Secure Component Model

EVERYTHING IS A MODEL!
RESOURCES

- XDIN (www.xdin.com)
- Model-Driven Software Engineering in Practice; by Marco Brambilla, Jordi Cabot, Manuel Wimmer; 2012
- Model-Based Engineering of Embedded Systems (MBEES) research group (http://www.es.mdh.se/organization)