Model Transformations

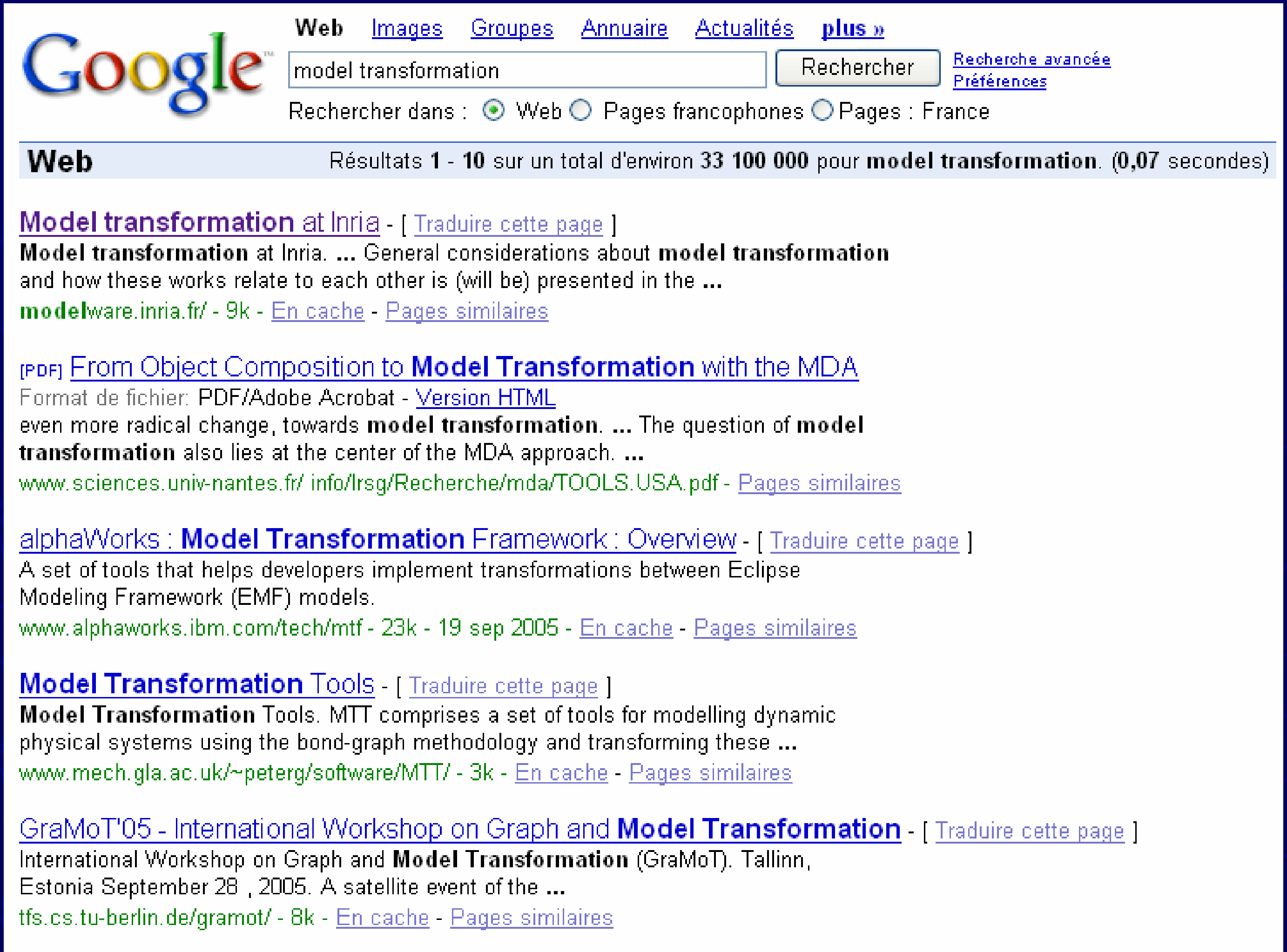
An overview

Pierre-Alain Muller

INRIA

pierre-alain.muller@irisa.fr

Model Transformations @ Google



model transformation

Rechercher dans : ● Web ○ Pages francophones ○ Pages : France

Web Résultats **1** - **10** sur un total d'environ **33 100 000** pour **model transformation**. (**0,07** secondes)

Model transformation at Inria - [Traduire cette page]
Model transformation at Inria. **...** General considerations about **model transformation** and how these works relate to each other is (will be) presented in the **...**
modelware.inria.fr/ - 9k - En cache - Pages similaires

[PDF] From Object Composition to **Model Transformation** with the MDA
Format de fichier: PDF/Adobe Acrobat - Version HTML
even more radical change, towards **model transformation**. **...** The question of **model transformation** also lies at the center of the MDA approach. **...**
www.sciences.univ-nantes.fr/ info/lrsg/Recherche/mda/TOOLS.USA.pdf - Pages similaires

alphaWorks : **Model Transformation** Framework : Overview - [Traduire cette page]
A set of tools that helps developers implement transformations between Eclipse Modeling Framework (EMF) models.
www.alphaworks.ibm.com/tech/mtf - 23k - 19 sep 2005 - En cache - Pages similaires

Model Transformation Tools - [Traduire cette page]
Model Transformation Tools. MTT comprises a set of tools for modelling dynamic physical systems using the bond-graph methodology and transforming these **...**
www.mech.gla.ac.uk/~peterg/software/MTT/ - 3k - En cache - Pages similaires

GraMoT'05 - International Workshop on Graph and **Model Transformation** - [Traduire cette page]
International Workshop on Graph and **Model Transformation** (GraMoT). Tallinn, Estonia September 28 , 2005. A satellite event of the **...**
tfs.cs.tu-berlin.de/gramot/ - 8k - En cache - Pages similaires

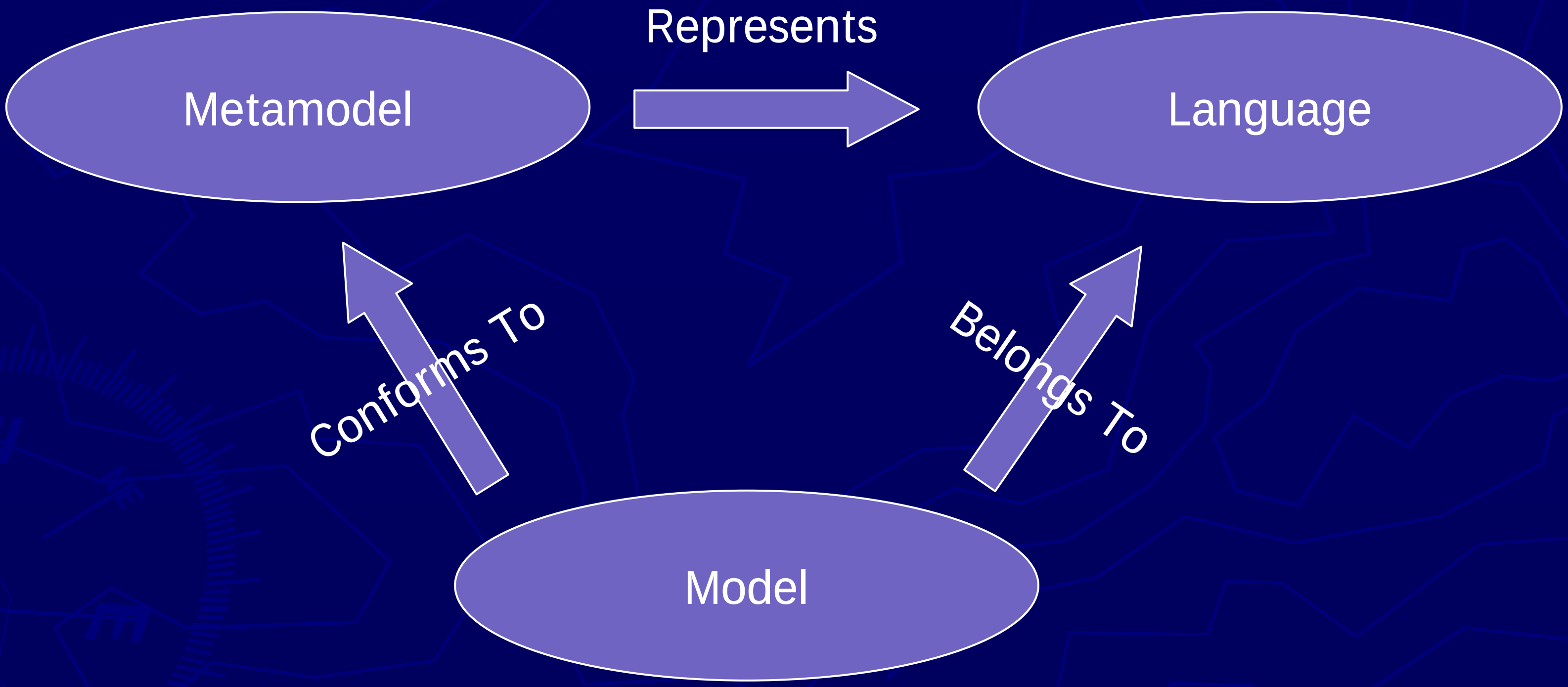
Outline

- MDE basic principles
- What is a model transformation?
- Typology of model transformations
- Examples of transformations

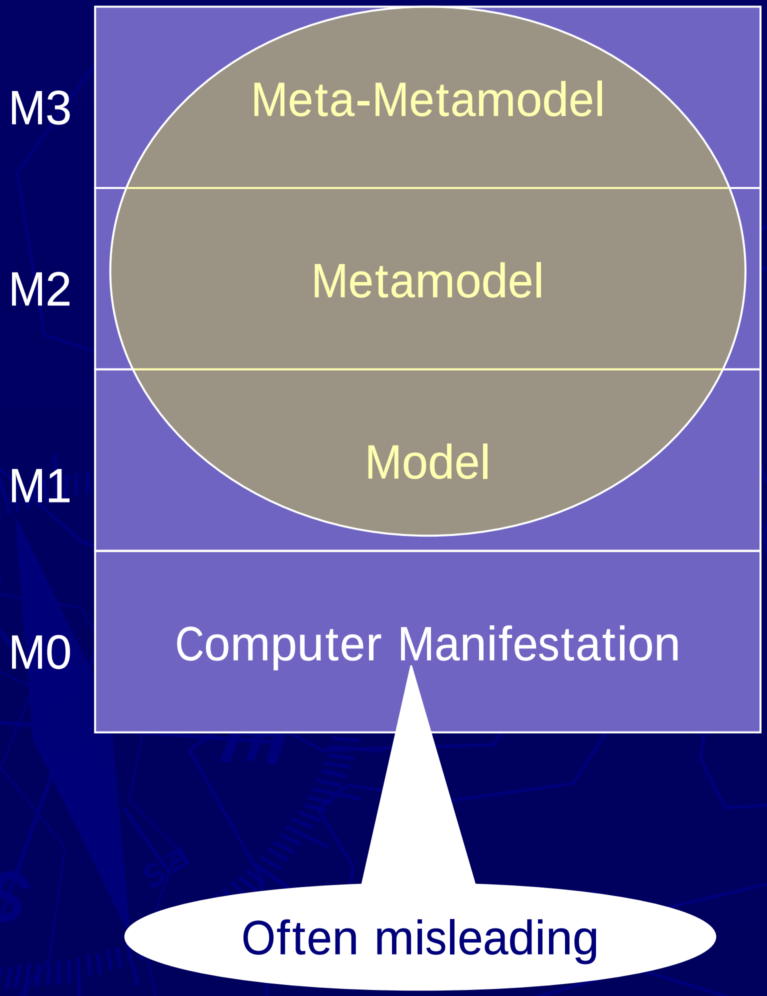
Principles of Model-Driven Engineering

- A kind of (software) development approach
- Models as first class entities
- Everything is a model
- A model conforms to an other model (meta-model)
- A model transformation takes models and produces models
- A model transformation is a model

Metamodels and models



Meta-modeling architecture



M3

Meta-Metamodel

M2

Metamodel

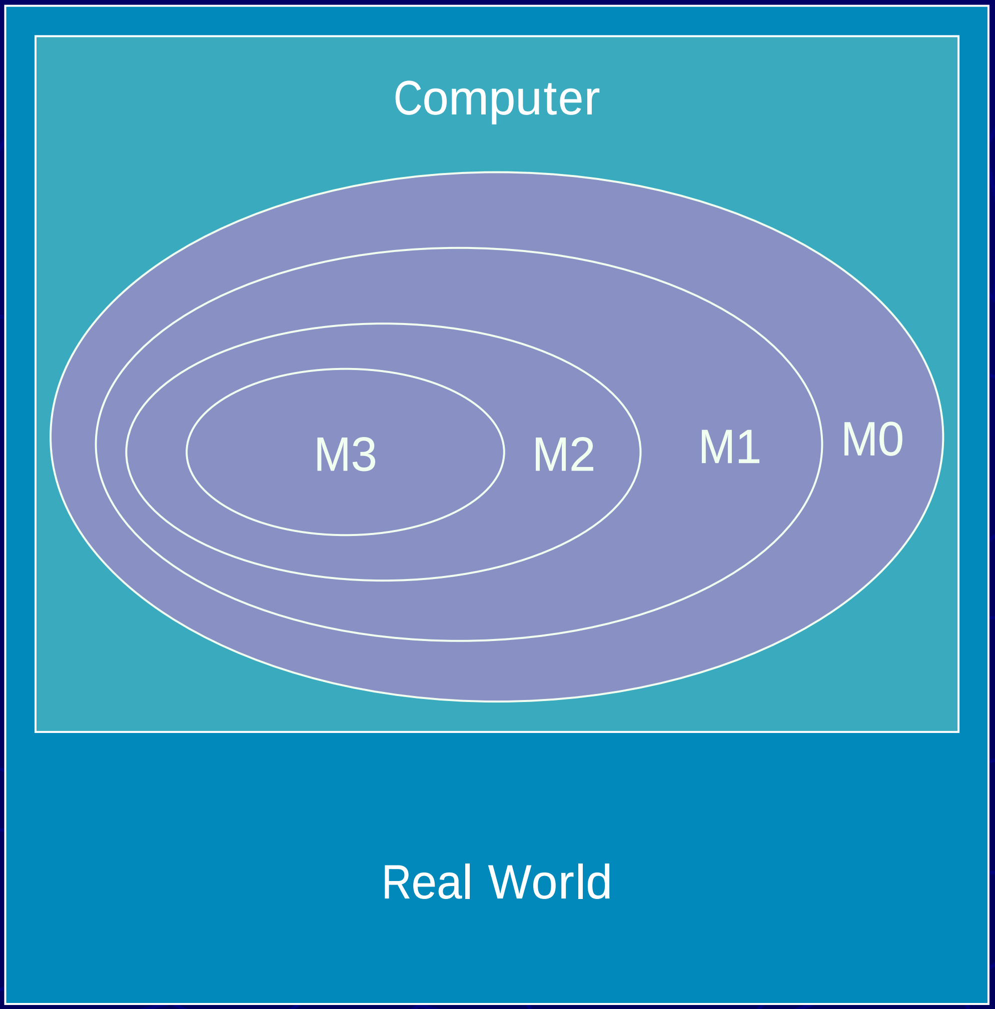
M1

Model

M0

Computer Manifestation

Often misleading



Computer

M3

M2

M1

M0

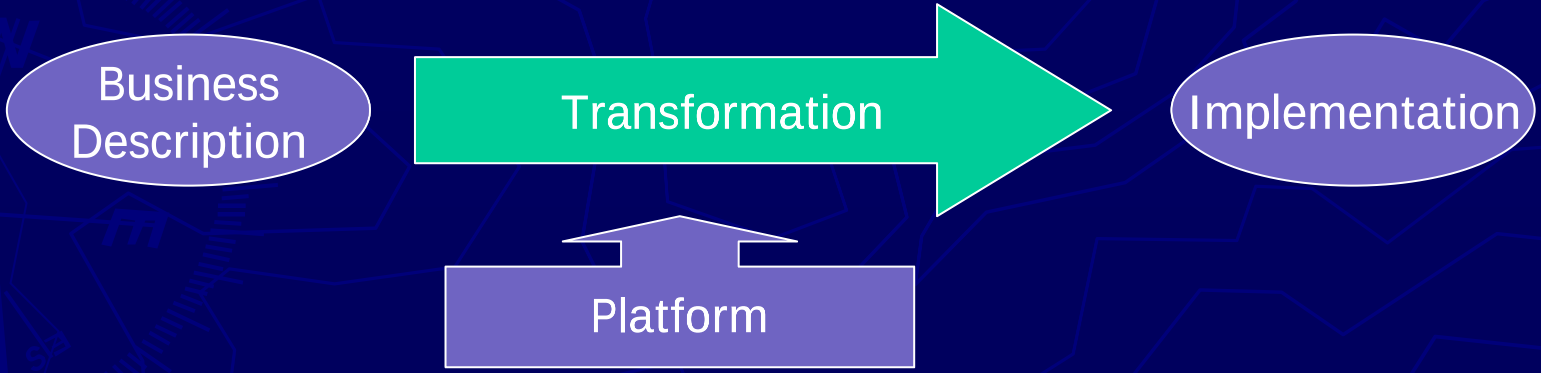
Real World

MDA = MDE à la OMG

- OMG, Object Management Group
- MDA, Model-Driven Architecture
- PIM, Platform Independent Model
- PSM, Platform Specific Model
- (PDM, Platform Description Model)
- Transformation (PIM, PDM) -> PSM
  - RFP MOF Q/V/T Query, Views, Transformations
  - RFP MOF to Text

Motivation

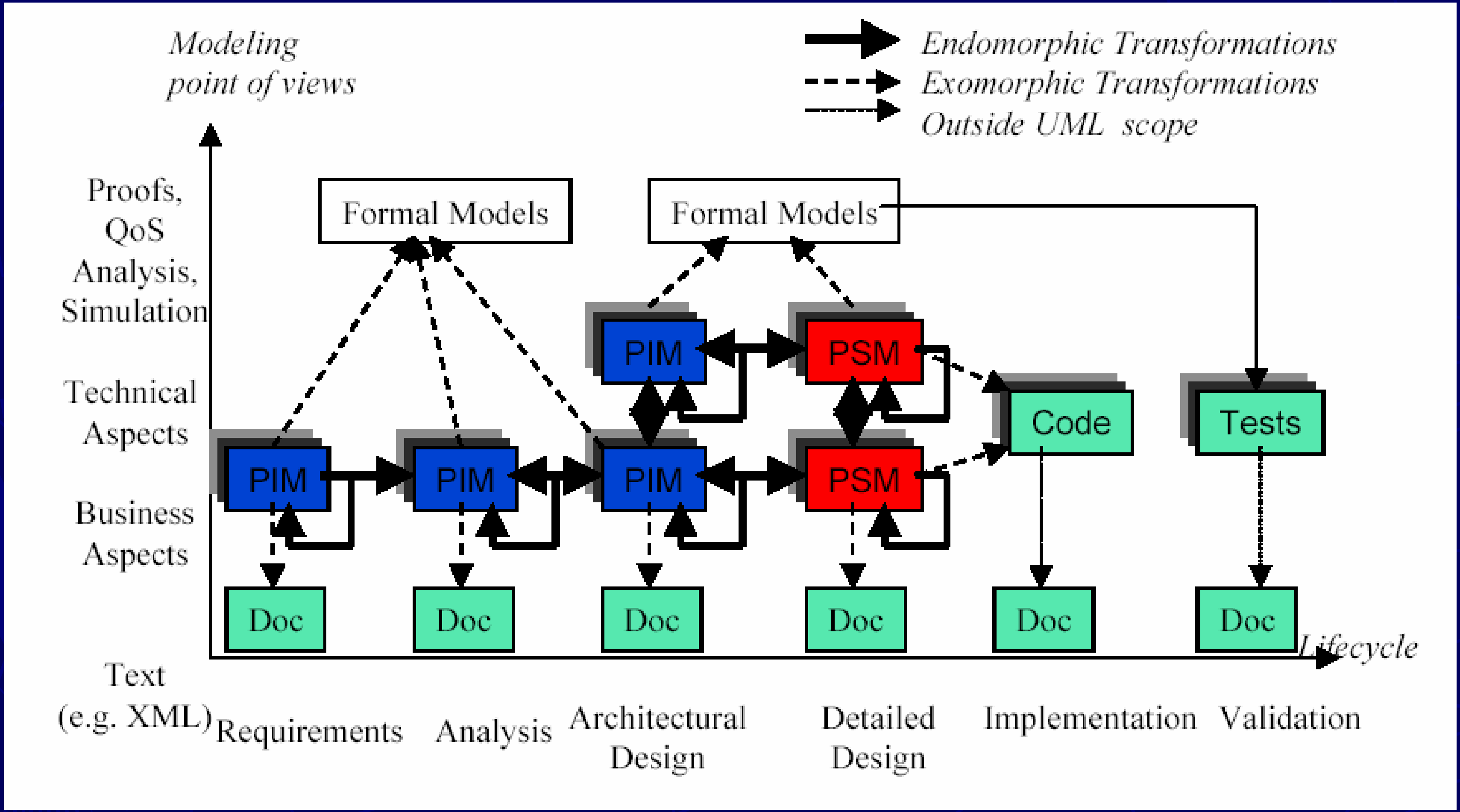
- Model transformation is key to Model-Driven Engineering
  - Automation of the transition from Business models to Implementation models



But also

- Refining models
- Reverse engineering (code to models)
- Generating new views
- Applying design patterns
- Refactoring models

Typical scope for transformations



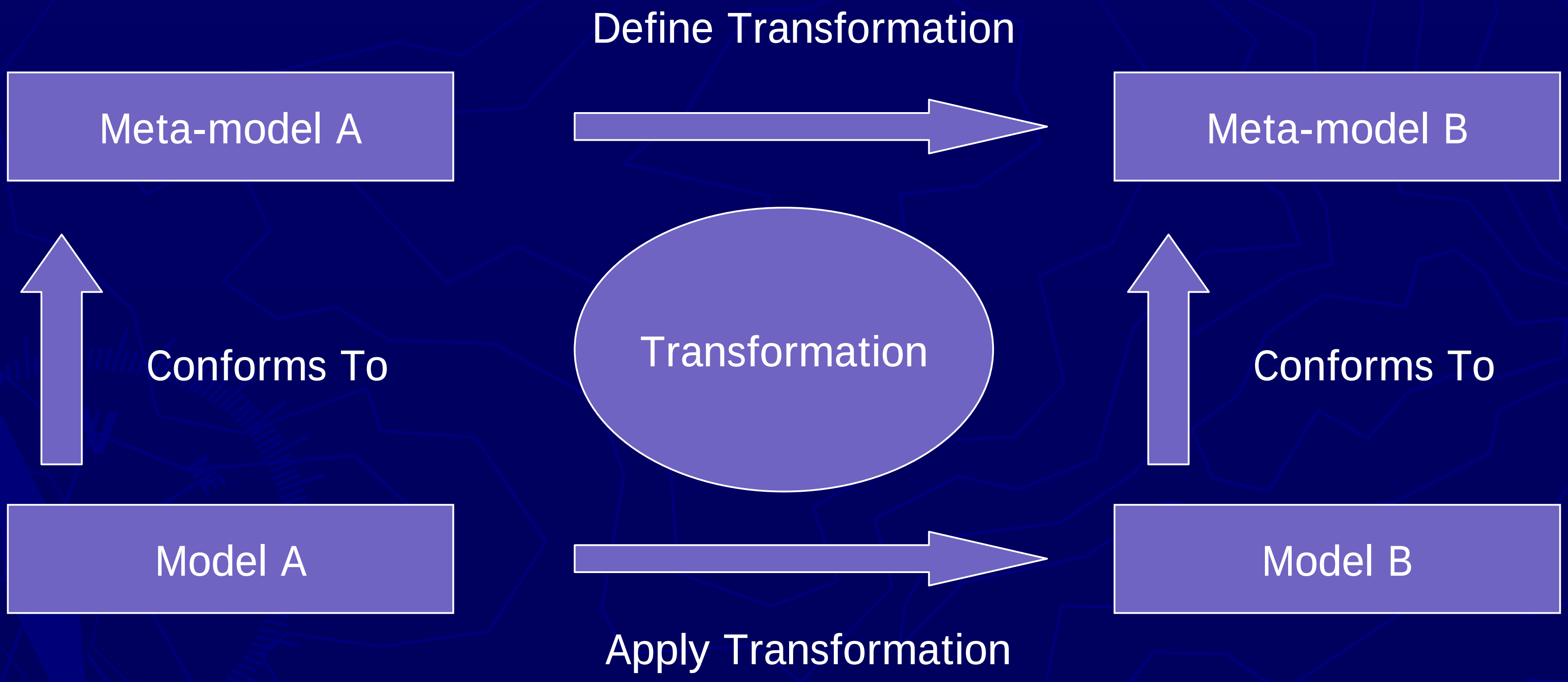
Related fields

- Program transformation and compiler techniques
- Meta-programming techniques
- Graph rewriting techniques

MOF 2.0 Queries/Views/Transformations RFP

- Define a language for querying MOF models
- Define a language for transformation definitions
- Allow for the creation of views of a model
- Ensure that the transformation language is declarative and expresses complete transformations
- Ensure that incremental changes to source models can be immediately propagated to the target models
- Express all new languages as MOF models

Transformation Architecture



Define Transformation

Meta-model A

Meta-model B

Conforms To

Transformation

Conforms To

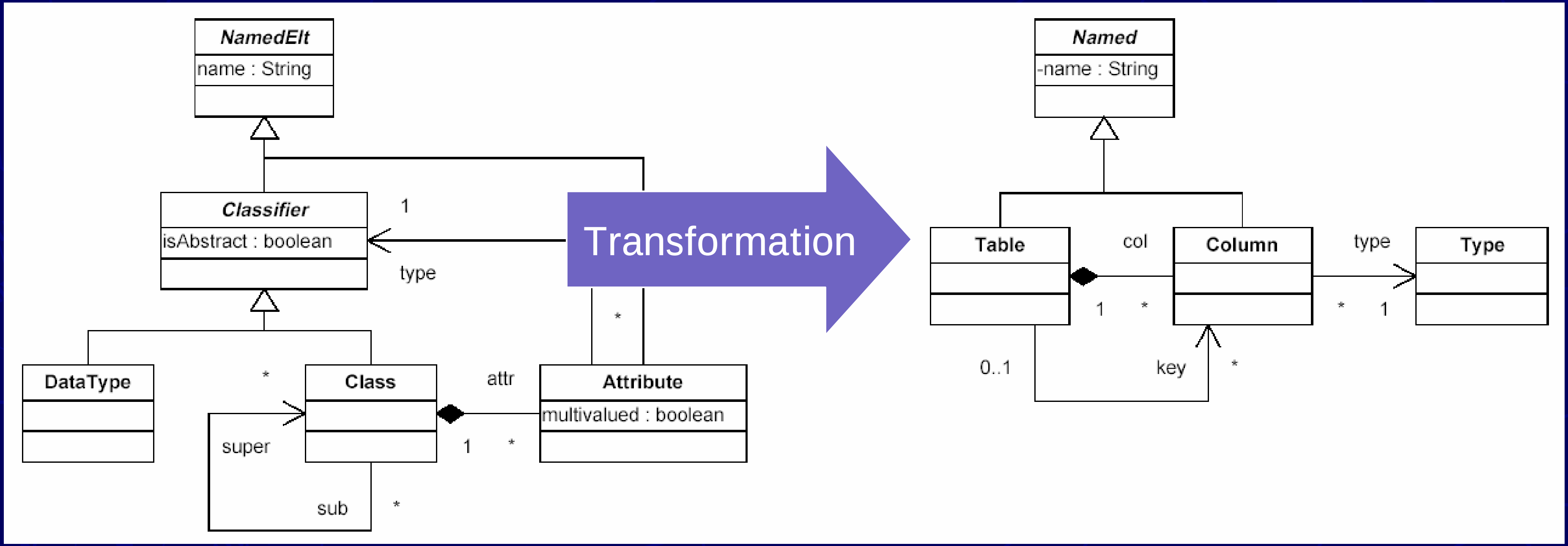
Model A

Model B

Apply Transformation

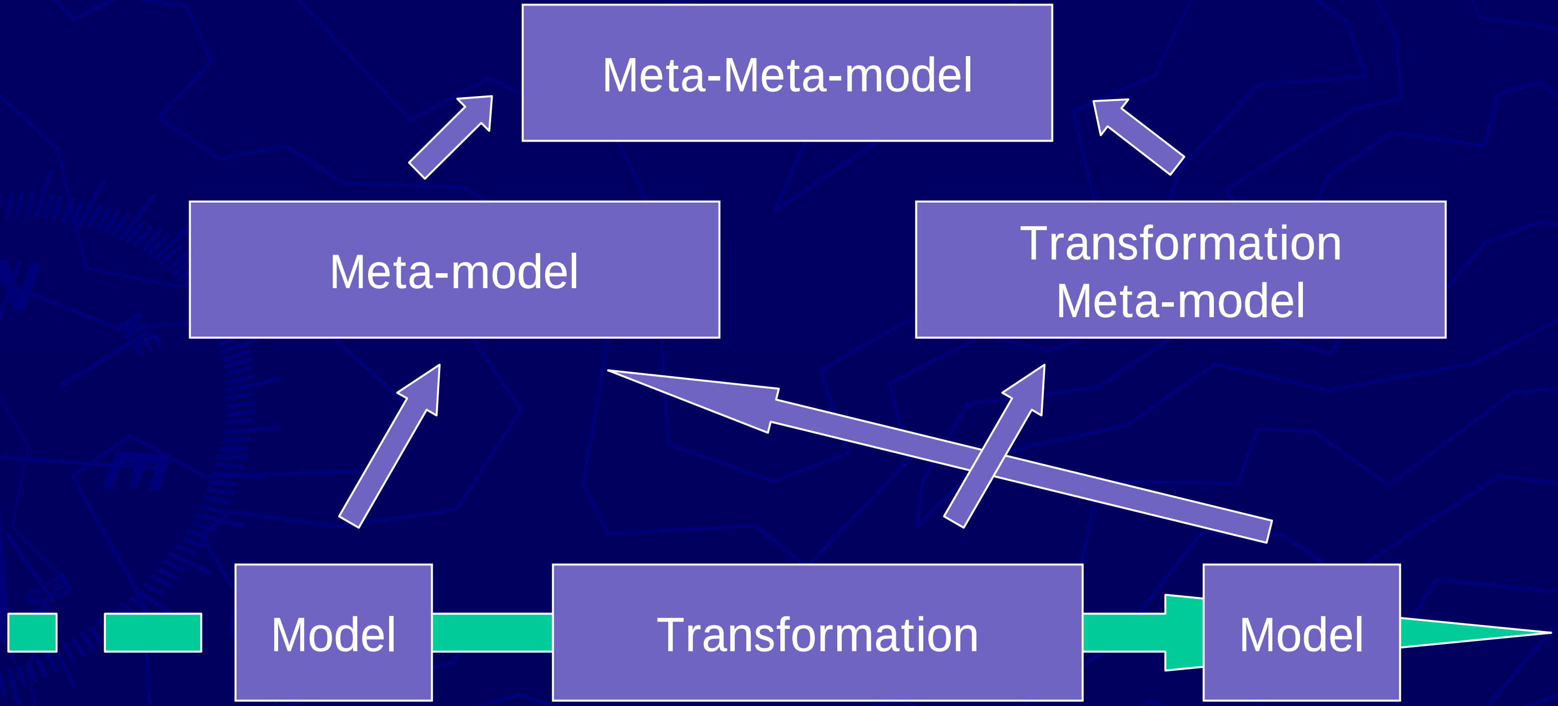
Typical Example

From UML to RDBMS



Transformations as models

- Composition of transformations
- Transformation of transformations



Toward Model-Transformations

- CRUD on model elements
  - Create, Read, Update, Delete
- Transformation rules written in
  - General purpose languages + API
  - Intermediate transformation language
  - Dedicated Model-Transformation languages

General purpose language approach

- Java, VB, C++, C#,...   Your favorite language!
- Currently available in the tools via APIs

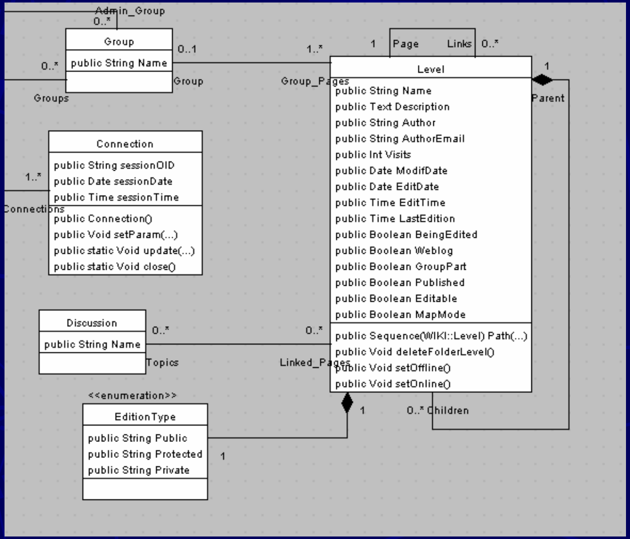
- No overhead to learn a new language
- Tool support to write the transformations

=> Monsieur Jourdain’s approach

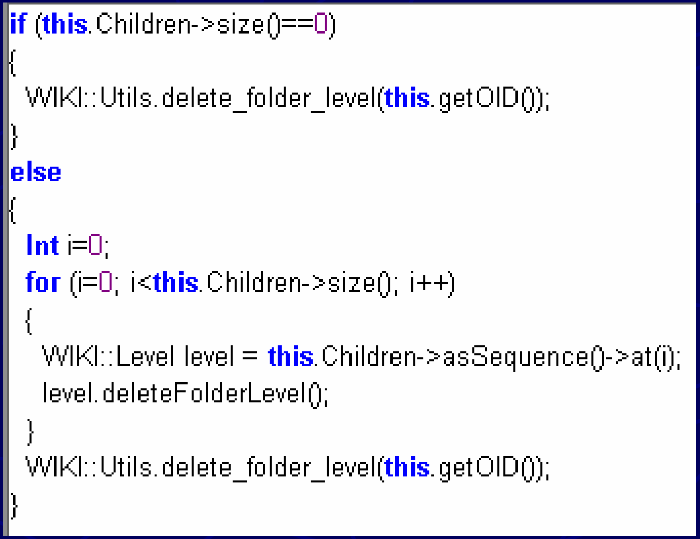
- It’s going to be challenging to do better!

Action Language

- Use a general purpose action language
  - Better navigation facility (associations)
  - Get access to the types defined in the models
  - Procedural rule description



Admin_Group
0..*
Group
public String Name
0..1
Group
0..*
Groups
1..*
Group_Pages
1
Page
Links
0..*
Level
public String Name
public Text Description
public String Author
public String AuthorEmail
public Int Visits
public Date ModifDate
public Date EditDate
public Time EditTime
public Time LastEdition
public Boolean BeingEdited
public Boolean Weblog
public Boolean GroupPart
public Boolean Published
public Boolean Editable
public Boolean MapMode
public Sequence(WIKI::Level) Path(...)
public Void deleteFolderLevel()
public Void setOffline()
public Void setOnline()
1
Parent
0..* Children
Connection
public String sessionOID
public Date sessionDate
public Time sessionTime
public Connection()
public Void setParam(...)
public static Void update(...)
public static Void close()
1..*
Connections
Discussion
public String Name
0..*
Topics
0..*
Linked_Pages
<<enumeration>>
EditionType
public String Public
public String Protected
public String Private
1
1

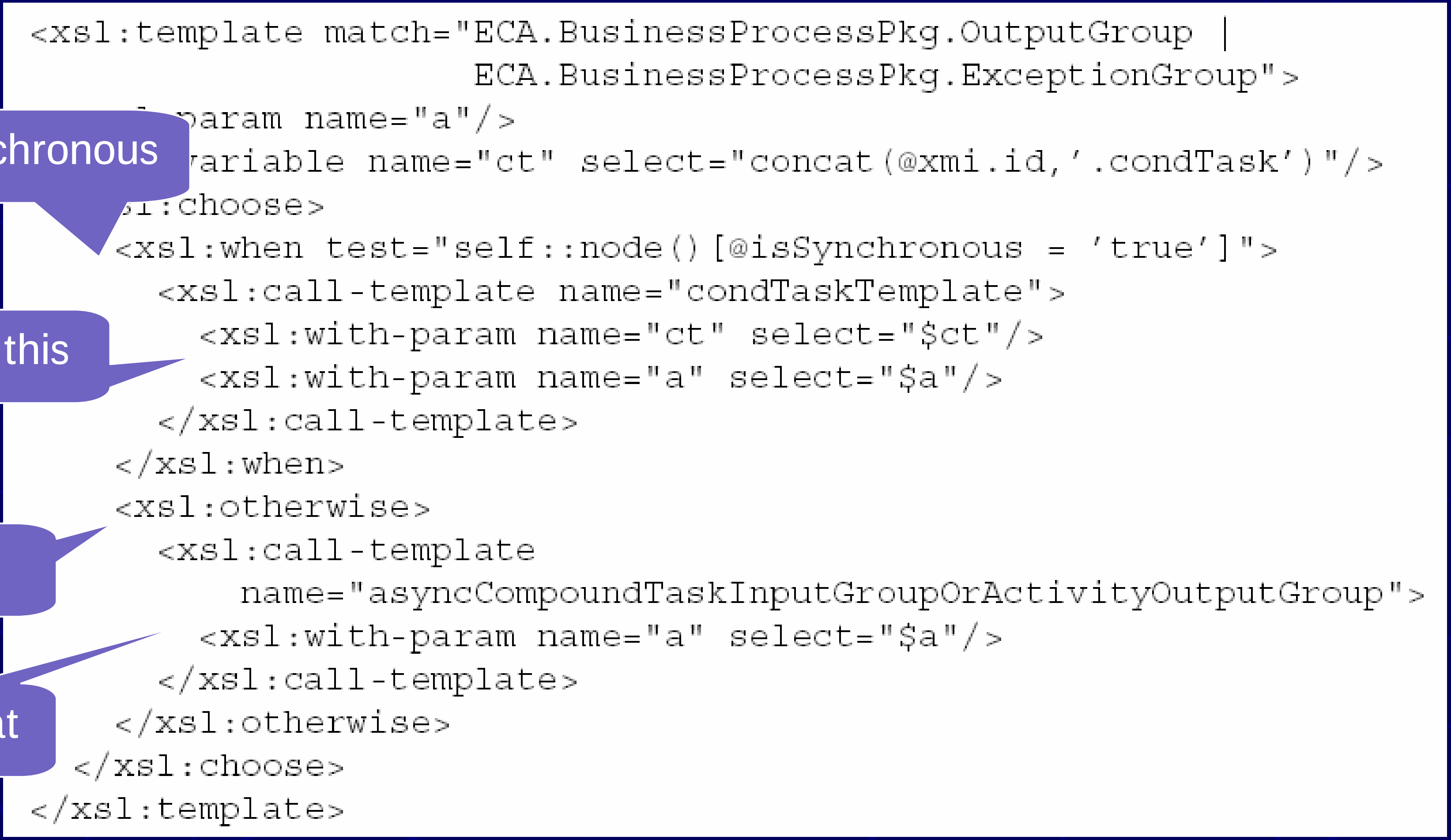


```
if (this.Children->size()==0)
{
  WIKI::Utils.delete_folder_level(this.getOID());
}
else
{
  Int i=0;
  for (i=0; i<this.Children->size(); i++)
  {
    WIKI::Level level = this.Children->asSequence()->at(i);
    level.deleteFolderLevel();
  }
  WIKI::Utils.delete_folder_level(this.getOID());
}
```

Intermediate transformation language

- Typically XML based
  - But XML (XMI) is verbose
- XSLT can be used to transform XML trees into other (XML) (trees)
  - More batch than interactive
  - Parameters are passed by values
  - XSLT transformations are not really easy to maintain
- Better for simple transformations

Example of XSLT transformation



```
<xsl:template match="ECA.BusinessProcessPkg.OutputGroup |
                     ECA.BusinessProcessPkg.ExceptionGroup">
  <xsl:param name="a"/>
  <xsl:variable name="ct" select="concat(@xmi.id,'.condTask')"/>
  <xsl:choose>
    <xsl:when test="self::node()[@isSynchronous = 'true']">
      <xsl:call-template name="condTaskTemplate">
        <xsl:with-param name="ct" select="$ct"/>
        <xsl:with-param name="a" select="$a"/>
      </xsl:call-template>
    </xsl:when>
    <xsl:otherwise>
      <xsl:call-template
          name="asyncCompoundTaskInputGroupOrActivityOutputGroup">
        <xsl:with-param name="a" select="$a"/>
      </xsl:call-template>
    </xsl:otherwise>
  </xsl:choose>
</xsl:template>
```

If isSynchronous

Do this

Else

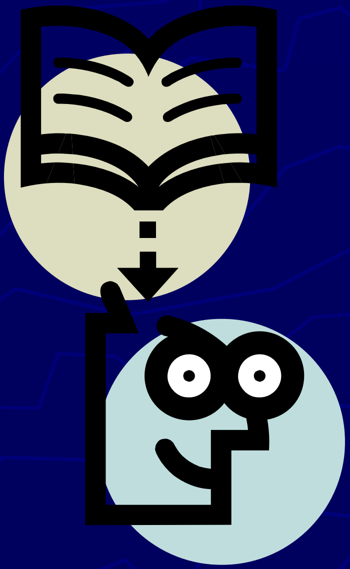
Do that

Dedicated Transformation Language

- Kind of DSL for transformation
- Simplify development and maintenance of model-transformations
- Higher expression power
- Enhanced structuration
  - Composition of rules
  - Interoperability

Dedicated transformation languages

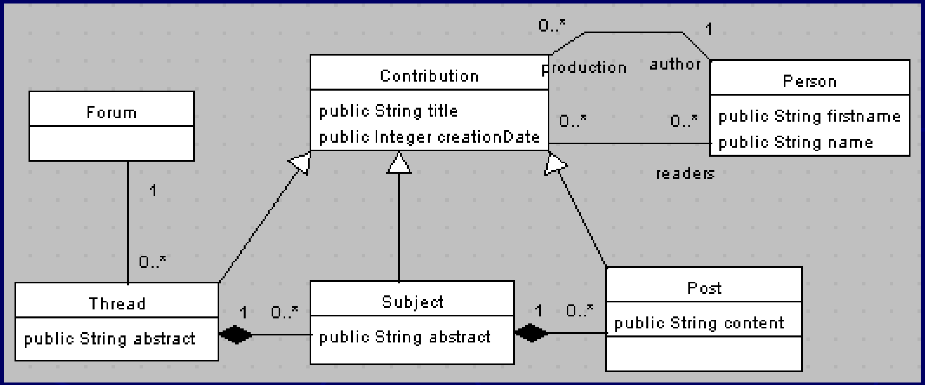
- Terminology
- Features of model transformations



Query

- An expression evaluated over a model
  - Returns one or more instances of types defined either in the source model or by the query language
- OCL is an example of a query language

Examples of OCL queries



OCL Standard Lib Type

Query: *Has Pierre-Alain Muller sent a message about a given subject s?*

```
s.post->exists (author.name='Muller' and author.firstname='Pierre-Alain')
```

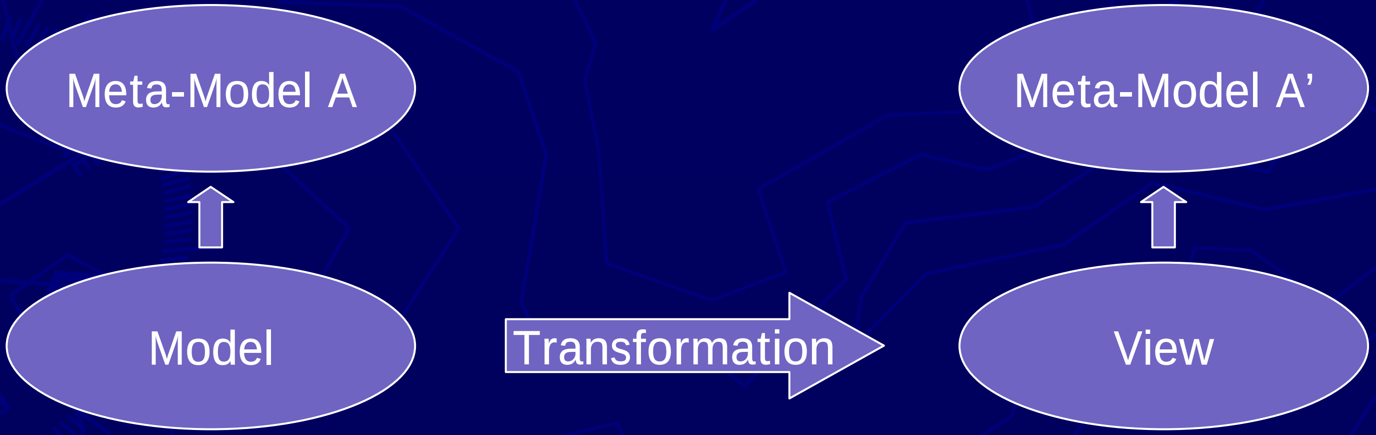
Query: *Knowing that there is only one subject about QVT, I want to retrieve it.*

```
Subject.allInstances()->any (title = ‘QVT’)
```

Model Type

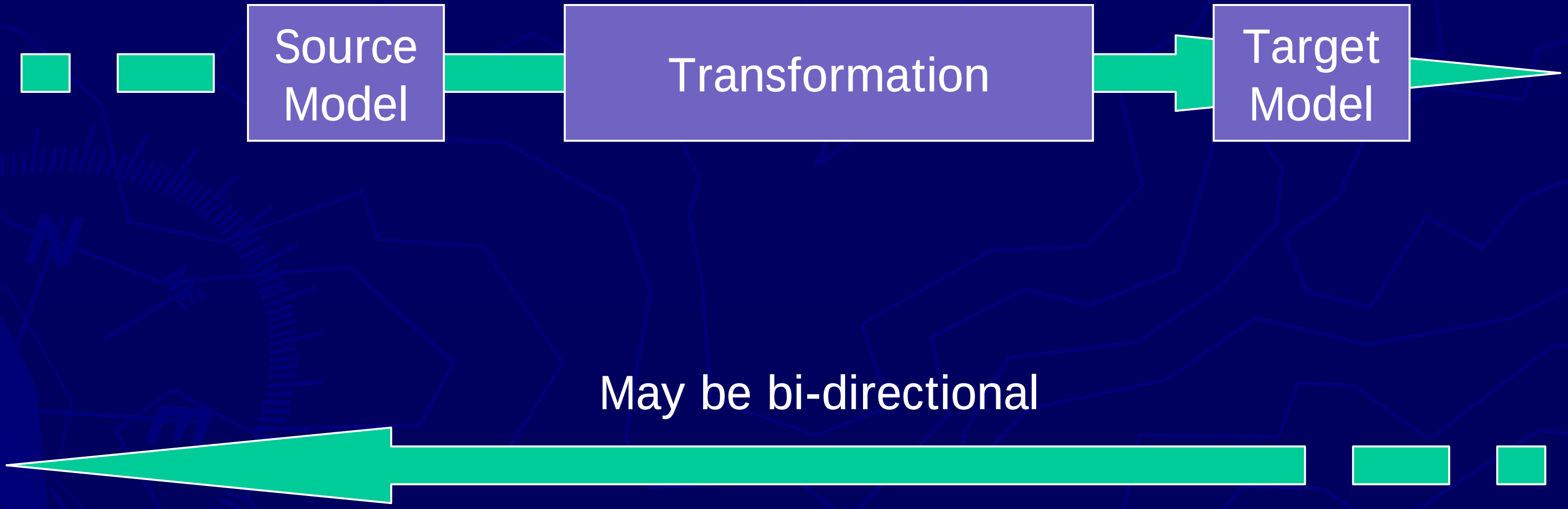
View

- A view is a model that is completely derived from another model
  - The meta-model of the view is typically not the same as the meta-model of the source



Transformation

- A transformation generates target models from source models



Q vs V vs T

- A query is a restricted kind of view
- A view is a restricted kind of transformation
  - The target model cannot be modified independently of the source model
- A transformation generates target models from source models

Declarative

- Declarative languages describe relationships between variables in terms of functions or inference rules and the language executor (interpreter or compiler) applies some fixed algorithm to these relations to produce a result

Imperative

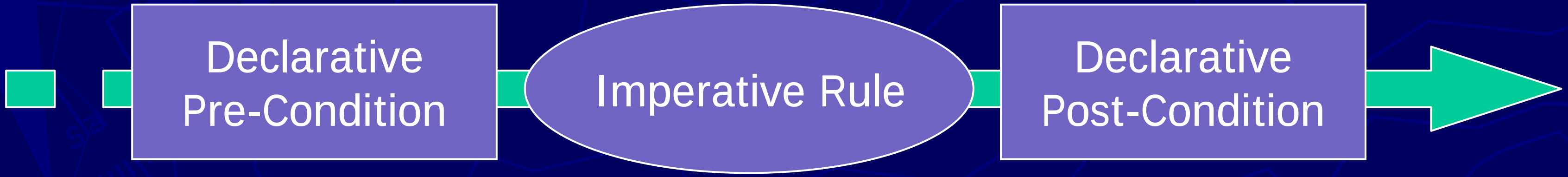
- Any programming language that specifies explicit manipulation of the state of the computer system, not to be confused with a procedural language

Declarative vs. Imperative Style

- Declarative (what to do)
  - Invariant relations between source and target models

- Imperative (how to do it)
  - How to derive a target from a source

- May be combined via pre- and post-conditions



Declarative Pre-Condition

Imperative Rule

Declarative Post-Condition

Execution Strategy

- Invocation of the transformation rules
  - Explicit, via invocation operations (Java like)
  - Implicit, based on context and rules’ signature (Prolog like)

Trace

- Trace associates one (or more) target element with the source elements that lead to its creation
  - For Round-trip development
  - Incremental propagation
- Rules may be able to match elements based on the trace without knowing the rules that created the trace

Rule

- Rules are the units in which transformations are defined
  - A rule is responsible for transforming a particular selection of the source model to the corresponding target model elements.

Declaration

- A declaration is a specification of a relation between elements in the LHS and RHS models

Implementation

- An implementation is an imperative specification of how to create target model elements from source model elements
  - An implementation explicitly constructs elements in the target model
  - Implementations are typically directed

Match

- A match occurs during the application of a transformation when elements from the LHS and/or RHS model are identified as meeting the constraints defined by the declaration of a rule
  - A match triggers the creation (or update) of model elements in the target model

Incremental

- A transformation is incremental if individual changes in a source model can lead to execution of only those rules which match the modified elements

Classification of model transformations

Model-to-Text Approaches

- Visitor-Based Approaches
- Template-Based Approaches

Model-to-Model Approaches

- Direct-Manipulation Approaches
- Relational Approaches
- Graph-transformation-based Approaches
- Structure-Driven Approaches
- Hybrid Approaches
- Other

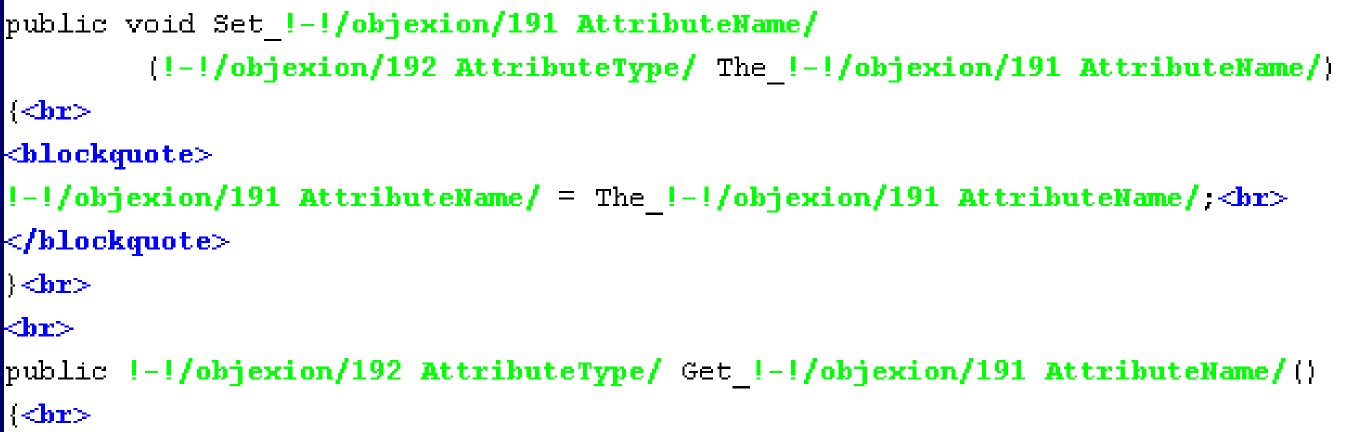
M2T: Visitor-based

- Some visitor mechanisms to traverse the internal representation of a model and write code to a text stream
  - Iterators
  - Write ()

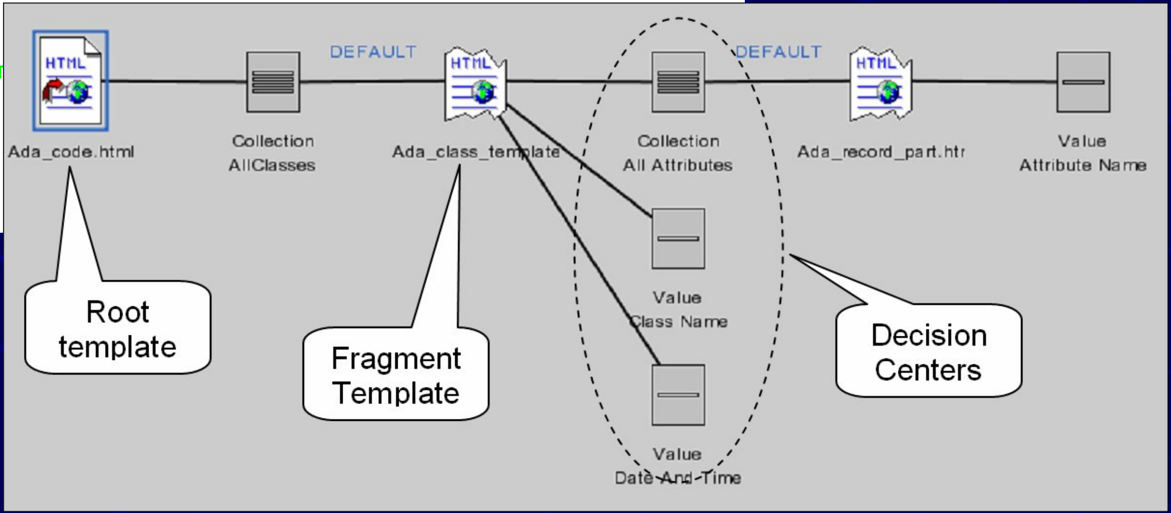
M2T: Template-Based

- A template consists of the target text containing slices of meta-code to access information from the source and to perform text selection and iterative expansion
  - The structure of a template resembles closely the text to be generated
  - Textual templates are independent of the target language and simplify the generation of any textual artefacts

M2T : Template



```
public void Set_!-!/objexion/191 AttributeName/
        (!-!/objexion/192 AttributeType/ The_!-!/objexion/191 AttributeName/)
{<br>
<blockquote>
!-!/objexion/191 AttributeName/ = The_!-!/objexion/191 AttributeName/;<br>
</blockquote>
}<br>
<br>
public !-!/objexion/192 AttributeType/ Get_!-!/objexion/191 AttributeName/()
{<br>
<blockquote>
return !-!/objexion/191 AttributeNam
</blockquote>
}<br>
<br>
```



Root template

Fragment Template

Decision Centers

M2M: Direct Manipulation

- Internal representation plus some API to manipulate it
- Object-oriented framework
- Rules and scheduling implemented from scratch using a programming language

- JMI (MOF-compliant Java Interface)
  - JSR-000040 Java™ Metadata Interface

JMI examples

```
package javax.jmi.model;

import javax.jmi.reflect.*;

public interface Attribute extends StructuralFeature {
    public boolean isDerived();
    public void setDerived(boolean newValue);
}
```

Attributes

Operations

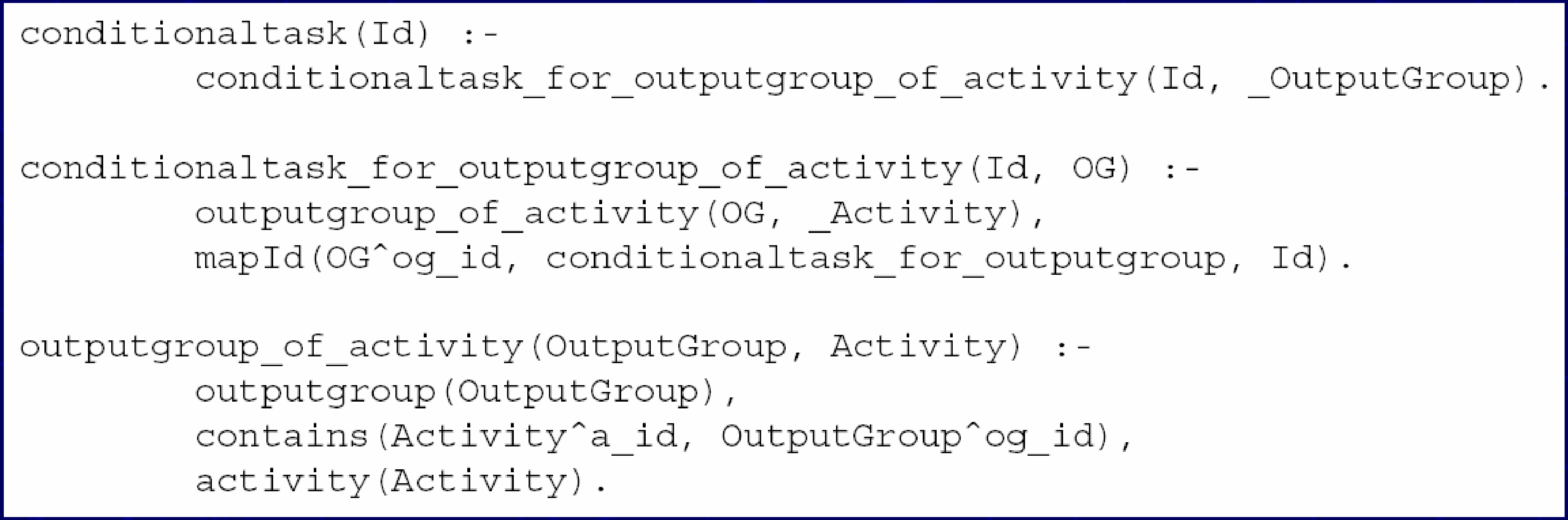
```
package javax.jmi.model;

import javax.jmi.reflect.*;

public interface Operation extends BehavioralFeature {
    public boolean isQuery();
    public void setQuery(boolean newValue);
    public java.util.List getExceptions();
}
```

M2M: Relational Approaches

- Declarative, based on mathematical relations
  - Good balance between flexibility and declarative expression
- Implementable with logic programming
  - Mercury, F-Logic programming languages
  - Predicate to describe the relations
  - Unification based-matching, search and backtracking

Example of logic programming

- Excerpt of Mercury code



```
conditionaltask(Id) :-
        conditionaltask_for_outputgroup_of_activity(Id, _OutputGroup).

conditionaltask_for_outputgroup_of_activity(Id, OG) :-
        outputgroup_of_activity(OG, _Activity),
        mapId(OG^og_id, conditionaltask_for_outputgroup, Id).

outputgroup_of_activity(OutputGroup, Activity) :-
        outputgroup(OutputGroup),
        contains(Activity^a_id, OutputGroup^og_id),
        activity(Activity).
```

M2M : Graph-Transformation-Based

- Declarative, based on the theoretical work on graph transformations
  - Operates on typed, attributed, labeled graphs
  - Rule (LHS, RHS : Graph Pattern)
- Automated source element selection

About Graphs

- G. Rozenberg (ed.); “Handbook of graph grammars and computing by graph transformation: Volume I Foundations”. World Scientific Publishing, 1997.
- Web site of Reiko Heckel ☺

M2M : Graph-Transformation-Based

- Powerfull, but complex because of the non-determinism in scheduling and application strategy
  - Require careful consideration of termination of the transformation process and the rule application ordering

- It is unclear how practitioners will receive these complex approaches

M2M : Structure-Driven Approaches

- 1st Phase
  - Creation of hierarchical structure of target model
- 2nd Phase
  - Set the attributes and references in the target
- Users provide the transformation rules
- Framework determines the scheduling

M2M : Structure-Driven Approaches

- Pragmatic approaches developed in the context of EJB and Databases schema generation from UML models
- Strong support for 1-to-1 and 1-to-n correspondence between source and target
- Unclear how well these approaches can support other kinds of applications

M2M : Hybrid Approaches - others

- Any combination of different techniques
- Practical approaches are very likely to have the hybrid character

Practically speaking

- How many developers are familiar with the prolog-like style of rules writing?
- Where is the advantage of a dedicated explicit language vs. a general purpose language?
- Hybrid Languages or transformation libraries for general purpose languages...

Tools

- Generic transformation tools
- CASE tools scripting languages
- Dedicated model transformation tools
- Meta-modeling tools

Generic transformation tools

- XSLT
- Graph Transformation tools
  - Ask Reiko ☺

CASE tools scripting languages

- **Arcstyler** from Interactive Objects
  - MDA-Cartridge, JPython (Python & Java)
- **Objecteering** from Objecteering Software
  - J language
- **OptimalJ** from Compuware
  - TPL language
- **Fujaba** (From UML to Java and Back Again)
  - Open Source

Dedicated model transformation tools

- **Mia-Transformation** from Mia-Software
  - Inference rules + Java
- **PathMATE** from Pathfinder Solutions
  - Esay to integrate with modeling tools
- **Open-Source**
  - **ATL**, **MTL**, **AndroMDA**, **BOTL**, **Coral Mod-Transf**, **QVTEclipse** or **UMT-QVT**

Meta-modeling tools

- **MetaEdit+** from MetaCase
- **XMF-Mosaic** from Xactium

- **Open-Source**
  - KerMeta from INRIA
  - www.kermeta.org

Coming soon

- **Model Transformations in Practice Workshop**
  - **October 3rd 2005**
  - **Part of the MoDELS 2005 Conference**

- **Comparing and contrasting various approaches**
    - **On Executable Meta-Languages applied to Model Transformations**

References

- M. Andries, G. Engels, A. Habel, B. Hoffmann, H.-J. Kreowski, S. Kuske, D. Kuske, D. Plump, A. Schürr, and G. Taentzer. Graph Transformation for Specification and Programming. Technical Report 7/96, Universität Bremen, 1996, see http://citeseer.nj.nec.com/article/andries96graph.html
- D. H. Akehurst, S.Kent. A Relational Approach to Defining Transformations in a Metamodel. In J.-M. Jézéquel, H. Hussmann, S. Cook (Eds.): *UML 2002 - The Unified Modeling Language 5th International Conference*, Dresden, Germany, September 30 - October 4, 2002. Proceedings, LNCS 2460, 243-258, 2002.
- Alcatel, Softeam, Thales, TNI-Valiosys, Codagen Corporation, et al. MOF Query/Views/Transformations, Revised Submission. OMG Document: ad/03-08-05
- CBOP, DSTC, and IBM. MOF Query/Views/Transformations, Revised Submission. OMG Document: ad/03-08-03
- C. Cleaveland. *Program Generators with XML and Java*. Prentice-Hall, 2001, see http://www.craigc.com/pg/
- K. Czarnecki, S. Helsen, Classification of Model Transformation Approaches, OOPSLA'03 Workshop on Generative Techniques in the Context of Model-Driven Architecture.

References

- T.Gardner, C. Griffin, J. Koehler, R. Hauser, A review of OMF MOF 2.0 QVT Submissions and Recommandations towards the finalm standard, Metamodeliing for MDA, Firstr International Workshop, York, UK, November 2003.
- A. Gerber, M. Lawley, K. Raymond, J. Steel, A. Wood. Transformation: The Missing Link of MDA, In A. Corradini, H. Ehrig, H.-J. Kreowski, G. Rozenberg (Eds.): *Graph Transformation: First International Conference (ICGT 2002)*, Barcelona, Spain, October 7-12, 2002. Proceedings. LNCS vol. 2505, Springer-Verlag, 2002, pp. 90 – 105
- Object Management Group, The Object Constraint Language Specification 2.0, OMG Document: omg/2003-01-07
- Object Management Group, the Model-Driven Architecture Guide, OMG Document: omg/2003-06-01
- Object Management Group, MOF 2.0 Query / Views / Transformations RFP, OMG Document: ad/2002-04-10, revised on April 24, 2002
- QVT-Partners. MOF Query/Views/Transformations, Revised Submission. OMG Document: ad/2003-08-08
- Model Transformation – the Heart and Soul of Model-Driven Software Development, tech report 200352

Questions?