Model Driven Engineering

Myths, reality and potential

Jacky Estublier
Laboratoire LSR, Grenoble and AS CNRS,

with slides and ideas from Jean Bezivin, Jean-Marie Favre
The OMG initiative in November 2000

- The concerns
  - No unique technology.
    Platform are and will remain heterogeneous.
  - Too much Complexity.
    Specialist shortage.
  - Technologies are evolving too fast.
    Application are obsolete to fast, porting is too expensive.
  - ..... (nothing new)

- THE solution : MDA (Model Driven Architecture)
  - Separate
    - The business part (PIM : Platform Independent Model)
      Stable, does not need computer specialist ...
    - Implementation on a platform (PSM Platform Specific Model)
  - Transform automatically a PIM into a PSM ....
Myth 1: The Y cycle

- PIMs (Platform Independent Models)
- PSMs (Platform Specific Models)

Merge → The running system
Myth 2 : A unique 3 levels architecture

- 4 -
A few characteristics of models

• A model is an abstraction of a system
  • Built with an intention in mind
  • Can answer questions on the system.
  • Is a set of facts on the system.

• A model represents an aspect of a system
• Each system admit an infinite number of aspects
• A complex system can only be described by a large number of models
• The characteristics of the aspect captured in a model must be clearly defined by a metamodel
The Y revisited
Many aspects (Business and platform)
Many aspects. Aspect Oriented Modeling v.s. AOP

Separation of concerns => composition of concerns

Composition, weaving and correspondances are key issues
Transformations

Weaving, composition

Transformation, generation

PIM

PDM
Transformations

Composition complexity may become a major issue

An intrinsic part of the application

Weaving, composition

Transformation, generation
Transformations

Composition complexity may become a major issue

An intrinsic part of the application

MDE: Is based on model transformation, generation

Generative programming, DSL …
Architecture and methodology

Composition

Weaving

Transformation

Generation
Architecture and methodology

Concerns
(many acteurs, many users)

Composition

Weaving

Transformation

Generation
Architecture and methodology

Concerns
(many acteurs, many users)

Engineery
(Many steps)

J. Estublier
Architecture and methodology

Concerns
(many acteurs, many users)

Engineery
(Many steps)

All the SE issues are found again:
Evolution, Process, Versioning, Validation, Verification, Volume, Complexity, etc.
Architecture and methodology

Concerns
(many acteurs, many users)

Engineery
(Many steps)

Composition
Weaving

Transformation
Generation

All the SE issues are found again:
Evolution, Process, Versioning, Validation, Verification, Volume, Complexity, etc.

MDE : New methods, new tools and new platforms are required
Model Transformation, composition, correspondence.
Model Engineering. MDSE

We do not know what is a platform ....
Model Transformation, composition, correspondence.

Model Engineering. MDSE

We do not know what is a platform ....

Exit : Y, PIM and PSM
The architecture revisited

- $M_0$: "the real world"
- $M_1$: model
- $M_2$: meta-model
- $M_3$: meta-meta-model

- MOF
- UML

Different uses of this model

Instance-of
The current vision: The fundamental relationships

Real World

\( M_0 \)  

\( M_1 \)  

\( M_2 \)  

\( M_3 \)  

Meta-model

Model

Conform_to
The current vision: The fundamental relationships

The model $M_0$ represents the state (model).

The meta-model $M_1$ conforms to the model $M_0$.

The meta-meta-model $M_2$ represents the meta-model $M_1$.

The meta-meta-meta-model $M_3$ conforms to the meta-model $M_2$.

The model $M_0$ represents the real world.
The current vision: The fundamental relationships
The current vision: The fundamental relationships

\[ M_0 \rightarrow M_1 \rightarrow M_2 \rightarrow M_3 \rightarrow \text{State (model)} \]

- \( M_0 \) represents the real world model.
- \( M_1 \) represents the model.
- \( M_2 \) represents the meta-model.
- \( M_3 \) represents the meta-meta-model.

\[ \text{Conform_to} \]

\[ \text{Represents} \]

\[ \text{Real World Model} \]

\[ \text{Model} \]
The current vision: The fundamental relationships

\[ M_0 \text{ State (model)} \rightarrow M_1 \text{ model} \rightarrow M_2 \text{ meta-model} \rightarrow M_3 \text{ meta-meta-model} \]

- Conform_to
- Represents
- Model
- Real World Model
The current vision: The fundamental relationships

- Model Represents Real World Model
- Model Conform_to Meta Model
The current vision: The fundamental relationships

- Model
  - Represents Real World Model
- Meta Model
  - Conform_to Model
- Meta-model
  - ?
- Meta-meta-model
  - ?

J. Estublier
**Fundamental relationships**

We can learn a lot from:
- Programming languages,
- Formal methods,
- Ontologies,
- Generative Programming,
- Domain Specific Languages,
- Validation techniques,
- Etc....

We can learn a lot from:

**Set of Meta Models**

- **Meta Meta Model** represents
- **Set of Meta Models**

**Set of Models (Language)**

- **Meta Model (Language Definition)** represents
- **Set of Models (Language)**

**Model**

- **System** represents

**Image of**

Represents == describe or specify
Image_of == Interpretation.
Conformity

- A meta model is the model of a set of models.
  - A meta model is a Language Specification.
    - A meta model is a DSL (Domain Specific Language) and vice versa.
    - UML, SPEM, QVT, etc. are DSLs
  - Specification en term of classes (Modelware)
    - An instance is conform to its class (conformity = instance_of)
  - Specification in term of a classification (Ontologies)
    - An element pertains to its ontological class (conformity = fits the definition)
  - Specification in term of a grammar (GrammarWare)...
    - A program is conform if it exists a sequence of derivation, ...

- A meta model is NOT a model of a model
  - But a model if the set of valid models.
MOF

The UML meta-Model

A Specific UML Model

Conform to

XML

An XML Schéma

A Specific XML Model

Algèbre Relationnelle

A DB Schéma

A Specific Database

(M3)

(M2)

(M1)
J. Estublier

- 20 -

From J. Bezivin
A recurrent pattern: The technological domains

\[ M^3 \]
- EBNF

\[ M^2 \]
- Pascal Language Grammar
- The UML meta-Model

\[ M^1 \]
- A specific Pascal Program
- A Specific UML Model

\[ (Grammarware) \]
\[ (Modelware) \]
\[ (Docware) \]
\[ (Relational ware) \]

Conform to

From J. Bezivin
• There exist many technological domains:
  • XML, Grammarware, Ontologies, Data Bases ...
  • A technological domain is defined by a family of MM models.
    • MOF, EMF, ...
    • Relational algebras
    • The languages for grammar representation (BNF, EBNF, ...) 

• Each domains has its strong and weak points.
  • There is definitively not a superior technology
  • The different spaces must cooperate
  • Bridges between spaces must be built
Correspondences and transformations

- Transformations Inter technological domains
Software Engineering Process
Software Engineering Process

MOF

XML

UML

XML Schema

meta-meta model

document

meta-model

XML Schema

document
Software Engineering Process
Software Engineering Process

MOF

XML

UML

meta-model

XML Schema

meta-meta

Relational Algebra

document

meta-

model

 Schema

BD

EBNF

Java

meta-model

XML

meta-

model

document

meta-

model

XML Schema

meta-

model
Software Engineering Process

MOF

XML

XML Schema

Relational Algebra

Schema

BD

UML

meta-model

meta-meta

model

meta-document

Java

EBNF

Relational Algebra

Schema

BD

MOF
From Objects ... ... to Models

1990

2000
OMG: standardisation

1990: Corba, the unique and universal middleware
OMG: standardisation

1990: Corba, the unique and universal middleware

2000: UML, the unique and universal modeling language
"MDA as the potential to greatly reduce development time and greatly increase the suitability of application. But does so not by magic, but by providing mechanisms by which developers can capture their knowledge of the domain and implementation technology more directly in a standardized form and by using this knowledge to produce automated tools that eliminate much of the low level work of development.

May 2004
"An MDA Manifesto"

Domain Specific Language

MDA

Direct Representation

Productive Models

Generative Programming

Automation

Standards

XML

MOF

EMF

IBMs Rational Software

MDA Journal

May 2004

Grady Booch
Alan Brown
Sridhar Iyengar
James Rumbaugh
Bran Selic

J. Estublier  J. M. Favre
Une convergence forte

Microsoft

MDA

2001

2001-2003

IBM

2004
So even a business could express in a formal, modeled way, not just scribbling on paper, how the business process is changing over time or how it's different from other companies. **So instead of having lots of code behind that, you just have visual, essentially model, customization.**

You know UML made the meta-models a little complex, so **I don't think UML alone is the answer.** Web services forces you to think modeling. And that's part of the good thing about it. And the promise here is that you write a lot less code, that you have a model of the business process. And you just look at that visually and say here is how I want to customize it.
MDE is not ...

- **MDE <> MDA**
  - MDA = an industrial standard
  - MDE = a federative approach
  - MDE => Multiples technological domains
  - Toward robust MDE fundamentals

- **MDE <> UML**
  - MDE et UML2.0 : Contradictory approaches
  - UML = define a new UML profile
  - MDA = define a new metamodel

- **MDE <> OO**
  - Everything is a model <> Everything is an object
  - A model is not an instance (OO) of a metamodel
  - OO : instance_of, inherits; MDE : represents, conform_to
  - MDE => Multiple technological domains
MDE is neither ...

- **MDE <> Modelisation method** (e.g. Meurise, Sadt, ...)
  - Modelisation: analysis and design
  - MDE: implementation, deployment, execution, evolution, ...
  - Contemplative Models (modelisation) vs. Productive Models (MDE)
  - Modelisation: models as 1st order elements
  - MDE: metamodels and transformations as 1st order elements

- **MDE <> AOP, MDE = AOM**
  - Aspect Oriented Programming: Too late in the life cycle
  - Aspect Oriented Modeling: natural solution
  - Model Weaving

- **MDE <> "traditional" Compilation**
  - Extensible Languages, Modular Languages, Languages weaving
  - User defined transformations.
  - Transformation evolution, etc.
Conclusion

• MDE do not propose anything radically different
• MDE seeks to integrate the good ideas developed elsewhere
  • Separation and composition of concerns
  • Generalized Transformation and composition Techniques
• Many fundamental open research issues
• MDE requires a long term and collaborative effort from various scientific communities.
• Consequences on Software Engineering and the Software industry will be deep and long lasting.