Model Driven Architecture

All you need are models

Anneke Kleppe, Klasse Objecten
Contents

• Limited Vision on MDA
• Modeling Maturity Levels
• Models
• Model Driven Development
• Model Driven Architecture
MDA in a Nutshell

- Automatic transformation from Platform Independent Model (PIM) to Platform Specific Models (PSM).
  - PIM usually written in UML.
  - PSM can be diverse:
    - Java, J2EE, SQL, C++, .NET, COBOL, C#, CORBA, XML, etc. etc.
    - Bridges between PSMs generated as well.
- Automatic transformation from PSM to Code
MDA Overview

Platform Independent Model

PSM SQL

SQL Code

← SQL - EJB →

PSM EJB

EJB Code

← EJB - JSP →

PSM JSP

JSP Code

Copyright Klasse Objecten
Why Model Driven?

• Productivity?
• Portability?
• Interoperability?
• Maintenance and documentation?
• Raising the level of abstraction!
  – Handling more complex systems
Part 0: Modeling Maturity Levels
Modeling Maturity Levels

MML 0: No specification
MML 1: Textual Specification
MML 2: Text with Diagrams
MML 3: Diagrams with Text
MML 4: Precise Models
MML 5: Models only
MML 0: No Specification

• Specification of software is kept in the heads of the developers.

My software …
MML 1: Textual

• Specification of software is written down in one or more natural language documents.
MML 2: Text with Diagrams

- Specification of software in one or more natural language documents…
- … plus several high-level diagrams to explain the overall architecture.
MML 3: Diagrams with Text

• Specification of software is written down in one or more models.

• Additional natural language text is used to explain the background and motivation of the models.
MML 4: Precise Models

- Specification of software is written down in one or more models.
  - Natural language text is used only to explain the background and motivation of the models.

- The models are precise enough to be directly transformed to actual code.

- This level models necessary for OMG’s Model Driven Architecture.
MML 5: Models Only

• The models are precise and detailed enough to allow complete code-generation.
• The code is invisible (as assembler is today).
• Modeling language → High level programming language.
• This is future technology (☹).
UML/OCL in the MMLs

MML 0: No use of UML
MML 1: No use of UML
MML 2: Moderate use of UML
MML 3: Extensive use of UML, light use of OCL
MML 4: Extensive use of UML, extensive use of OCL
MML 5: ???
Part 1: *Models*
Model and System

M1: Model

Customer
- title: String
- name: String

Order
- number: String

M0: System

Customer
- title = “Dr.”
- name = “nobody”

Customer
- title = “Mr.”
- name = “everyman”

Order
- Number = “13A”
Model and Meta-model

M1: Model

- **UML Class**: name = "Customer"
- **UML Class**: name = "Order"
- **UML Attribute**: name = "number"

M2: Model of a Model

- **UML Class**: name : String
- **UML Attribute**: name : String

<<instance of>> <<instance of>> <<instance of>>
Metamodel and Meta-metamodel

M3: Model of a Model of a Model

MOF Class
name: String

<<instance of>>

M2: Model of a Model

MOF Class
name = “UML Class”

<<instance of>>

MOF Class
name = “UML Attribute”
Meta-modeling Overview

M3: Model of a Model of a Model

M2: Model of a Model

M1: Model

M0: System

MOF

UML Metamodel

UML Model
MDA

• **MDA is defined around:**
  – Models
  – Model Transformations

• **Important model types:**
  – PIM : Platform Independent Model
  – PSM : Platform Specific Model
  – Code
"do the right thing: conform to your metamodel"
Part 2: Model *Driven* Software Development
MDA Transformations
MDA Framework
MDA

• To enable MDA we need
  – Modeling languages
  – Transformation definitions
  – Tools
MDA Framework

Language is written in Source Model by Transformation definition tool and is used by Target Model.
Defining Languages

- Metamodel
- Model
- Language
- Meta-language

Is written in
Is defined by
Is written in
MDA Framework

- **Meta language**
  - Is written in
  - Is written in

- **Language**
  - Is written in

- **Source Model**
  - Is written in

- **Transformation definition**
  - Is used by

- **Transformation tool**
  - Is used by

- **Target Model**
  - Is written in

Copyright Klasse Objecten
Defining Transformations

• A transformation definition maps
  – Element(s) from the source language
  – to
  – Element(s) from the target language

• based on the respective language definitions
Tools

• What type of tools are there

Model Editor  Model Validator  Model Repository  Transform Def.Editor  Transform Executor

Part 3. Model Driven Architecture
Where is the Architecture?

• MDA defines architecture of software development environment
  – Interoperability of tools
Filling the MDA Framework

• **Modeling Languages**
  – OMG: UML, OCL, CWM, CORBA, EDOC
  – Non-OMG: Java, C#, SQL, Petri-nets, DFD, etc.
  – UML Profiles

• **Meta-modeling Languages**
  – MOF is the OMGs meta-modeling language
  – BNF is a meta-language for defining textual languages

• **Transformation Definition Languages**
  – OMG: MOF QVT
  – Many scripting languages
Current Status of MDA

- **Modeling Language: UML**
  - Static description is ok
  - Dynamic specification is lacking
    - Much manual coding at PSM or Code level

- **Transformations**
  - Scripting languages: don’t scale
  - Template based: works better
  - Declarative: upcoming OMG-QVT

- **Tools**
  - Support is not optimal, although claims are different
  - ArcStyler, OptimalJ, AndroMDA
Part 4: A Look into the Crystal Ball
Software Development in 2015
So What’s Different?

- Software development at higher level of abstraction!!!
MDA: a Revolution

- 1960-1970: from assembler to functional languages
- 2000-2010: from functional languages to modeling languages
References

• MDA Explained, The Model Driven Architecture: Practice and Promise

• The Object Constraint Language, Getting Your Models Ready for MDA

• OMG website
  – www.omg.org/MDA
Questions

Klasse Objecten,
Soest
tel. 035-6037656
www.klasse.nl
a.kleppe@klasse.nl