Advances in Model Driven Engineering

Achievements and Challenges

Jean Bézivin

Common work with Frédéric Jouault

{Jean.Bezivin}{noSpamAt}{inria|emn}.fr

AtlanMod Team (INRIA & EMN),
Nantes, France

http://www.emn.fr/x-info/atlanmod/

Powerpoint version of these slides available from author

Nantes is a city
in Western France,
near the Atlantic coast, with
750,000 inhabitants in the
metropolitan area. Nantes is
the most important city
of Brittany
and the 6th town
in France.

UK
Germany
Paris
Italy
Spain

AtlanMod
A common INRIA and
EMNantes
research team
focusing on
Model Driven Engineering

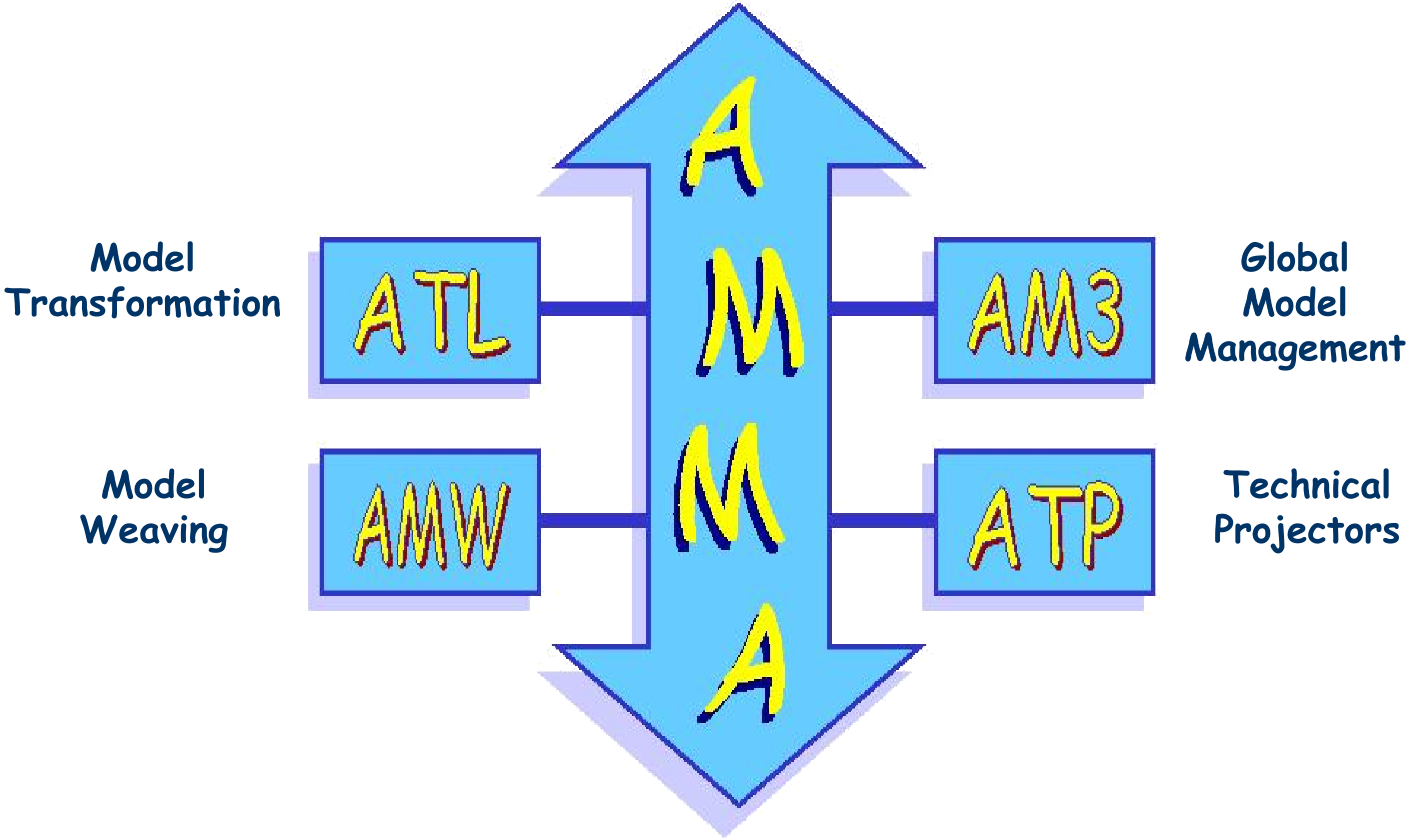
Modeling Technologies for
Software Production,
Operation and Evolution

Achievement and Chalenges in Model Driven Engineering

1. **Introduction**
2. **Basic Mechanisms**
3. **Technical Spaces**
4. **Applications**
5. **Model Taxonomy**
6. **Conclusions**

INTRODUCTION

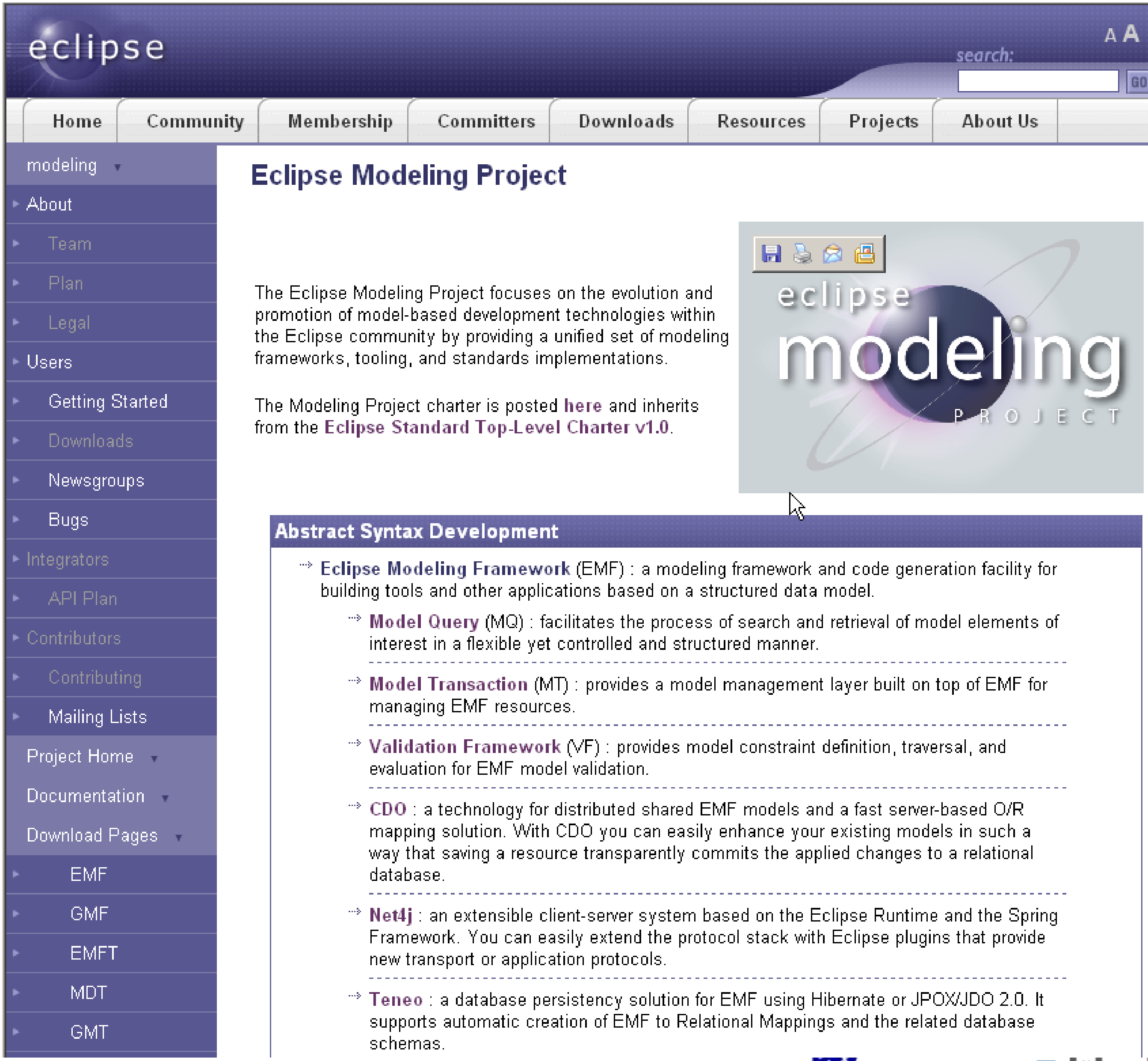
AtlanMod model management Architecture



AmmA as a "DSL Framework", i.e. a framework built from a set of DSLs (KM3, ATL, AMW, AM3, TCS, XCS, BCS, etc.) and intended to build new DSLs (CPL, SPL, etc.)

EMP

- EMP is a MDE technical space organized on the ECORE metameta model.



Eclipse Modeling Project

The Eclipse Modeling Project focuses on the evolution and promotion of model-based development technologies within the Eclipse community by providing a unified set of modeling frameworks, tooling, and standards implementations.

The Modeling Project charter is posted here and inherits from the Eclipse Standard Top-Level Charter v1.0.

Abstract Syntax Development

- **Eclipse Modeling Framework** (EMF) : a modeling framework and code generation facility for building tools and other applications based on a structured data model.
  - **Model Query** (MQ) : facilitates the process of search and retrieval of model elements of interest in a flexible yet controlled and structured manner.
  - **Model Transaction** (MT) : provides a model management layer built on top of EMF for managing EMF resources.
  - **Validation Framework** (VF) : provides model constraint definition, traversal, and evaluation for EMF model validation.
  - **CDO** : a technology for distributed shared EMF models and a fast server-based O/R mapping solution. With CDO you can easily enhance your existing models in such a way that saving a resource transparently commits the applied changes to a relational database.
  - **Net4j** : an extensible client-server system based on the Eclipse Runtime and the Spring Framework. You can easily extend the protocol stack with Eclipse plugins that provide new transport or application protocols.
  - **Teneo** : a database persistency solution for EMF using Hibernate or JPOX/JDO 2.0. It supports automatic creation of EMF to Relational Mappings and the related database schemas.

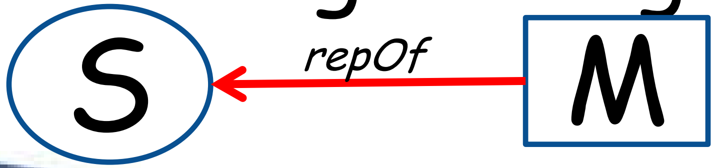
What is a model?

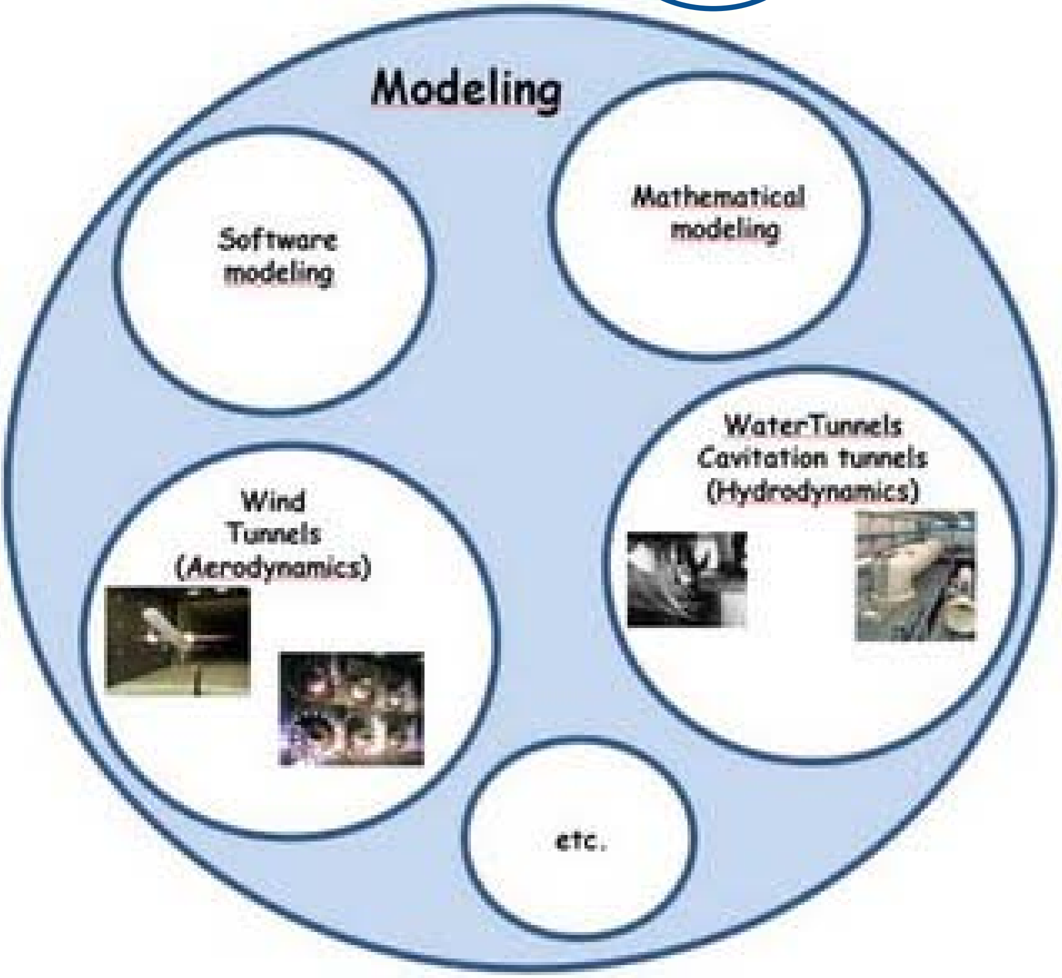
Modeling, in the broadest sense, is the cost-effective use of something in place of something else for some cognitive purpose. It allows us to use something that is simpler, safer or cheaper than reality instead of reality for some purpose. A model represents reality for the given purpose; the model is an abstraction of reality in the sense that it cannot represent all aspects of reality. This allows us to deal with the world in a simplified manner, avoiding the complexity, danger and irreversibility of reality.

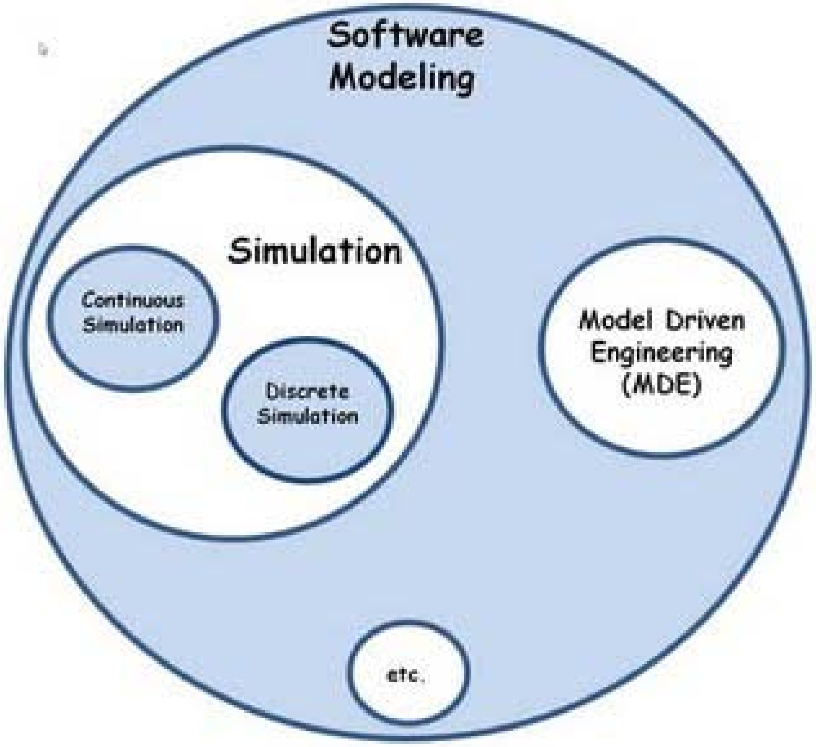
"The Nature of Modeling."
Jeff Rothenberg
in Artificial Intelligence, Simulation, and Modeling,
L.E. William, K.A. Loparo, N.R. Nelson, eds.
New York, John Wiley and Sons, Inc., 1989, pp. 75-92

http://poweredge.stanford.edu/BioinformaticsArchive/PrimarySite/NIHpanelModeling/RothenbergNatureModeling.pdf

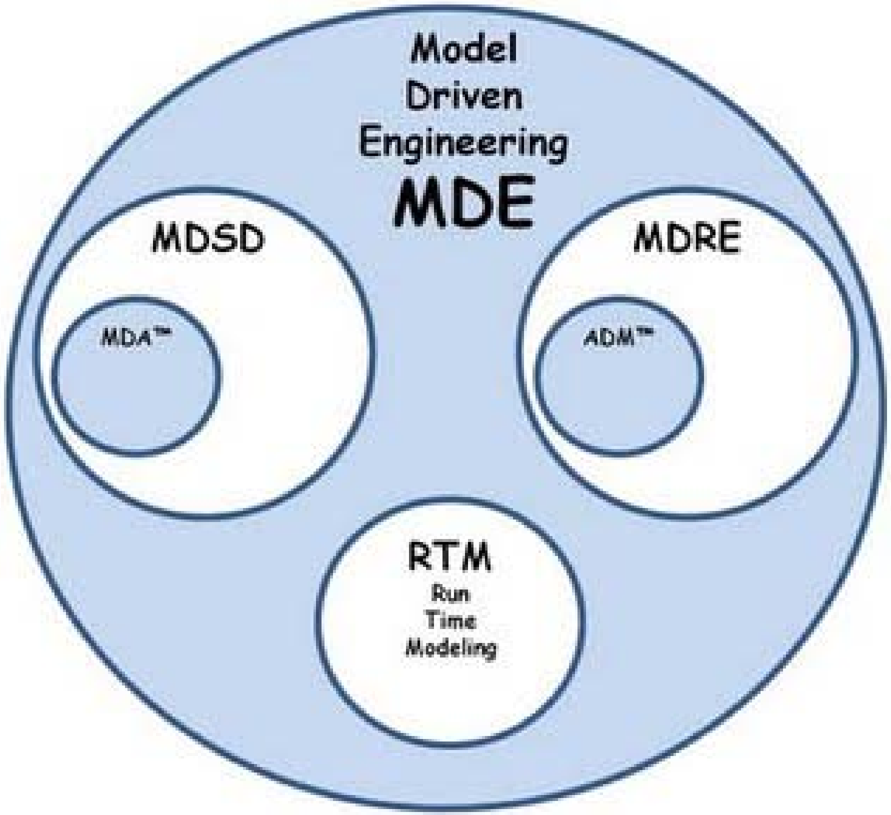
Modeling at large

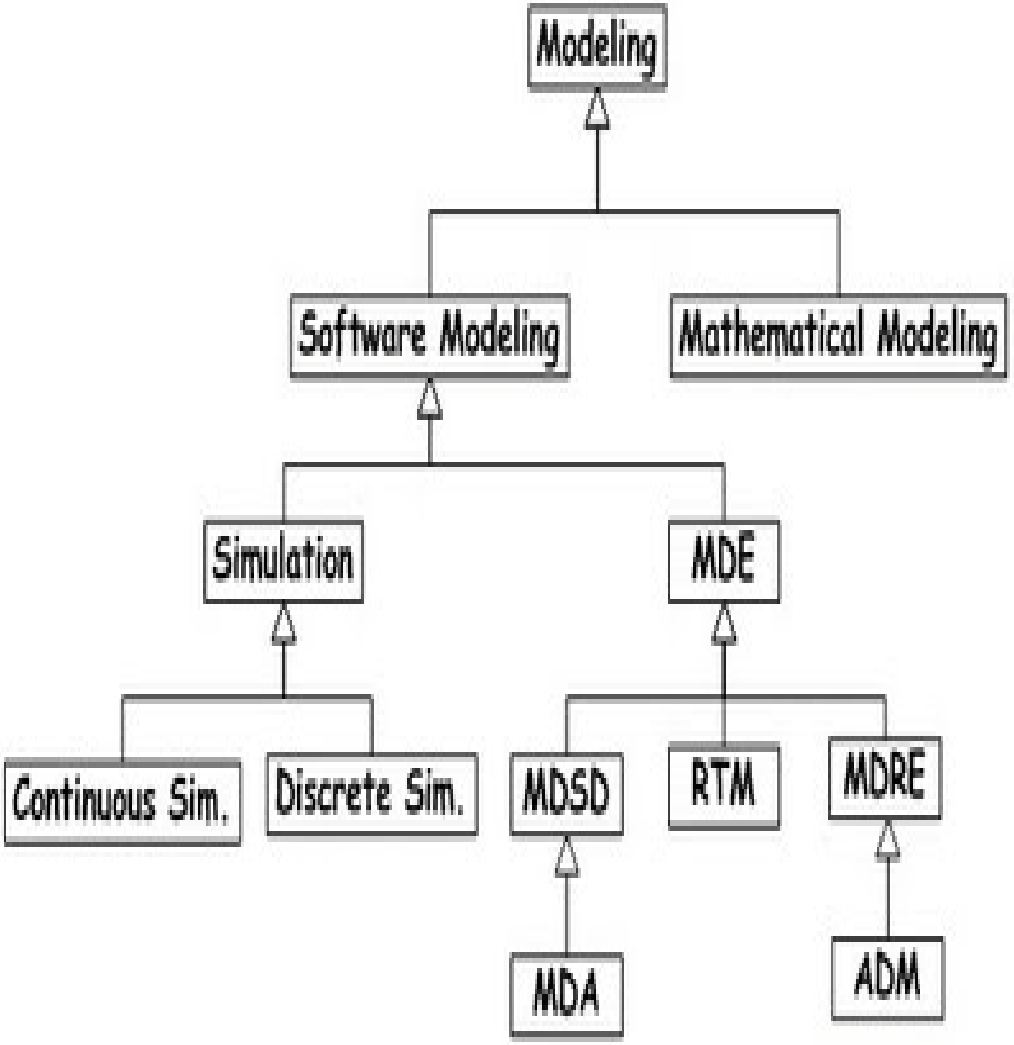



Modeling
Software modeling
Mathematical modeling
Wind Tunnels (Aerodynamics)
WaterTunnels Cavitation tunnels (Hydrodynamics)
etc.



Modeling at large





First some loose definitions of what is a model

- Phil Bernstein, “A Vision for Management of Complex Systems”.

  A model is a complex structure that represents a design artifact such as a relational schema, an interface definition (API), an XML schema, a semantic network, a UML model or a hypermedia document.

- OMG, “UML Superstructure”.

  A model captures a view of a physical system. It is an abstraction of the physical system, with a certain purpose. This purpose determines what is included in the model and what is relevant. Thus the model completely describes those aspects of the physical system that are relevant to the purpose of the model, at the appropriate level of detail.

- OMG, “MDA Guide”.

  A formal specification of the function, structure and/or behavior of an application or system.

- Steve Mellor, et al., “UML Distilled”

  A model is a simplification of something so we can view, manipulate, and reason about it, and so help us understand the complexity inherent in the subject under study.

- Anneke Kleppe, et. al. “MDA Explained”

  A model is a description of (part of) a system written in a well-defined language. A well-defined language is a language with well-defined form (syntax), and meaning (semantics), which is suitable for automated interpretation by a computer.

✓*All of these definitions are partially correct*
✓*None is complete*
✓*None is really useful for the real engineer*
✓ *We need a workable definition for “model”*

Multiple Acronyms

- **MDE** Model Driven Engineering
- **ME** Model Engineering
- **MDA** Model Driven Architecture
- **MDD** Model Driven Development
- **MDSD** Model Driven Software Development
- **MDSE** Model Driven Software Engineering
- **MBD** Model Based Development
- **MM** Model Management
- **ADM** Architecture Driven Modernization
- **DSL** Domain Specific Language
- **DSM** Domain Specific Modeling
- **DDD** Domain Driven Design
- **MDRE** Model Driven Reverse Engineering
- **MD*** (Markus Voelter)
- etc.

A definition of MDA

OMG/ORMSC/2004-06-01 (The OMG MDA Guide): A Definition of MDA (The following was approved unanimously at the ORMSC plenary session, meeting in Montreal on 23 August 26, 2004. The stated purpose of these two paragraphs was to provide principles to be followed in the revision of the MDA Guide.)

- MDA is an OMG initiative that proposes to define a set of non-proprietary standards that will specify interoperable technologies with which to realize model-driven development with automated transformations.
- MDA does not necessarily rely on the UML, but, as a specialized kind of MDD (Model Driven Development), MDA necessarily involves the use of model(s) in development, which entails that at least one modeling language must be used.
- Any modeling language used in MDA must be described in terms of the MOF language, to enable the metadata to be understood in a standard manner, which is a precondition for any ability to perform automated transformations.

IBM MDA manifesto: Three complementary ideas

MDA Journal

May 2004

Grady Booch

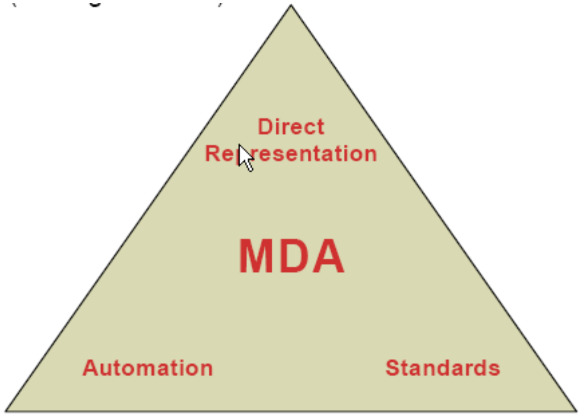
Alan Brown

Sridhar Iyengar

James Rumbaugh

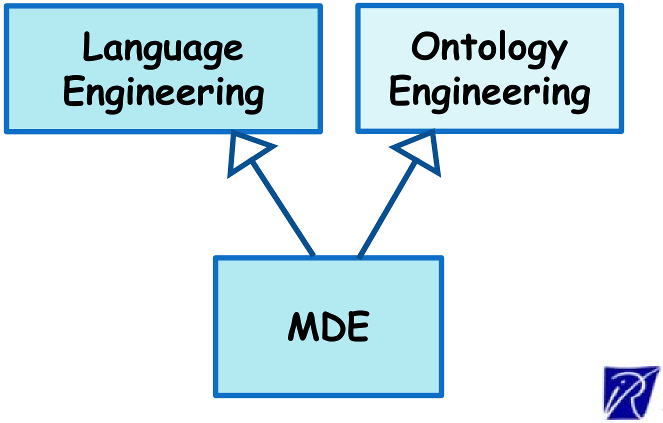
Bran Selic

IBM Rational Software



MDE vs DSLS

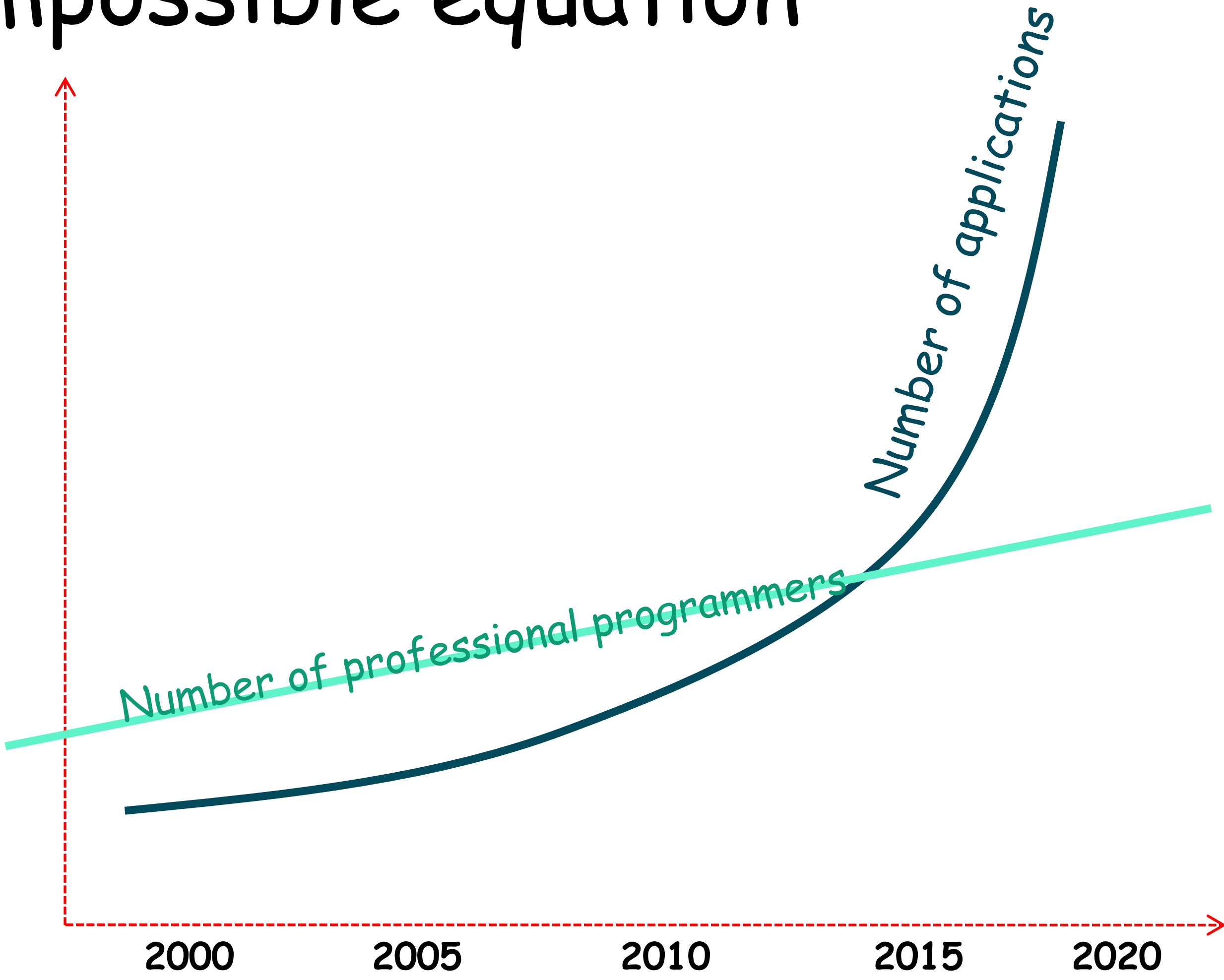
1. Direct representation
2. Automation
3. Standards


Language Engineering

Ontology Engineering

MDE

The impossible equation



NB:
Unfortunately no much relations between the research communities of MDE and End-User programming.

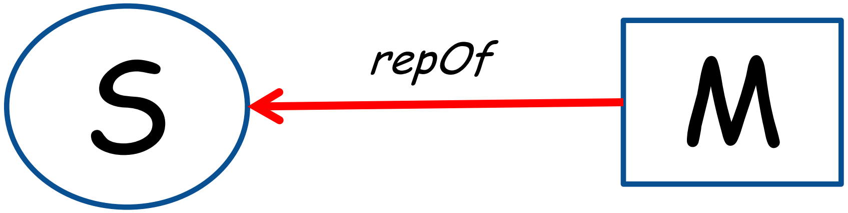
USA:

90 Millions computer users;
50 Millions Spreadsheet & DB users;
12 Millions self described programmers;
3 Millions professional programmers;

A definition of MDE?

- The use of typed graphs as the main artefact to represent phenomenon of the real world (to understand them, to act on them)

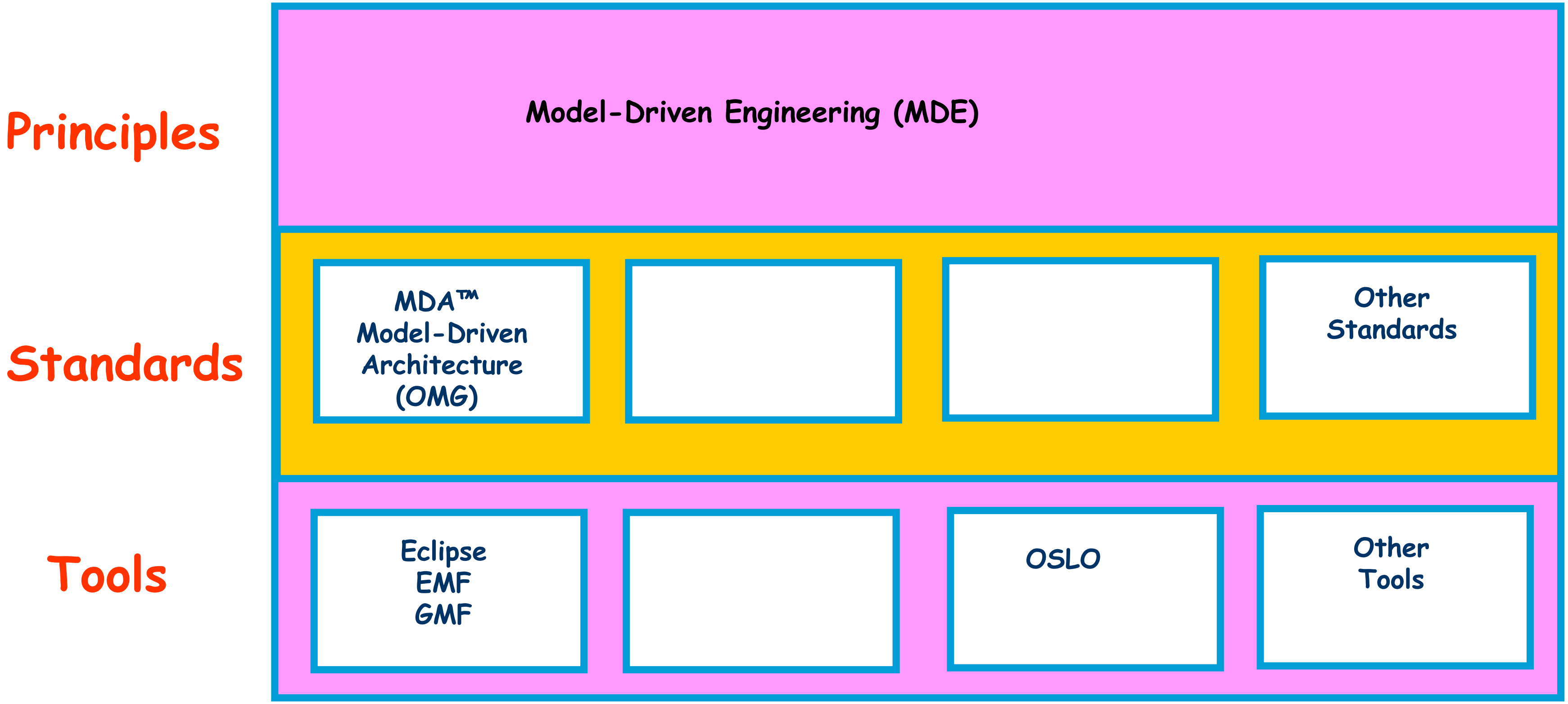
- Systematic use of the representation relation *repOf (M,S)* between systems and models



- Agile metamodeling (working with precise, open and explicit metamodels)

- Three main operations on models: create/delete, store/retrieve, transform.

Overview of Model-Driven Engineering (MDE)



Principles

Model-Driven Engineering (MDE)

Standards

MDA™
Model-Driven
Architecture
(OMG)

Other
Standards

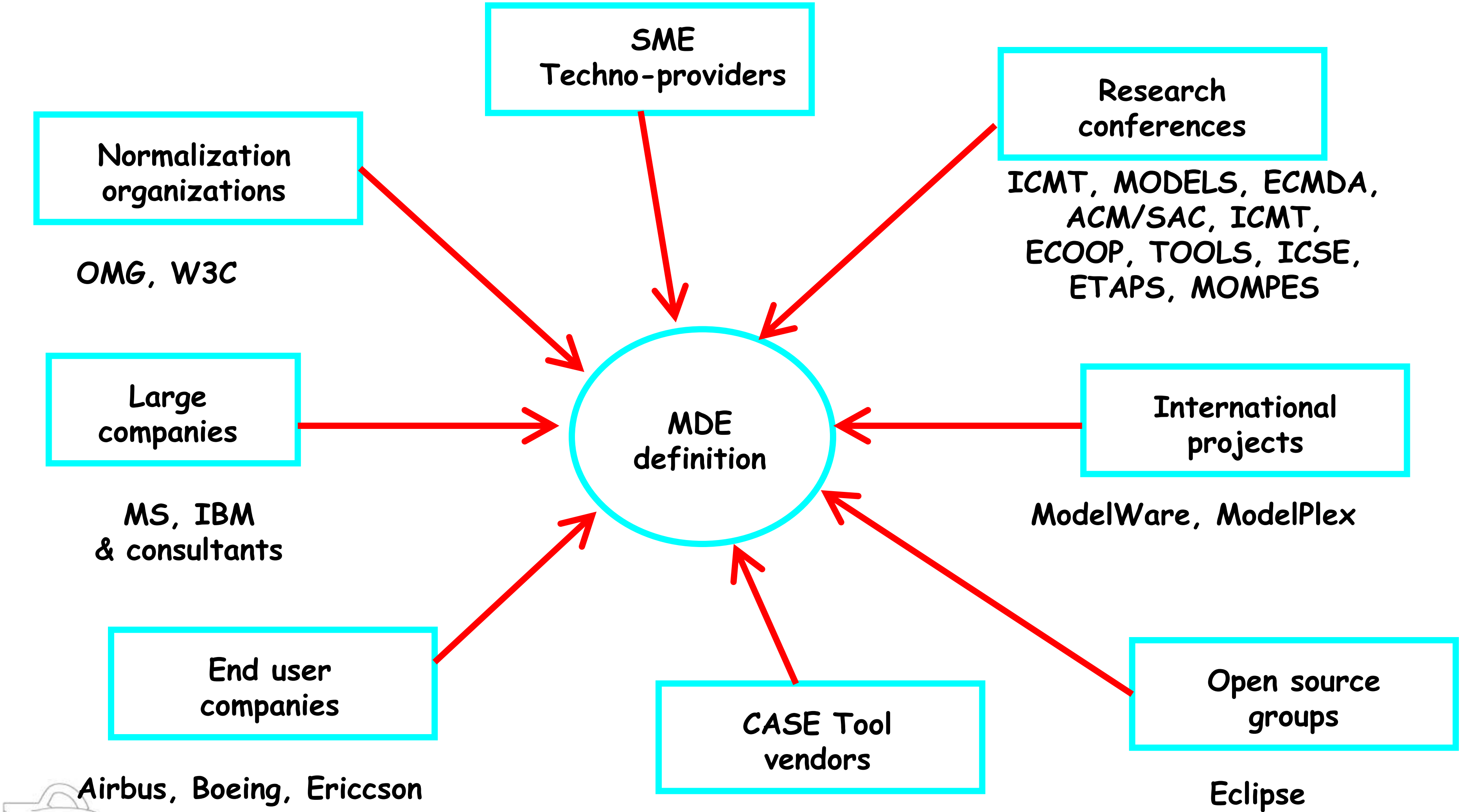
Tools

Eclipse
EMF
GMF

OSLO

Other
Tools

Influencing parties (some)



SME Techno-providers

Normalization organizations

OMG, W3C

Research conferences

ICMT, MODELS, ECMDA, ACM/SAC, ICMT, ECOOP, TOOLS, ICSE, ETAPS, MOMPES

Large companies

MS, IBM & consultants

MDE definition

International projects

ModelWare, ModelPlex

End user companies

Airbus, Boeing, Ericcson

CASE Tool vendors

Open source groups

Eclipse

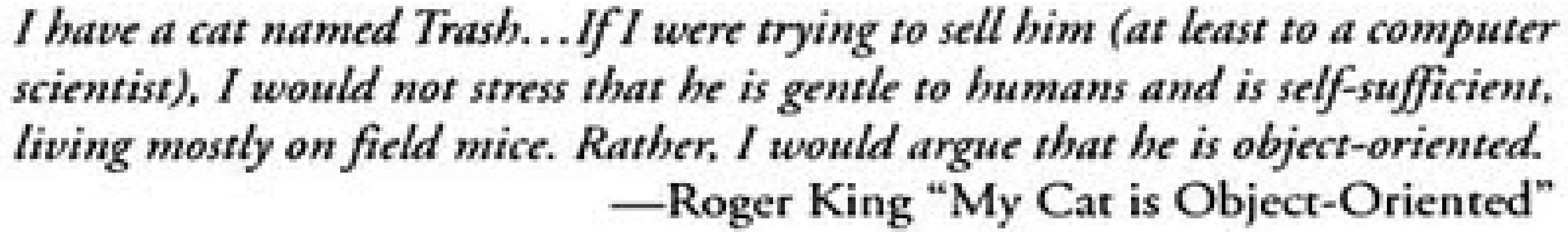
Found on a Blog

One observation:

The lack of grounded theories relating to models (ModelWare) is obvious, which can be seen in the road map for object-oriented models (ref…). Hence, the current modeling practice, UML, doesn't have a grounded theoretical foundation, although it is still widely used in software development world-wide.

My cat is Model-Driven

- Everything is nice with models, but …
  - Criticisms on MDE
  - Exaggerated hype
  - Limited tooling
  - Needs to be precise on the technology
  - Needs to be precise on the applicability scope
  - OVERSELLING

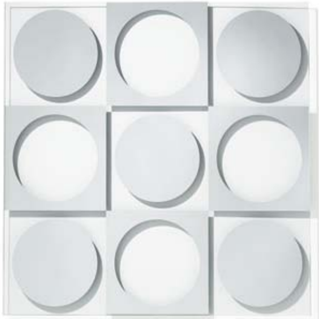


> *I have a cat named Trash...If I were trying to sell him (at least to a computer scientist), I would not stress that he is gentle to humans and is self-sufficient, living mostly on field mice. Rather, I would argue that he is object-oriented.*
>
> —Roger King "My Cat is Object-Oriented"

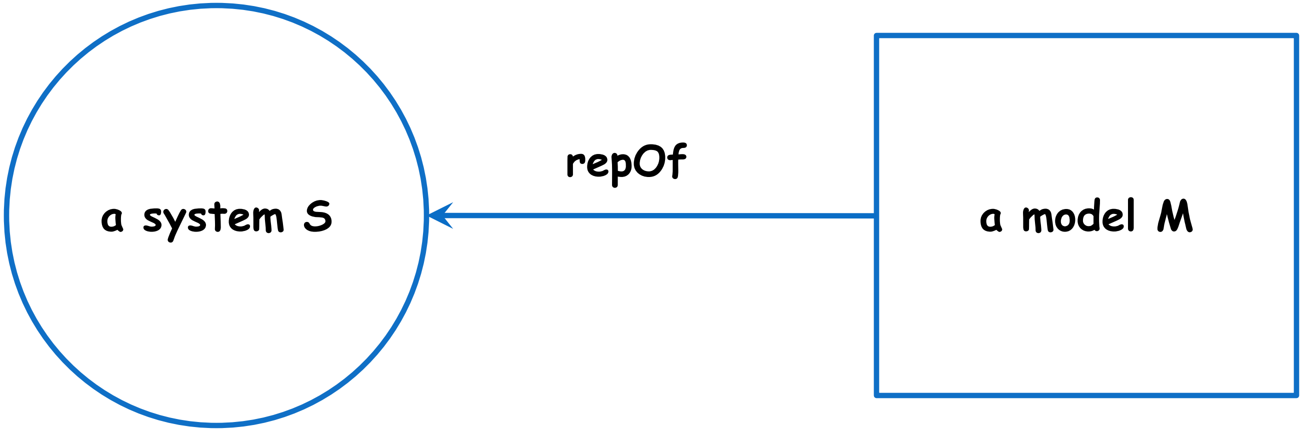
[1] Roger King *My Cat Is Object-Oriented.* Object-Oriented Concepts, Databases, and Applications 1989: 23-30

BASIC CORE MECHANISMS

Systems and Models



Squares and Circles
by David Riley



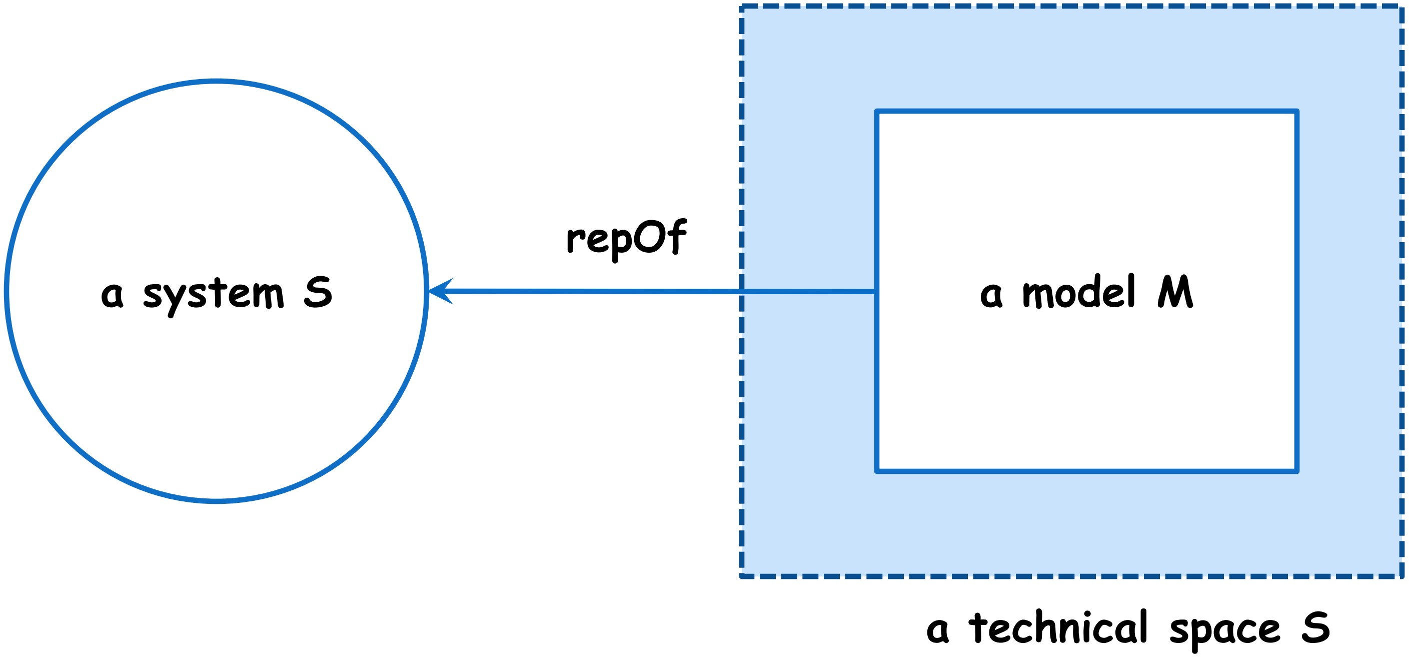
System and Model

Caution: These are only plastic food models, don't eat them.

Technical Spaces

✓ Each model is expressed in some representation system, named a "technical space"

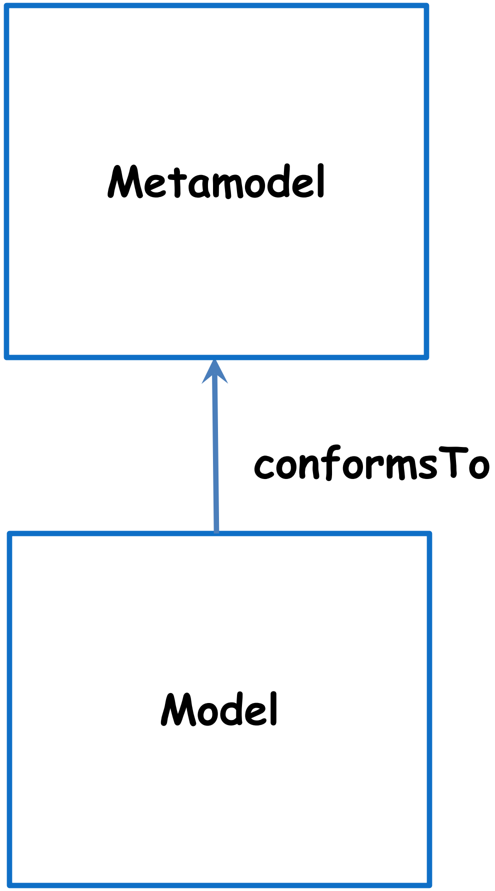
✓ Some technical spaces are based on trees, other on graphs, others on hypergraphs, etc. There are a lot of possible representation systems.



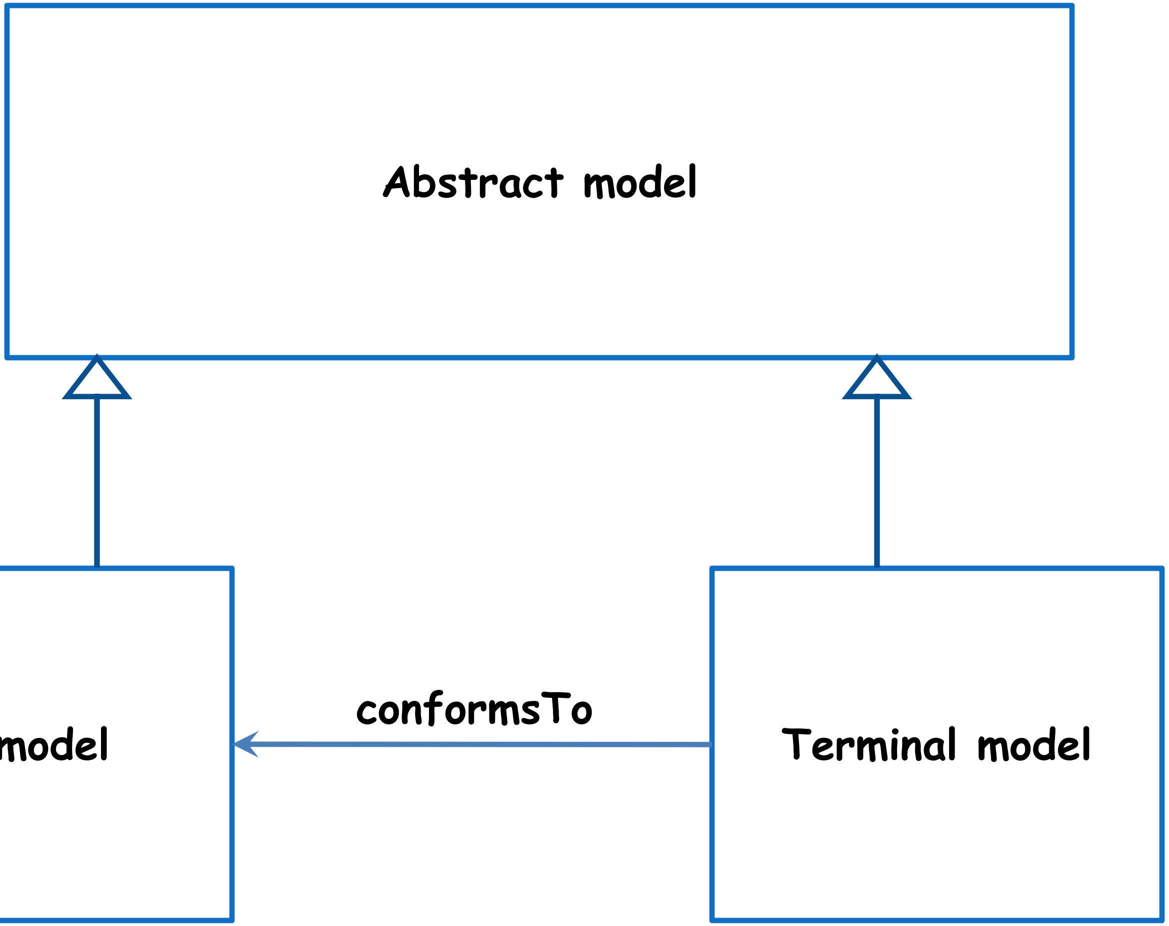
Metamodeling

A metamodel is a simplified ontology, i.e. a set of concepts and relations between these concepts.

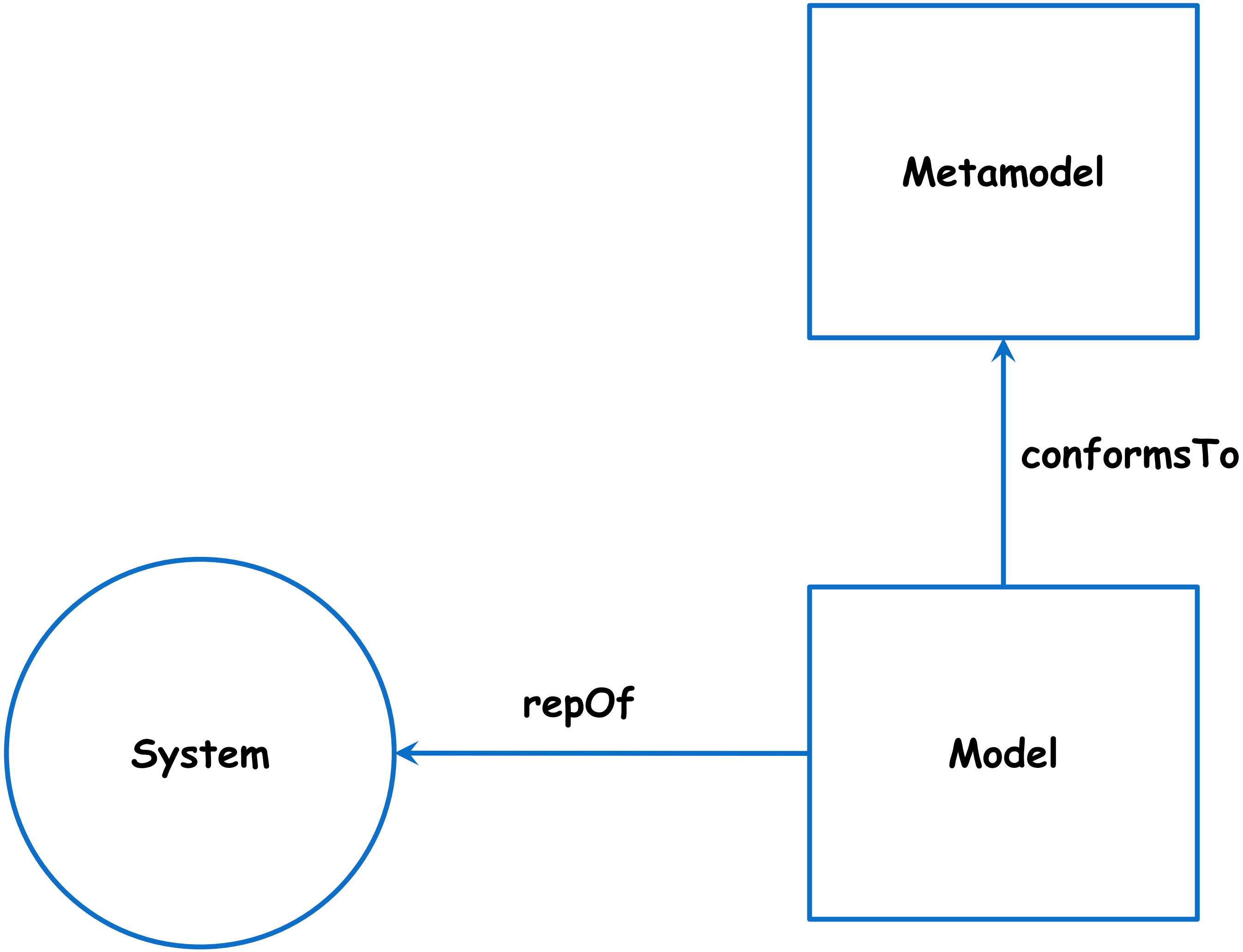
A model is a graph composed of elements (nodes and edges). Each such element corresponds to a concept in the metamodel.



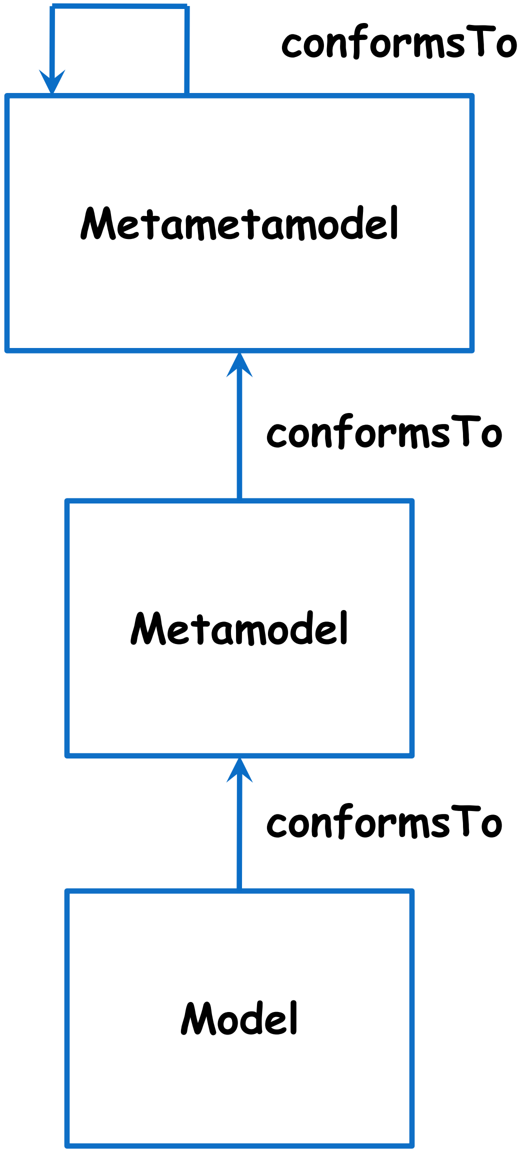
Abstract Models



Representation and Conformance



MetaMetaModels



conformsTo

Metametamodel

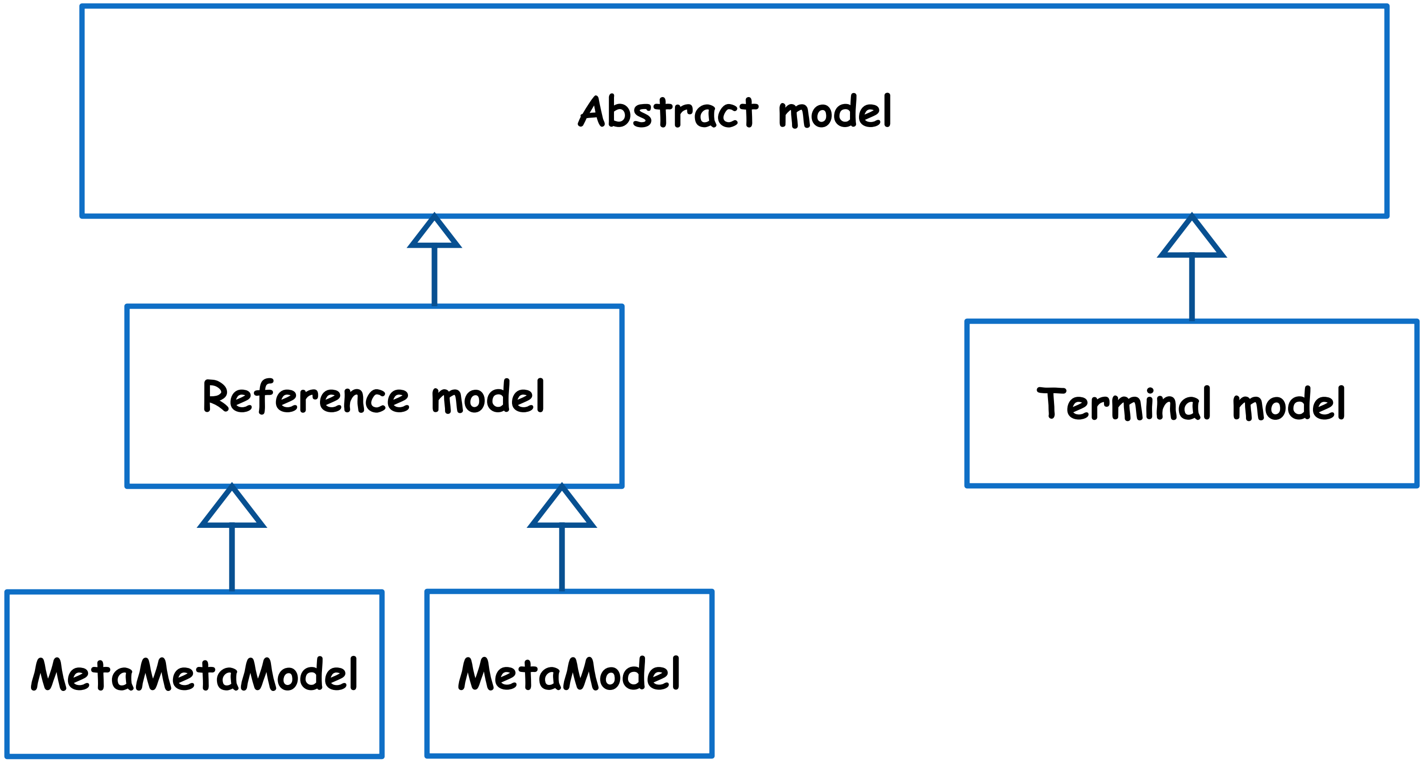
conformsTo

Metamodel

conformsTo

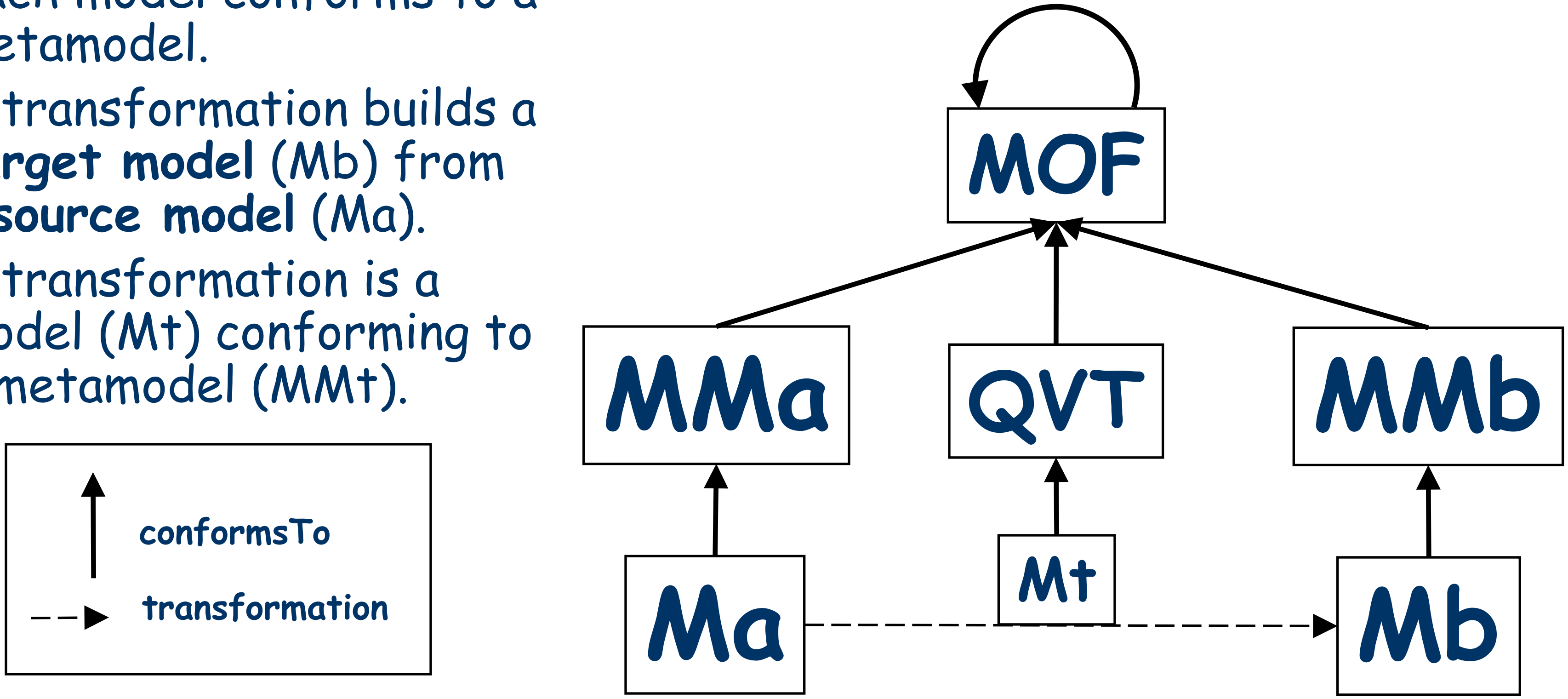
Model

Abstract Models



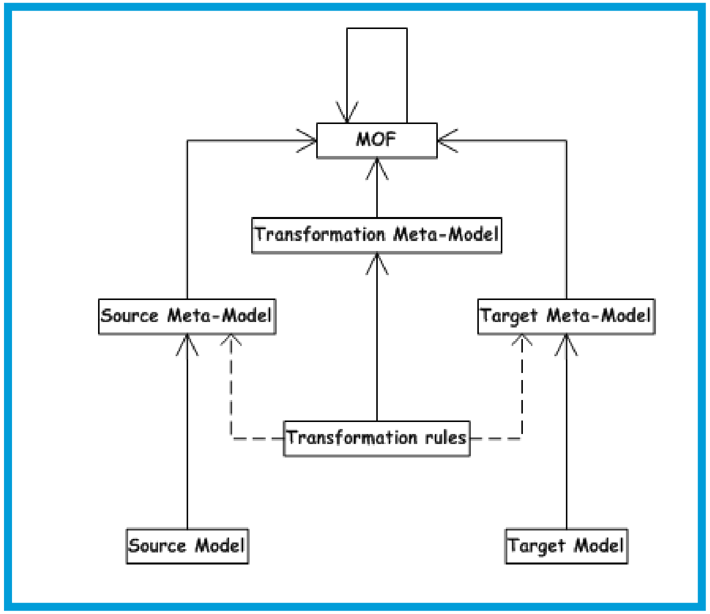
Transformations as Models

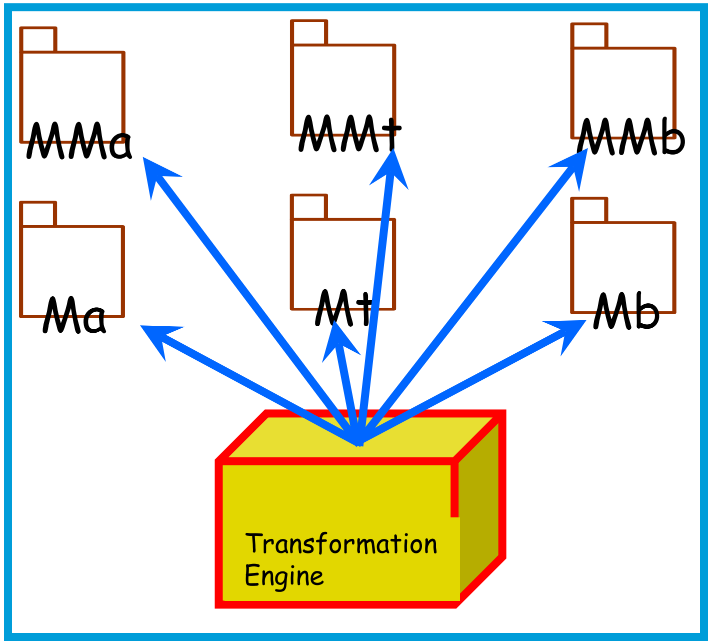
- Each model conforms to a metamodel.
- A transformation builds a **target model** (Mb) from a **source model** (Ma).
- A transformation is a model (Mt) conforming to a metamodel (MMt).



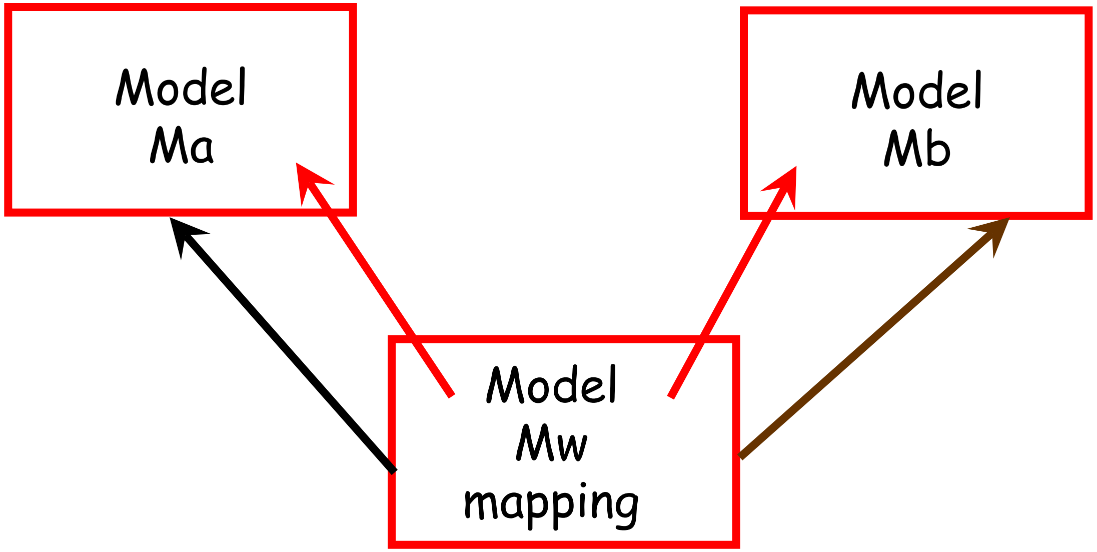
Transformations as models

- Treating everything as a model leads not only to conceptual simplicity and regular architecture, but also to implementation efficiency.
- ATL is composed of a transformation virtual machine plus a metamodel-driven compiler.
- The transformation VM allows uniform access to model and metamodel elements.
- Three generations of VMs:
  - Procedure oriented (Wirth's P-machine)
  - Object oriented (Smalltalk bytecode, Java VM)
  - Model oriented (ATL VM, uniform access to models and model elements)

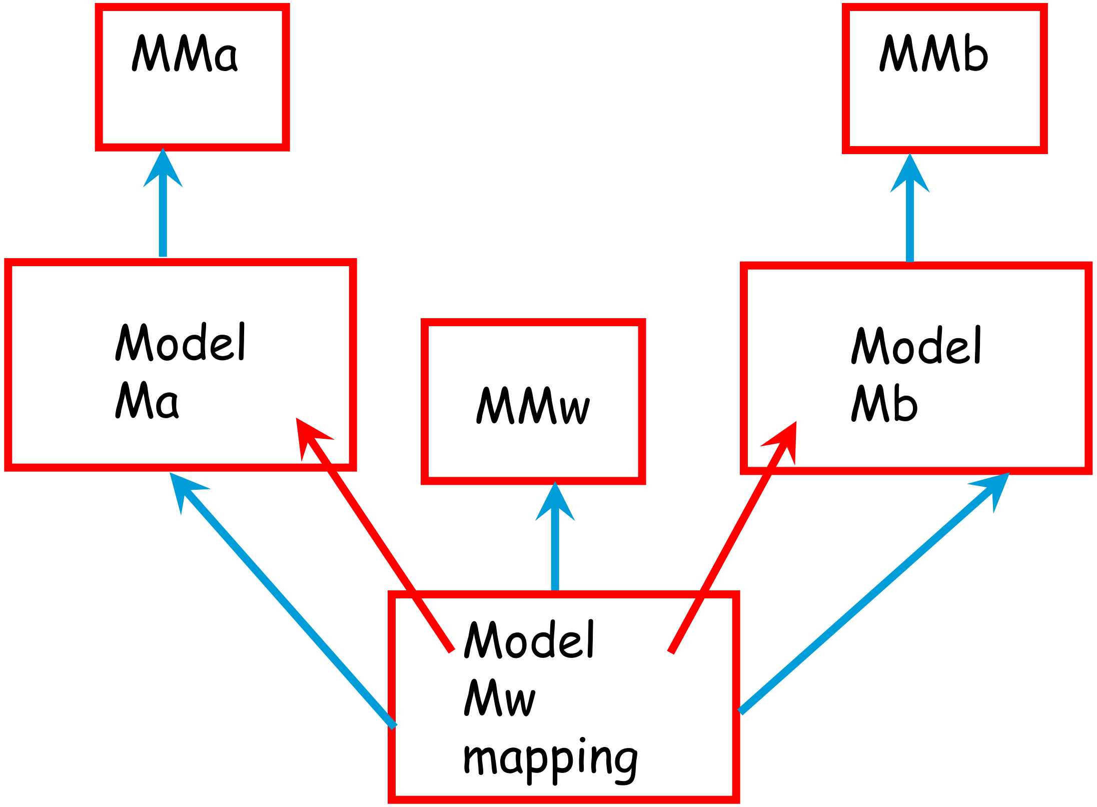




Correspondences as models (Weaving)



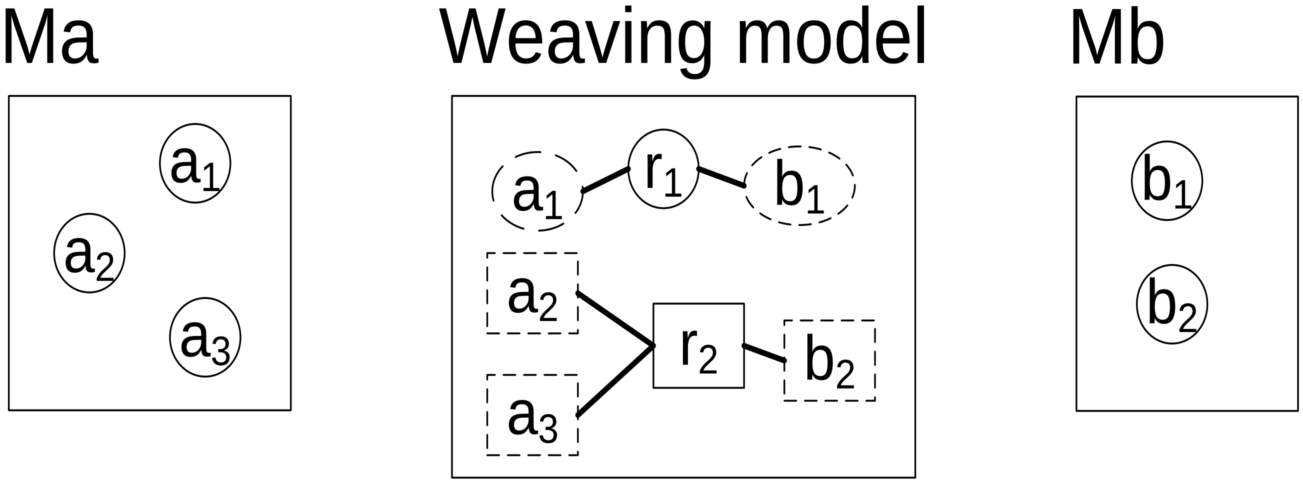
Correspondences as models (Weaving)



Relationships as a model: the weaving technique

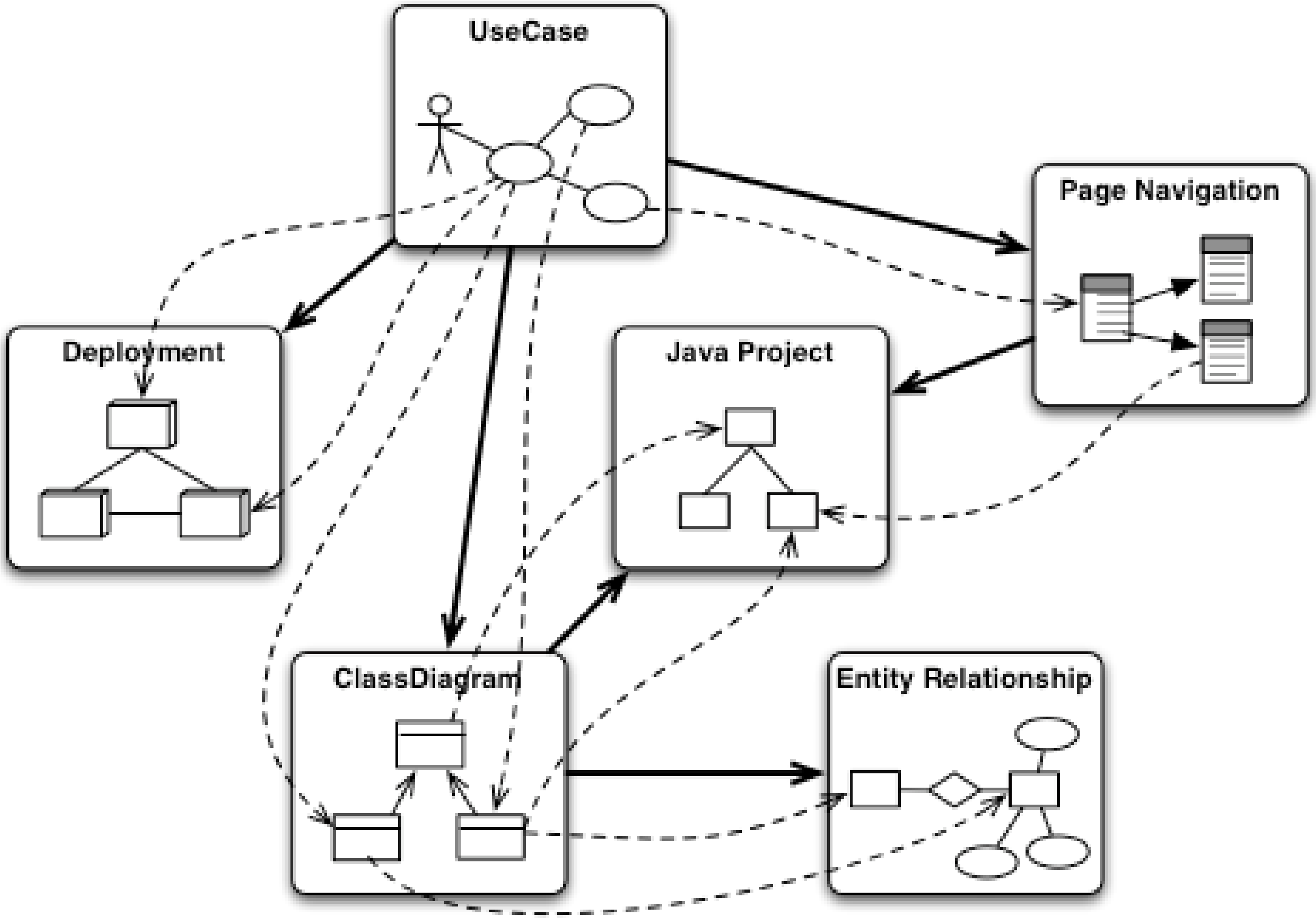
- To capture relationships between model elements
- Relationships are "reified" in a **weaving model**
  - The model elements represent the relationships and the related elements
  - As any kind of model, the weaving model can be saved, stored, transformed, modified, etc.

AMW

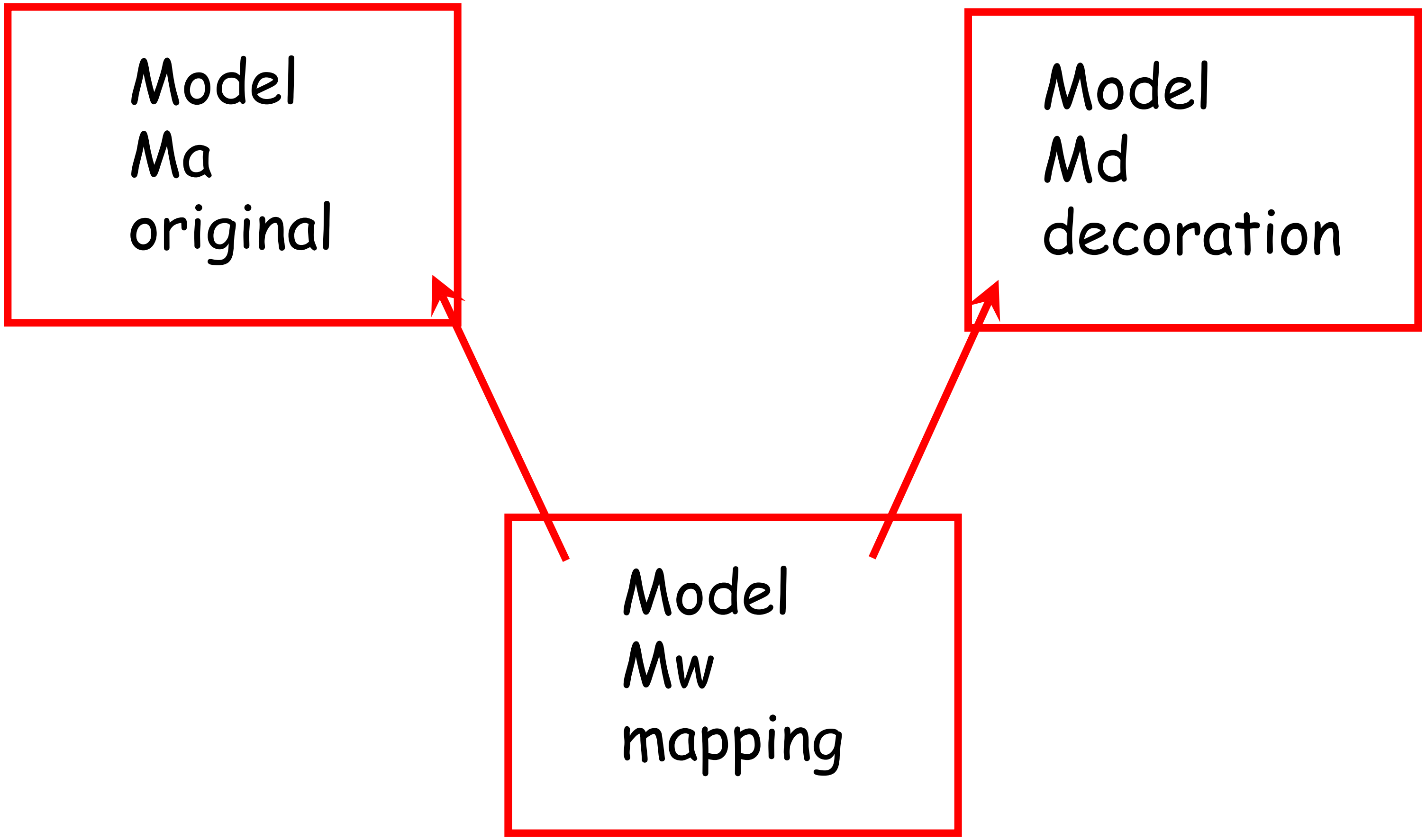


Pollution-free traceability

Petstore Application Navigability



General annotation/decoration scheme

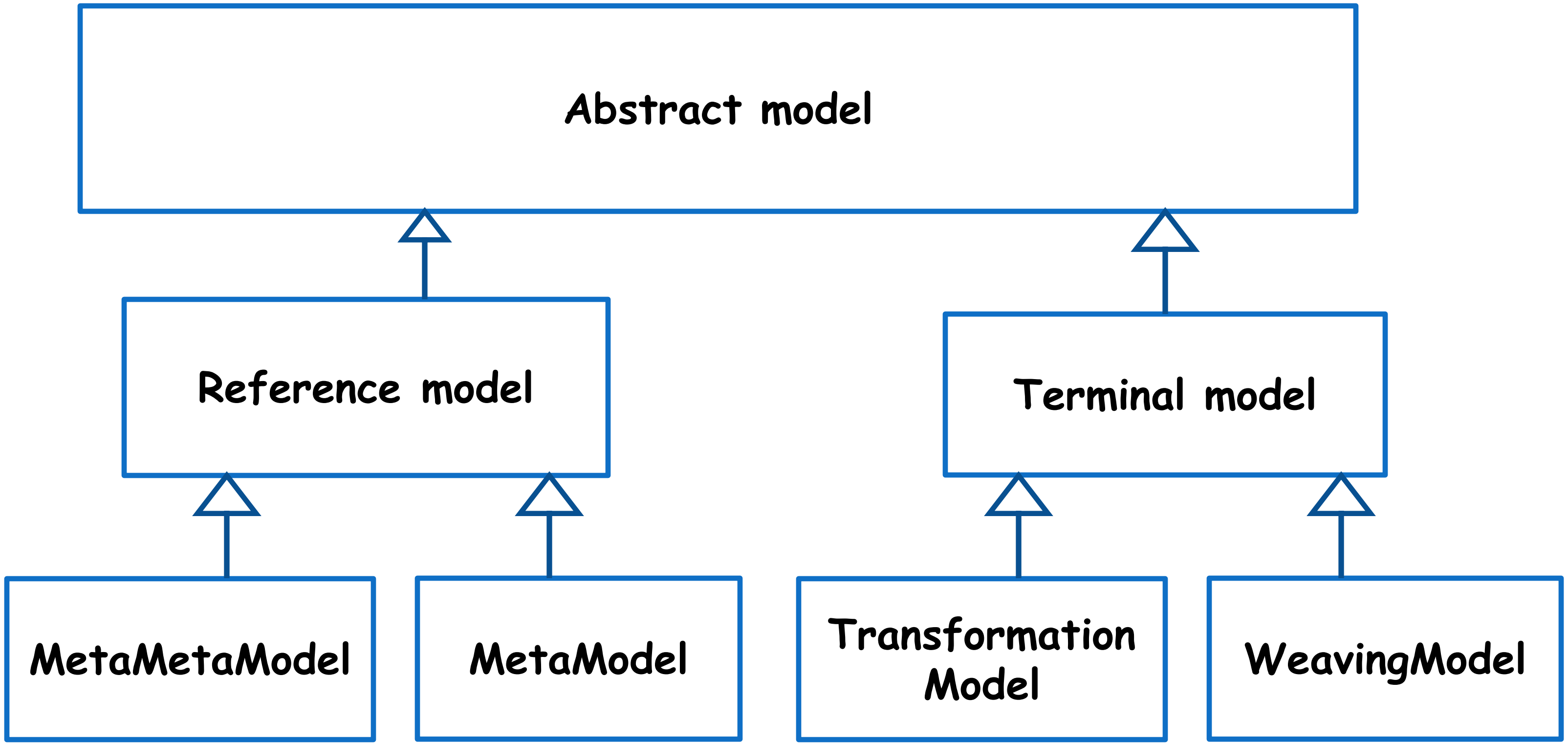


Model
Ma
original

Model
Md
decoration

Model
Mw
mapping

Abstract Models



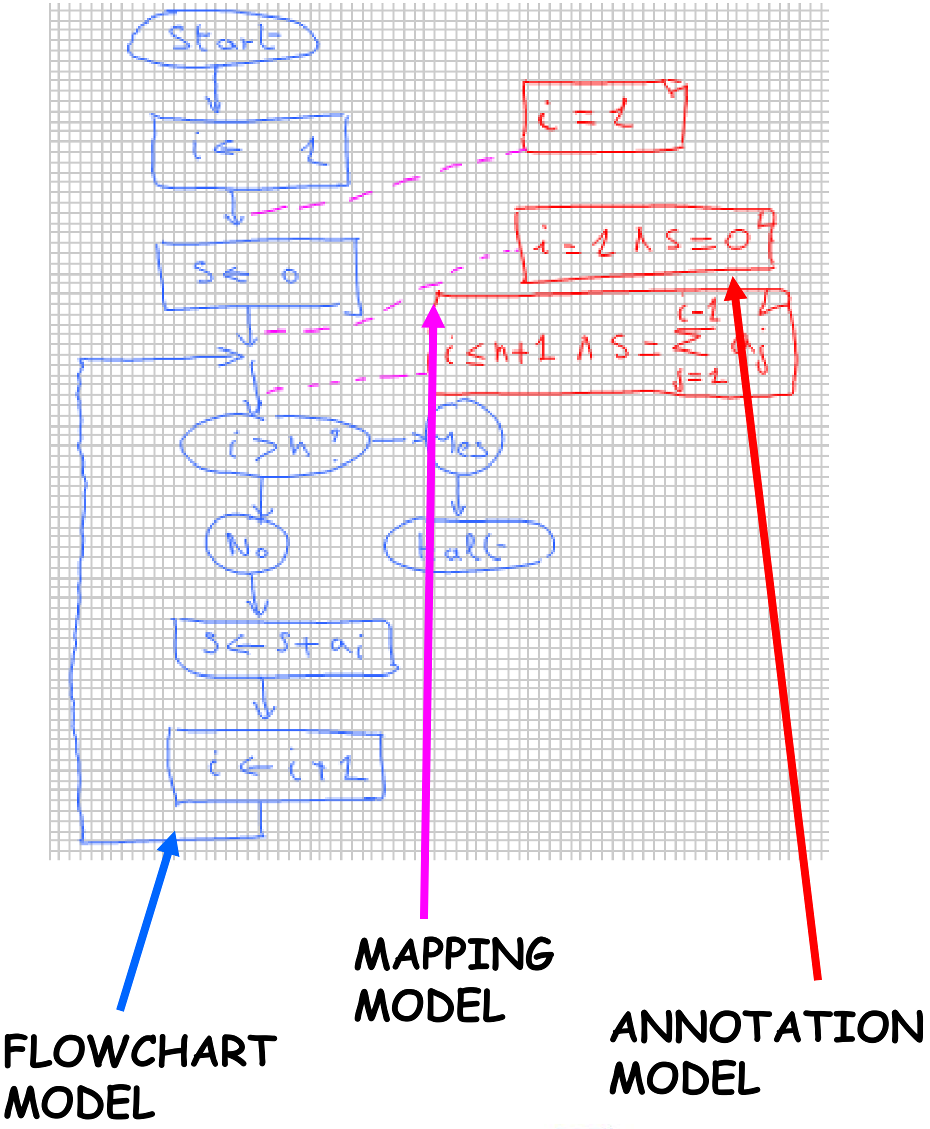
Assigning meanings to models ?

- Floyd established the foundation of modern assertion techniques by proposing to decorate a program with specific annotations (pre and post conditions)
- "An interpretation I of a flowchart is a mapping of its edges on propositions"

Robert W Floyd

"Assigning meanings to programs"

Symposia in applied mathematics, 1965

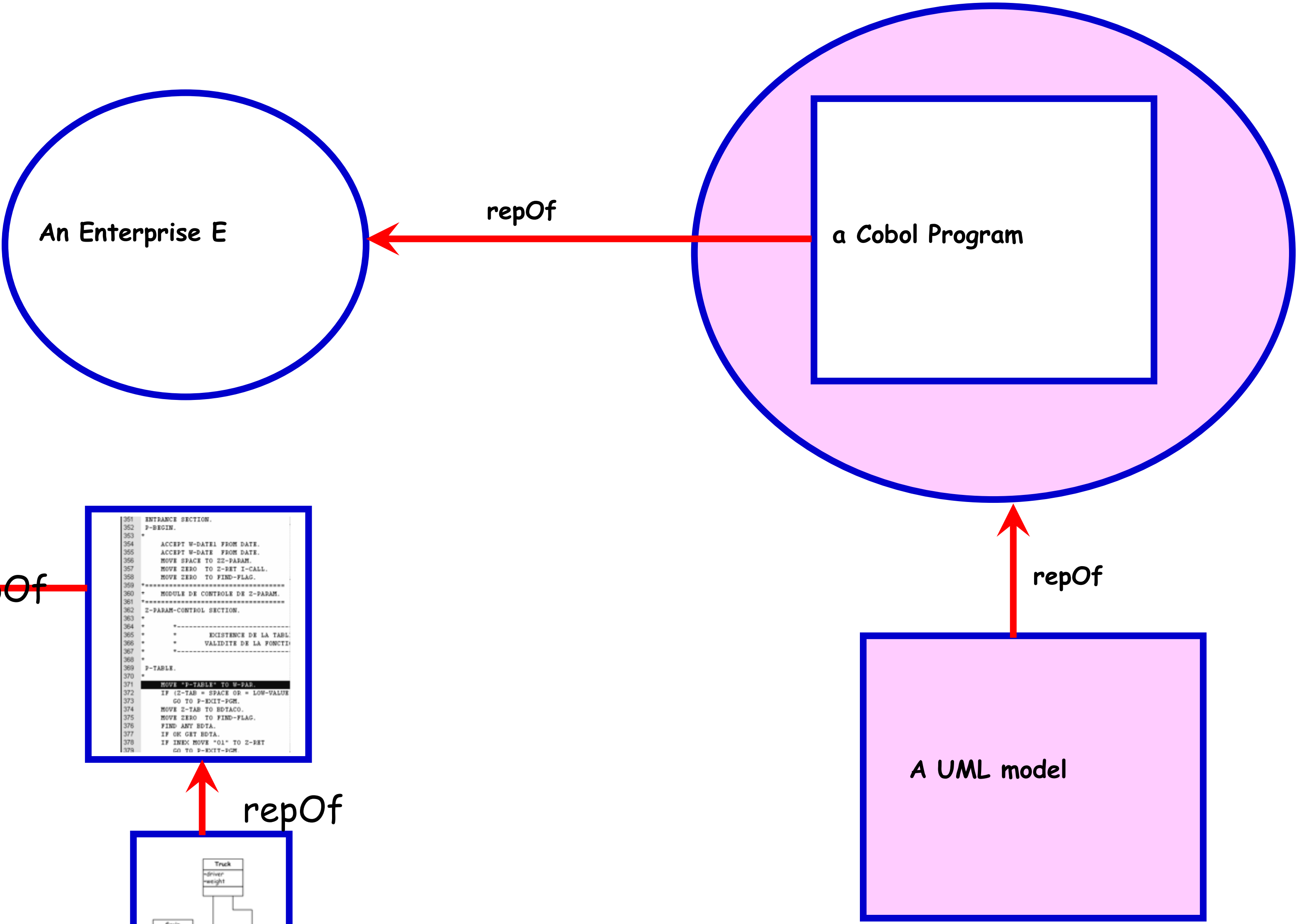


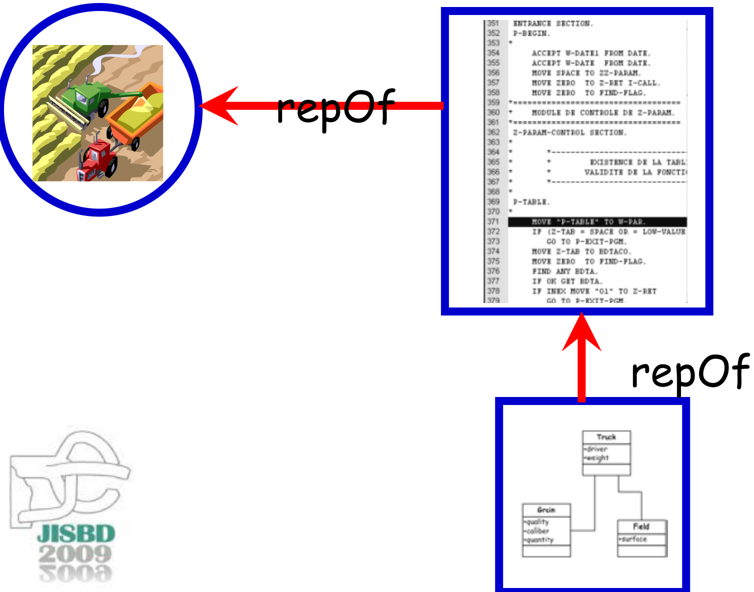
FLOWCHART MODEL

MAPPING MODEL

ANNOTATION MODEL

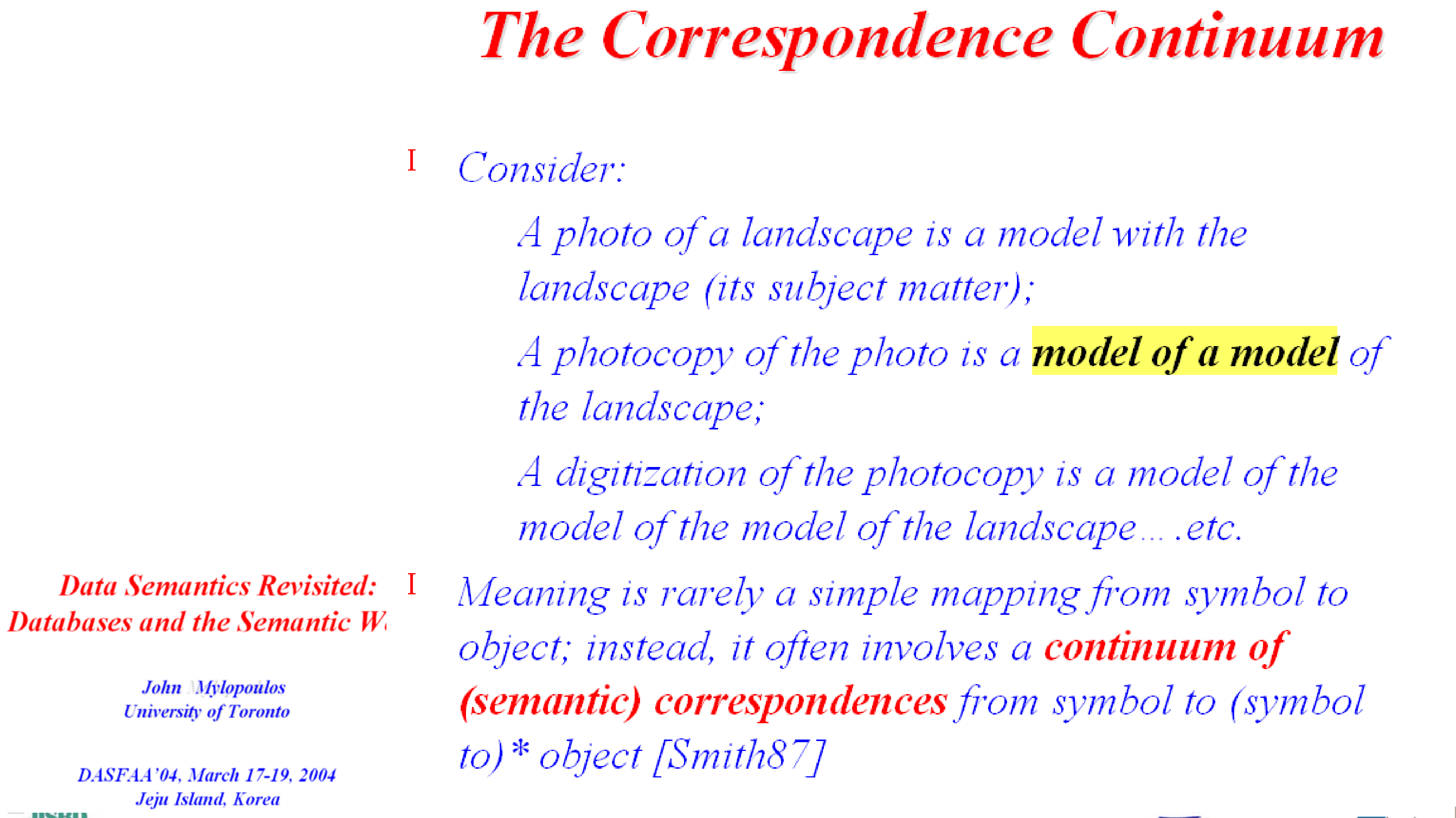
A model of a model





```
 ENTRANCE SECTION.
 P-BEGIN.
*
     ACCEPT W-DATE1 FROM DATE.
     ACCEPT W-DATE  FROM DATE.
     MOVE SPACE TO ZZ-PARAM.
     MOVE ZERO  TO Z-RET I-CALL.
     MOVE ZERO  TO FIND-FLAG.
*======================================
*    MODULE DE CONTROLE DE Z-PARAM.
*======================================
 Z-PARAM-CONTROL SECTION.
*
*        *-------------------------------
*        *        EXISTENCE DE LA TABL
*        *       VALIDITE DE LA FONCTI
*        *-------------------------------
*
 P-TABLE.
*
     MOVE "P-TABLE" TO W-PAR.
     IF (Z-TAB = SPACE OR = LOW-VALUE
        GO TO P-EXIT-PGM.
     MOVE Z-TAB TO BDTACO.
     MOVE ZERO  TO FIND-FLAG.
     FIND ANY BDTA.
     IF OK GET BDTA.
     IF INEX MOVE "01" TO Z-RET
        GO TO P-EXIT-PGM.
```

Model of a model



The Correspondence Continuum

- Consider:
  - A photo of a landscape is a model with the landscape (its subject matter);
  - A photocopy of the photo is a **model of a model** of the landscape;
  - A digitization of the photocopy is a model of the model of the model of the landscape… .etc.
- Meaning is rarely a simple mapping from symbol to object; instead, it often involves a **continuum of (semantic) correspondences** from symbol to (symbol to)* object [Smith87]

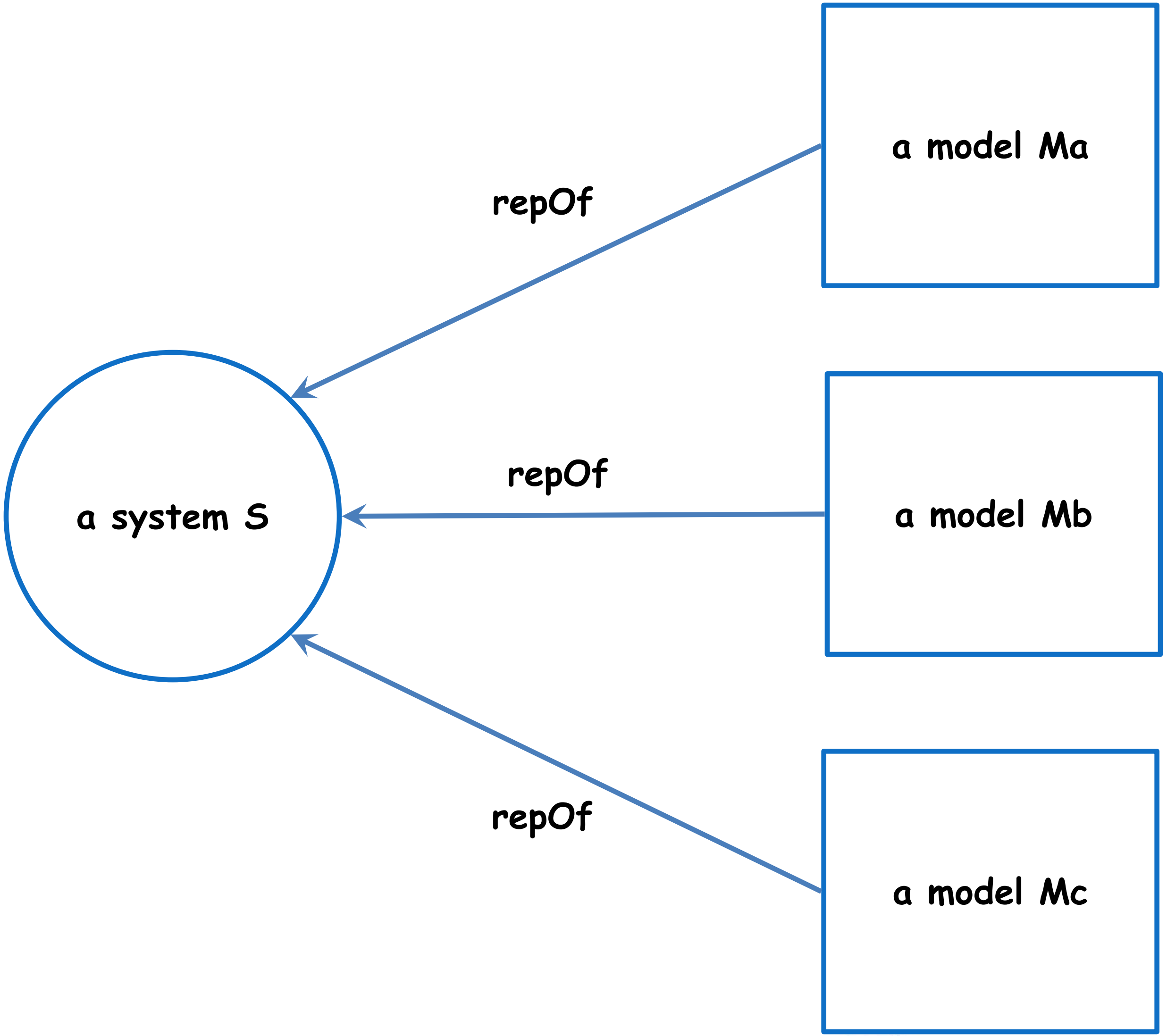
Data Semantics Revisited: Databases and the Semantic W

John Mylopoulos
University of Toronto

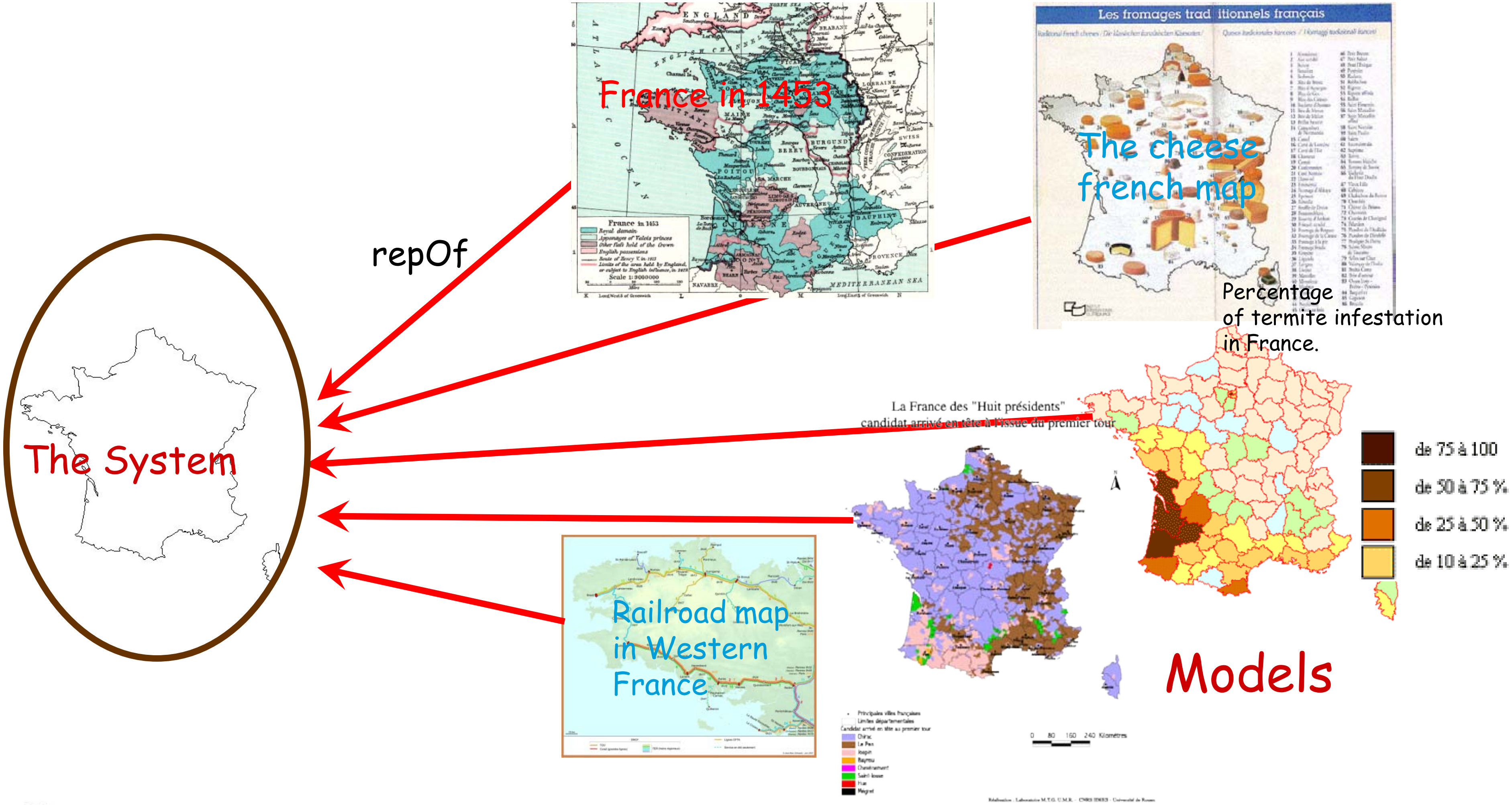
DASFAA'04, March 17-19, 2004
Jeju Island, Korea

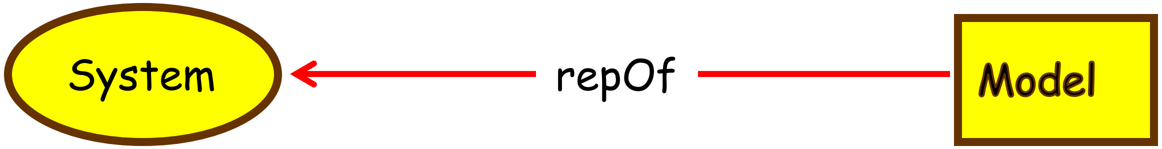
Multimodeling

- ✓ Multimodeling is the joint exploitation of different models representing the same system.
- ✓ These models usually conform to different metamodels.
- ✓ Multimodeling suggests to manage complex systems by collaborative reasoning based on multiple models, each one encompassing a specific type of knowledge (e.g. structural, behavioral, functional) and representation.

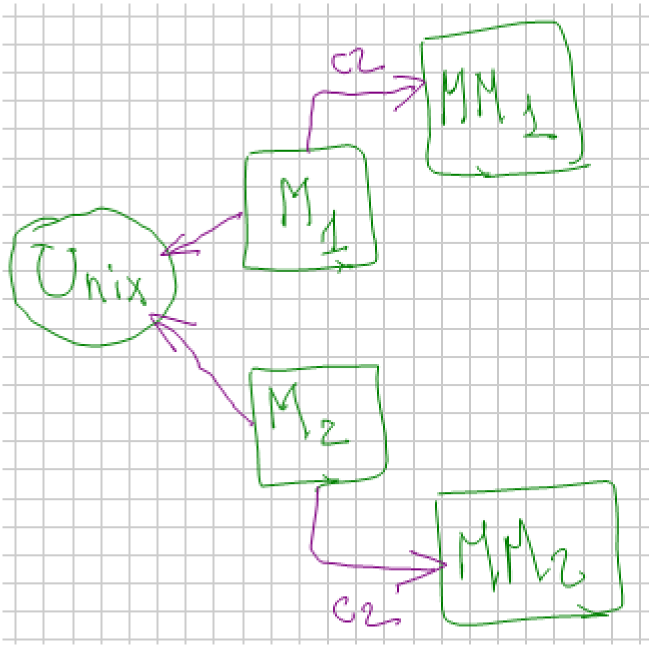


Naïve illustration of multimodeling

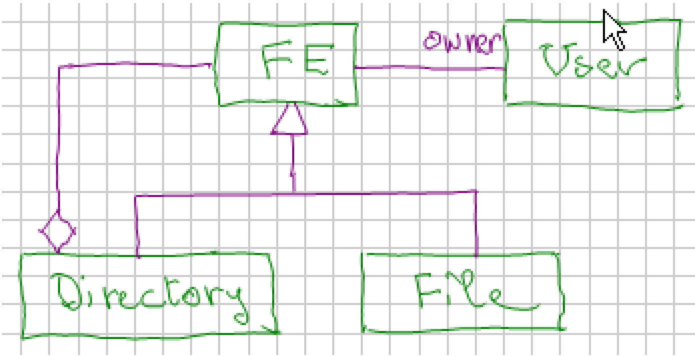

France in 1453
The cheese french map
Les fromages traditionnels français
repOf
Percentage of termite infestation in France.
La France des "Huit présidents"
candidat arrivé en tête à l'issue du premier tour
de 75 à 100
de 50 à 75 %
de 25 à 50 %
de 10 à 25 %
The System
Railroad map in Western France
Models



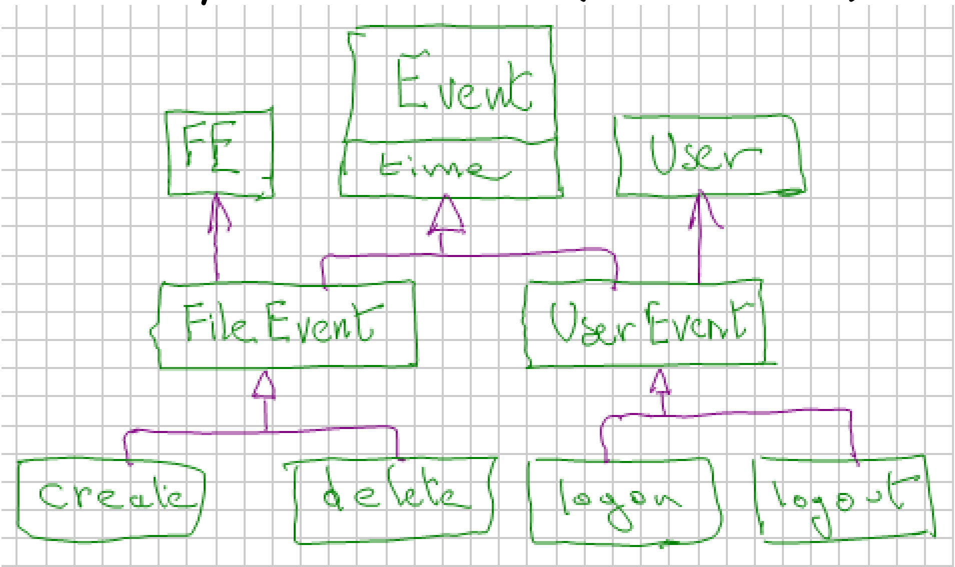
Simple example



MM1: Static Structure

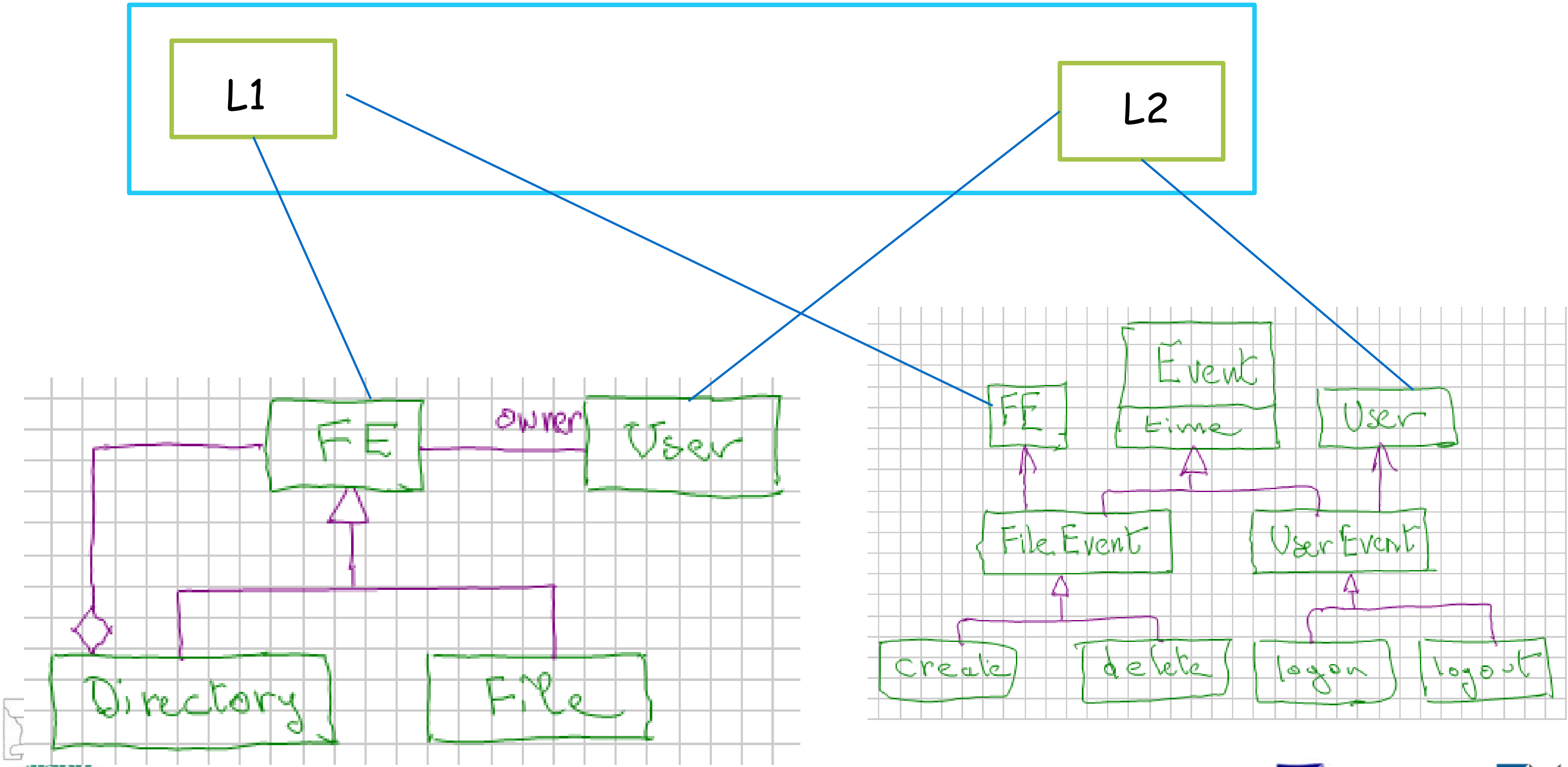


MM2: Dynamic Behavior (event trace)



Simple example

MW : Weaving Metamodel

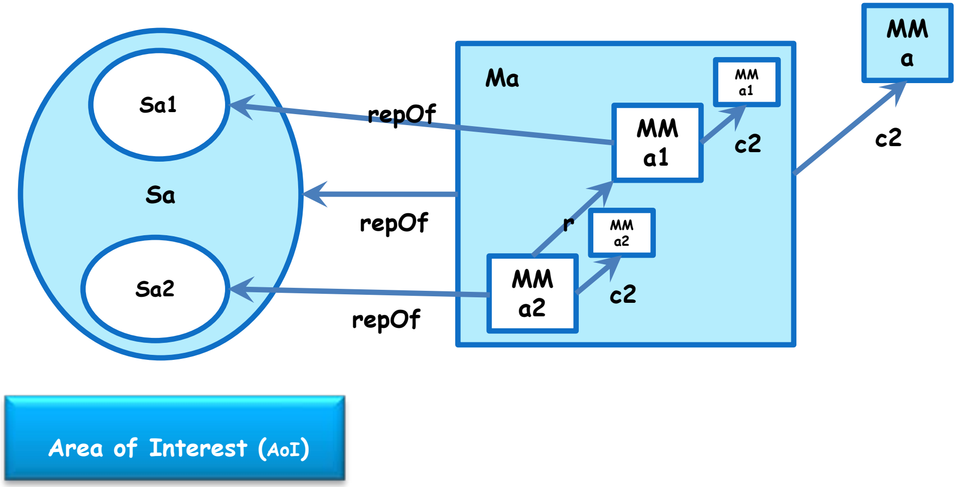


L1

L2

Megamodeling

- With megamodeling, a given model may describe a set of other models and mutual relationships between them.
- Since a megamodel is itself a model, this allows to represent deeply nested systems of systems.



Summary

- Systems
- Models
- Technical Spaces
- Abstract models
- Metamodels
- Metametamodels
- Transformations
- Correspondence (Weaving)
- Megamodels
- Multimodeling

Structural definition of a model

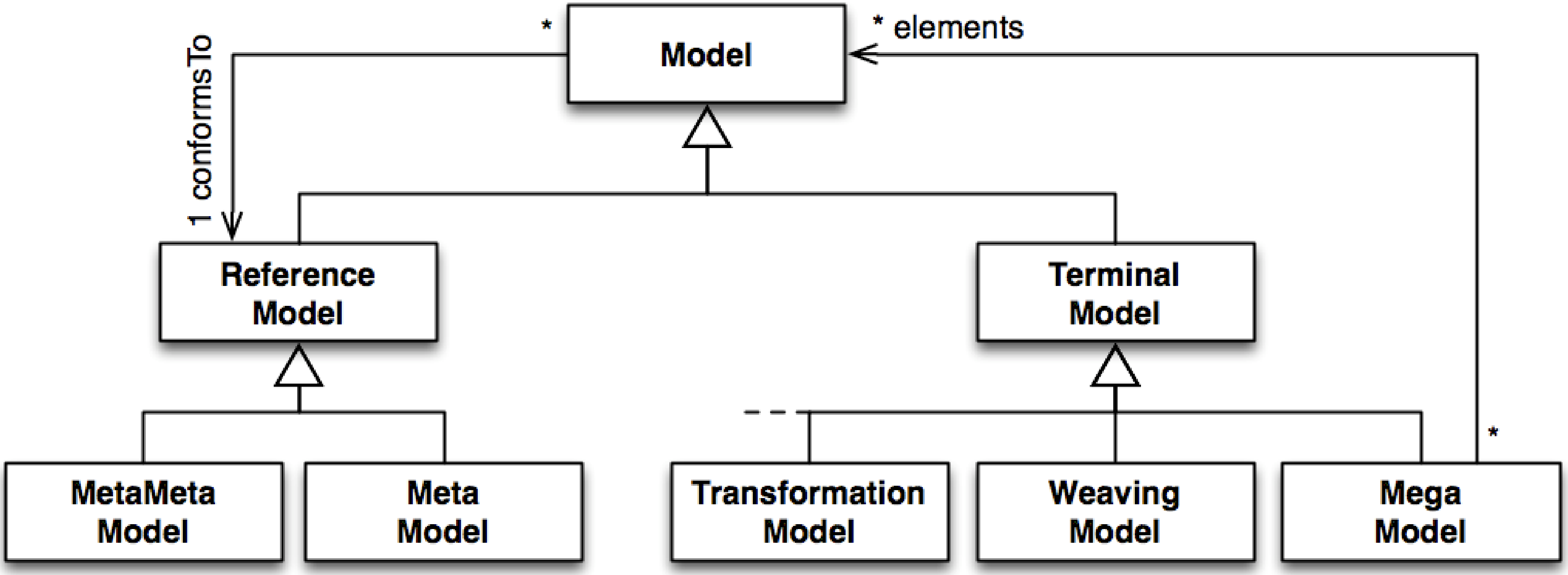
- **Definition 1**. A directed multigraph $G = (N_G, E_G, \Gamma_G)$ consists of a set of distinct nodes N_G, a set of edges E_G and a mapping function $\Gamma_{G:}$: $E_G \rightarrow N_G \times N_G$

- **Definition 2**. A model $M = (G, \omega, \mu)$ is a triple where:
  - ✓ **$G = (N_G, E_G, \Gamma_G)$ is a directed multigraph**
  - ✓ **ω is itself a model, called the reference model of M, associated to a graph $G_\omega = (N_\omega, E_\omega, \Gamma_\omega)$**
  - ✓ **μ: $N_G \cup E_G \rightarrow N_\omega$ is a function associating elements (nodes and edges) of G to nodes of G_ω (metaElements)**

Definitions

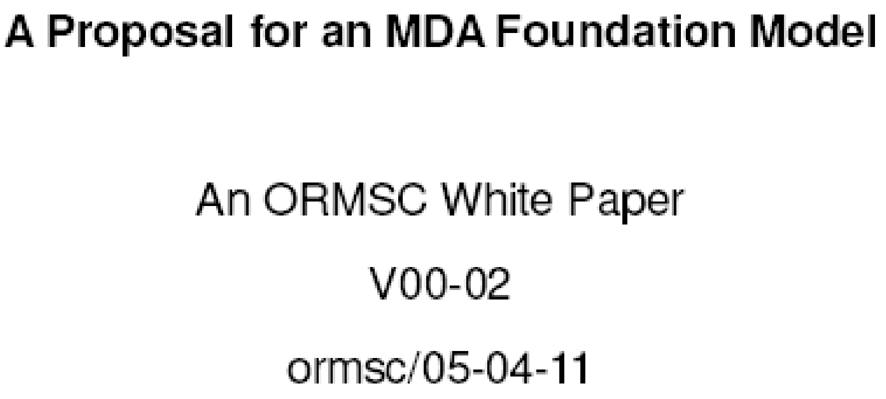
- **Definition 3**. *A* metametamodel is a model that is its own reference model (i.e. it conforms to itself).
- **Definition 4**. *A* metamodel is a model such that its reference model is a metametamodel.
- **Definition 5**. *A* terminal model is a model such that its reference model is a metamodel.

Classification



```
context MetaMetaModel inv: self.conformsTo = self
context MetaModel inv: self.conformsTo.oclIsKindOf(MetaMetaModel)
context TerminalModel inv: self.conformsTo.oclIsKindOf(MetaModel)
```

These definitions are compatible with OMG view

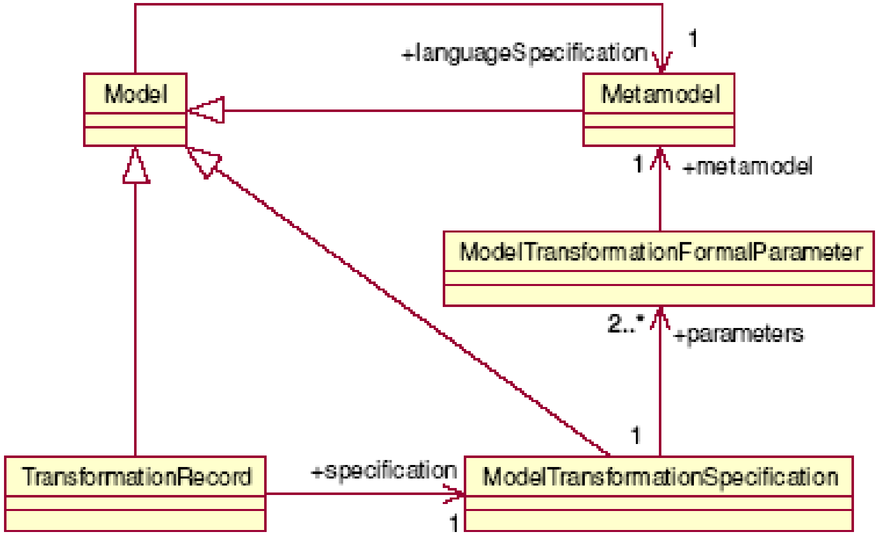


A Proposal for an MDA Foundation Model

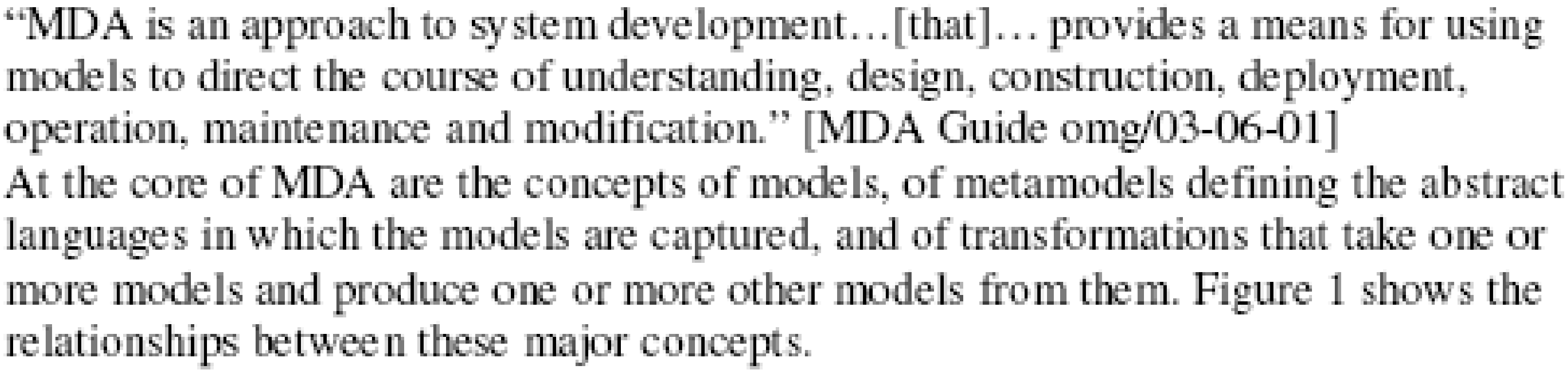
An ORMSC White Paper

V00-02

ormsc/05-04-11



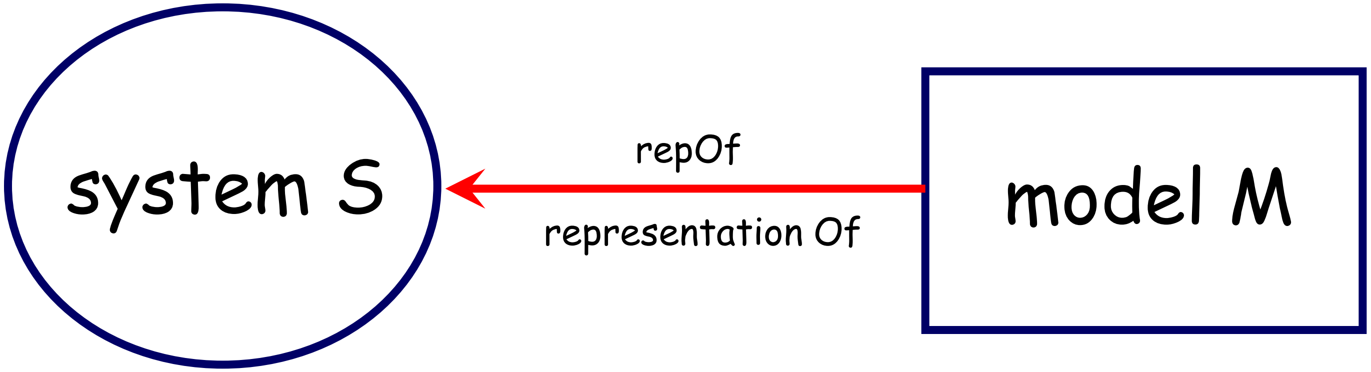
Object Reference Model SubCommittee (ORMSC)
"The MDA guide"



“MDA is an approach to system development…[that]… provides a means for using models to direct the course of understanding, design, construction, deployment, operation, maintenance and modification.” [MDA Guide omg/03-06-01]

At the core of MDA are the concepts of models, of metamodels defining the abstract languages in which the models are captured, and of transformations that take one or more models and produce one or more other models from them. Figure 1 shows the relationships between these major concepts.

Utilization definition

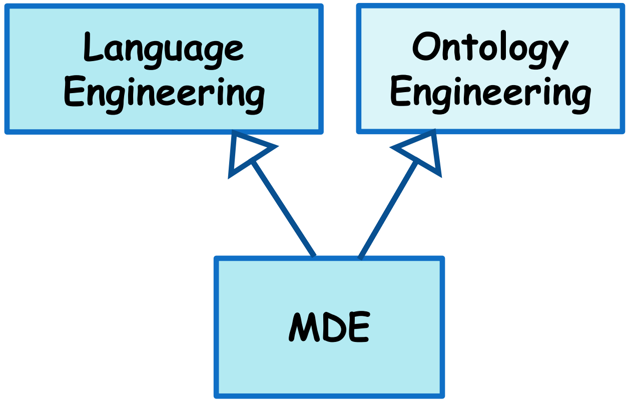


system S

repOf

representation Of

model M



Language Engineering

Ontology Engineering

MDE

After the *language engineering* part (conformsTo),
we also need to cope
with the *ontology engineering* part (representationOf).

This is more difficult.

Utilization definition

The objective here is to define the possible usages of a model. Consequently, in all the present subsection, model will mean "terminal model".

- ✓ **Definition 6**. A system S is a delimited part of the world considered as a set of elements in interaction.
- ✓ **Definition 7**. A model M is a representation of a given system S, satisfying the substitutability principle (see below).
- ✓ **Definition 8**. (Principle of substitutability). A model M is said to be a representation of a system S for a given set of questions Q if, for each question of this set Q, the model M will provide exactly the same answer that the system S would have provided in answering the same question.

Principle of limited substitutability according to Minsky

"If a creature can answer a question about a hypothetical experiment without actually performing it, then it has demonstrated some knowledge about the world. …

We use the term "model" in the following sense: To an observer B, an object A* is a model of an object A to the extent that B can use A* to answer questions that interest him about A. …

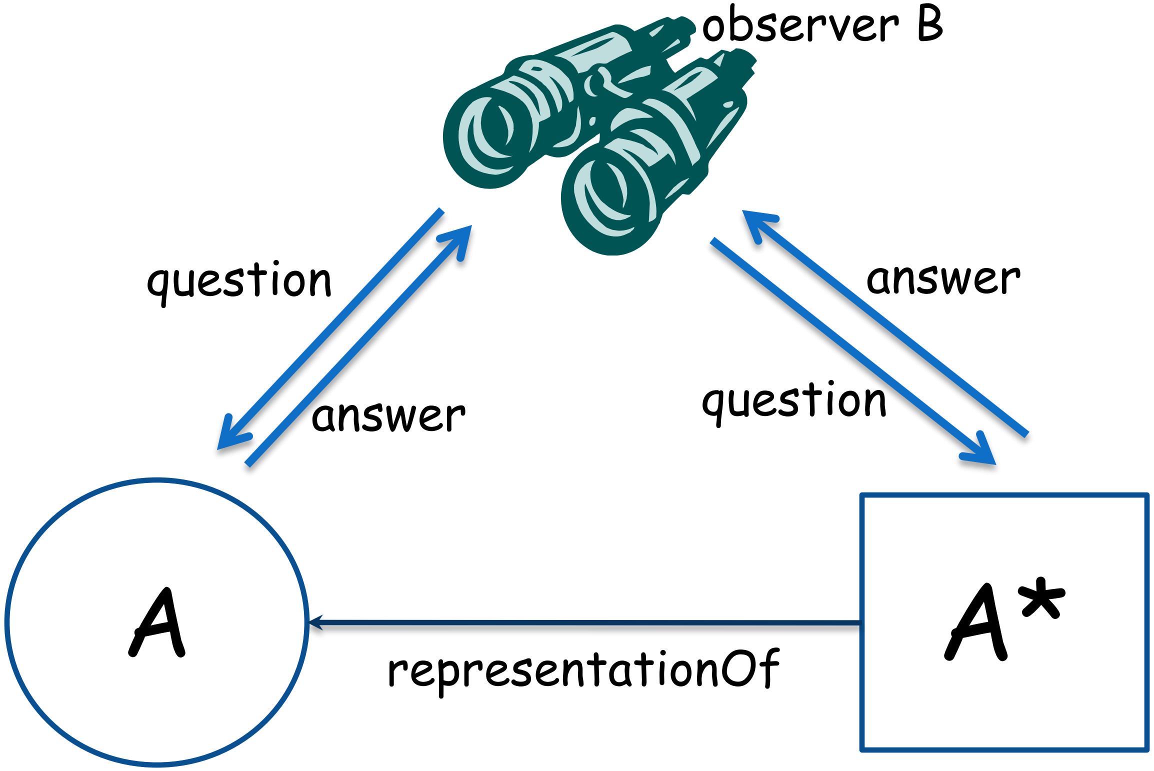
It is understood that B's use of a model entails the use of encodings for input and output, both for A and A*.
If A is the world, questions for A are experiments. ...

A* is a good model of A, in B's view, to the extent that A*'s answers agree with those of A's, on the whole, with respect to the questions important to B. …"

Marvin L. Minsky

Matter, Mind and Models Semantic Information Processing, MIT Press, 1968

Limited substitutability

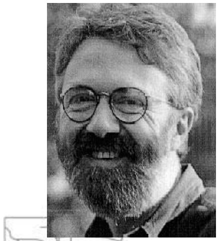


We use the term "model" in the following sense: To an observer B, an object A* is a model of an object A to the extent that B can use A* to answer questions that interest him about A.

Taking the representation relation seriously

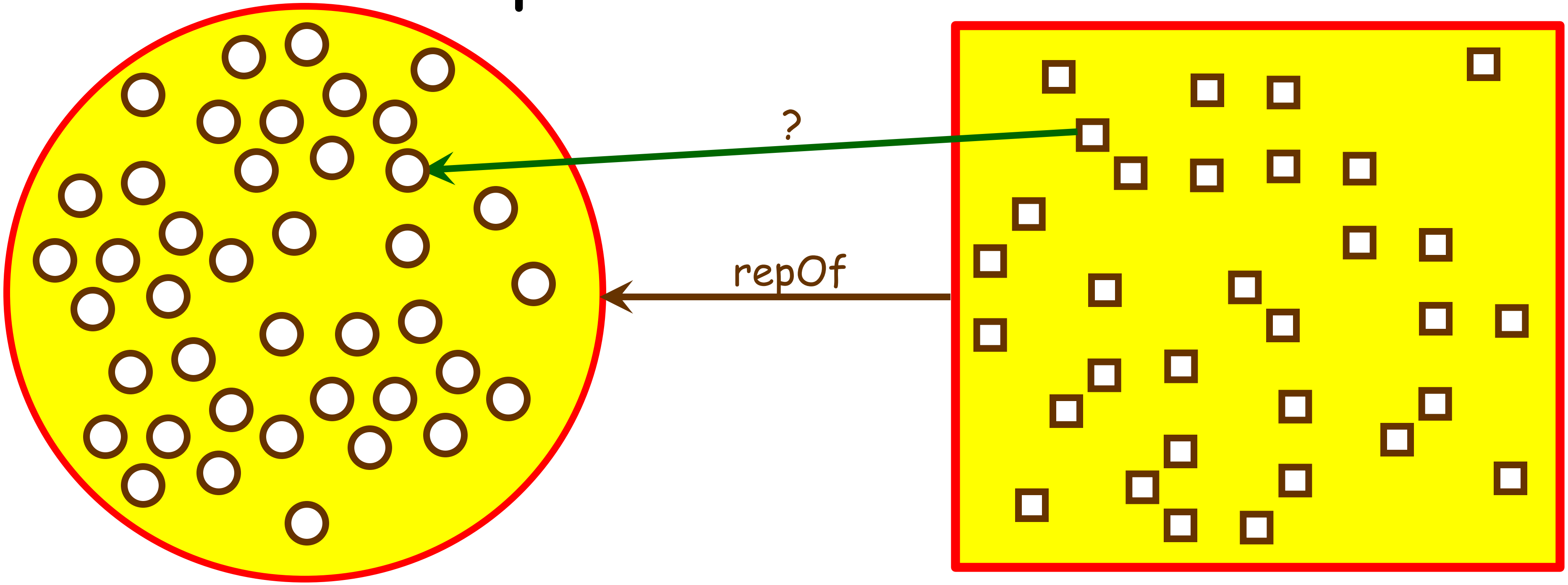
> " *What about the [relationship between model and real-world]? The answer, and one of the main points I hope you will take away from this discussion, is that, at this point in intellectual history, we have no theory of this [...] relationship*".

Brian Cantwell Smith **The Limits of Correctness**;
a paper prepared for the Symposium on Unintentional Nuclear War,
Fifth Congress of the International Physicians for the Prevention of
Nuclear War, Budapest, Hungary, June 28 – July 1 1985.



See also "on the origin of objects"

The "representation" relation



System and System elements
(after discretisation)

Model and Model elements

Simple set interpretation of the *repOf* relation is probably as correct as simple set interpretation of the *instanceOf* relation in object technology.

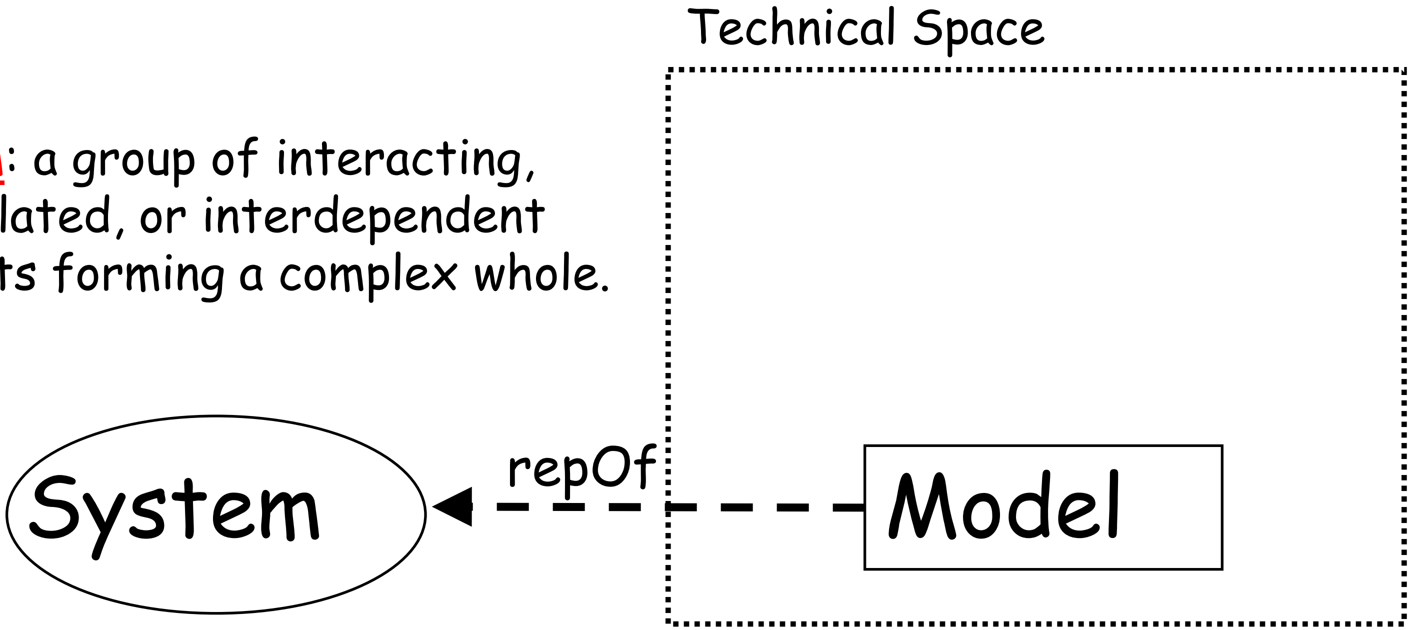
Where are models coming from?

TECHNICAL SPACES

Basic entities

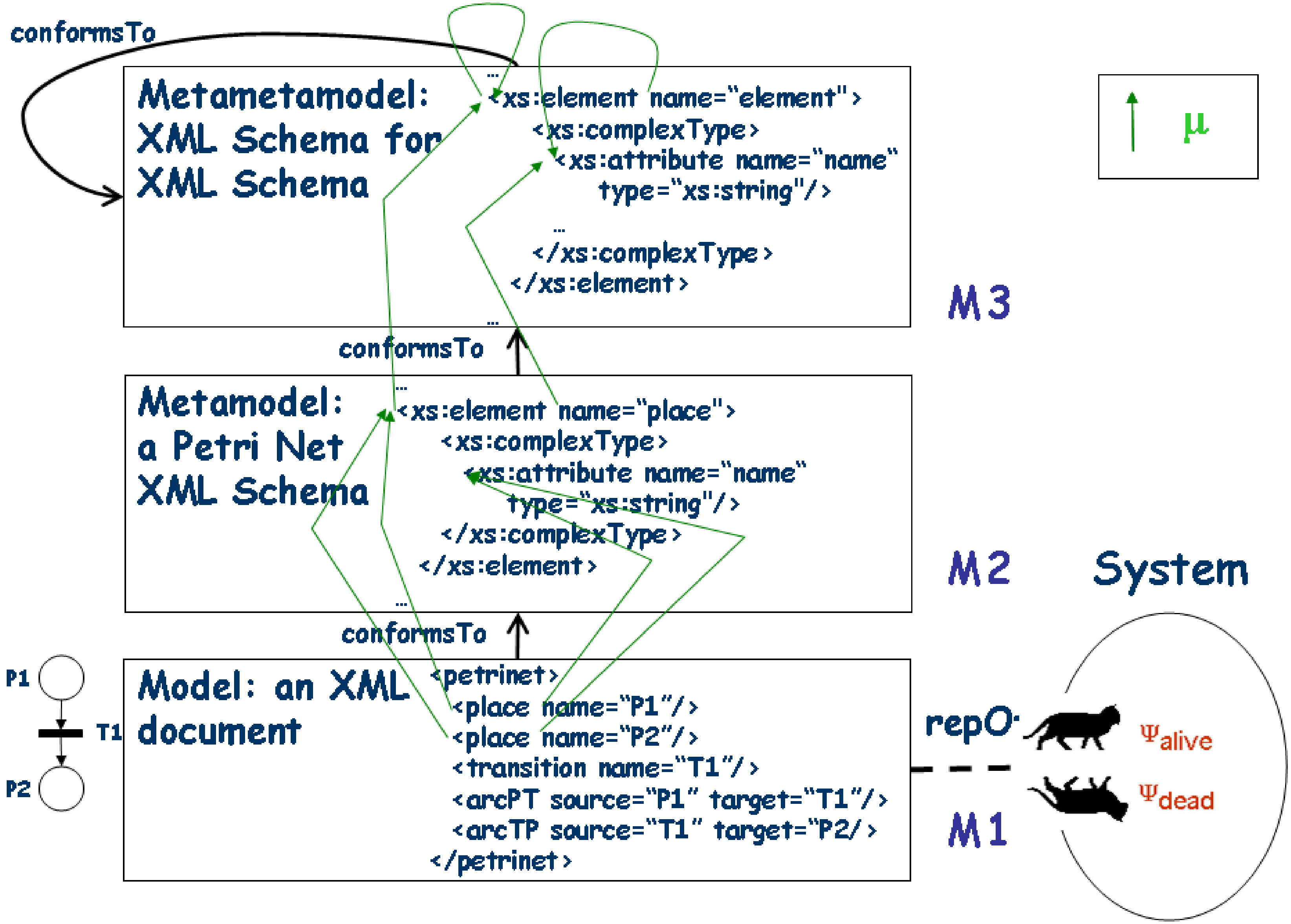
Technical Space: a model management framework usually based on some algebraic structures (trees, graphs,hypergraphs, etc.).

System: a group of interacting, interrelated, or interdependent elements forming a complex whole.



Model: an abstract representation of a system created for a specific purpose.

XML Technical Space



conformsTo

Metametamodel: XML Schema for XML Schema

```
…
<xs:element name="element">
  <xs:complexType>
    <xs:attribute name="name"
       type="xs:string"/>
    …
  </xs:complexType>
</xs:element>
…
```

M3

conformsTo

Metamodel: a Petri Net XML Schema

```
…
<xs:element name="place">
  <xs:complexType>
    <xs:attribute name="name"
       type="xs:string"/>
  </xs:complexType>
</xs:element>
…
```

M2

System

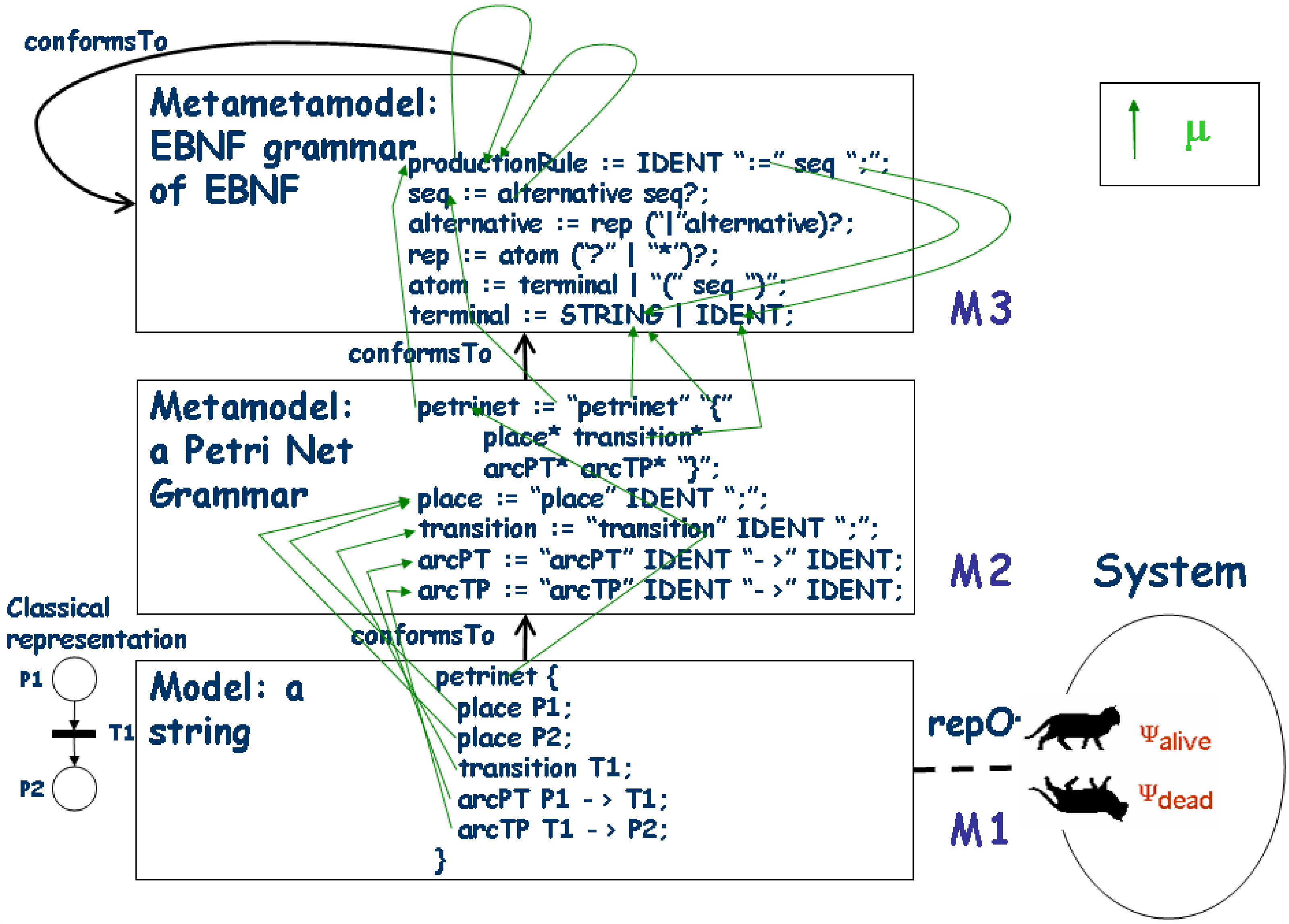
conformsTo

Model: an XML document

```
<petrinet>
  <place name="P1"/>
  <place name="P2"/>
  <transition name="T1"/>
  <arcPT source="P1" target="T1"/>
  <arcTP source="T1" target="P2/>
</petrinet>
```

M1

EBNF Technical Space



conformsTo

Metametamodel:
EBNF grammar
of EBNF

```
productionRule := IDENT ":=" seq ";";
seq := alternative seq?;
alternative := rep ("|"alternative)?;
rep := atom ("?" | "*")?;
atom := terminal | "(" seq ")";
terminal := STRING | IDENT;
```

M3

↑ µ

conformsTo

Metamodel:
a Petri Net
Grammar

```
petrinet := "petrinet" "{"
        place* transition*
        arcPT* arcTP* "}";
place := "place" IDENT ";";
transition := "transition" IDENT ";";
arcPT := "arcPT" IDENT "->" IDENT;
arcTP := "arcTP" IDENT "->" IDENT;
```

M2

System

Classical
representation

conformsTo

Model: a
string

```
petrinet {
  place P1;
  place P2;
  transition T1;
  arcPT P1 -> T1;
  arcTP T1 -> P2;
}
```

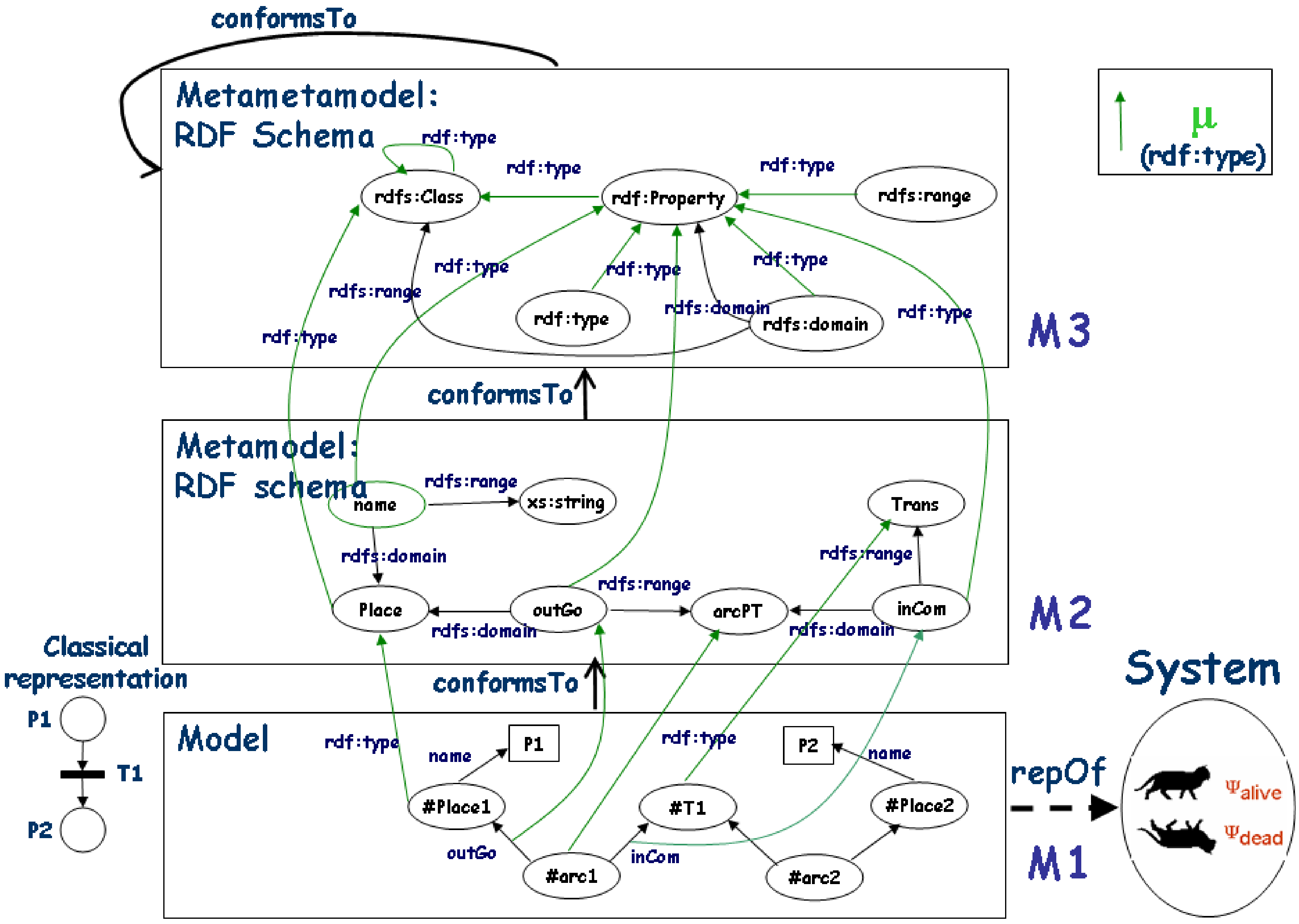
repOf

Ψalive

Ψdead

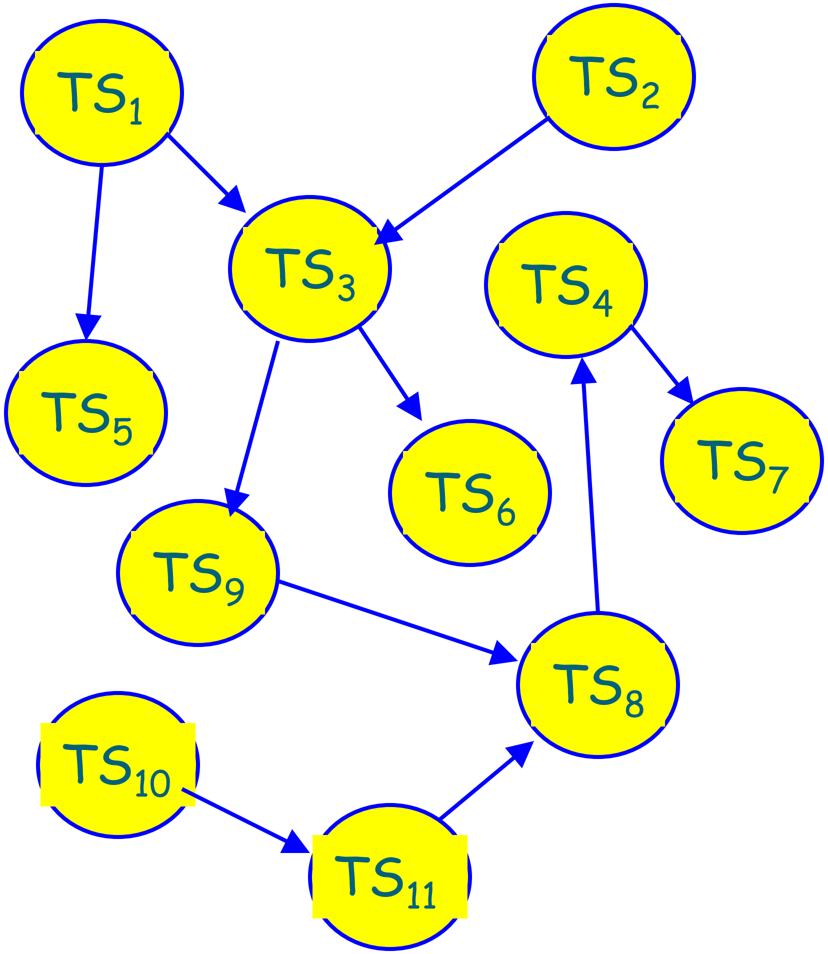
M1

RDF Technical Space



The influence map

- No technology is an island
- No technology is uniformly superior to others
- Technologies are active and evolving
  - Even if sometimes they may stay idle for long periods (expert systems, etc.)
- Technologies never die: they just hide in deep software layers
  - e.g. RPG, Cobol, etc.
  - Today edge cut technologies are tomorrow legacy
- Technologies are mutually influencing



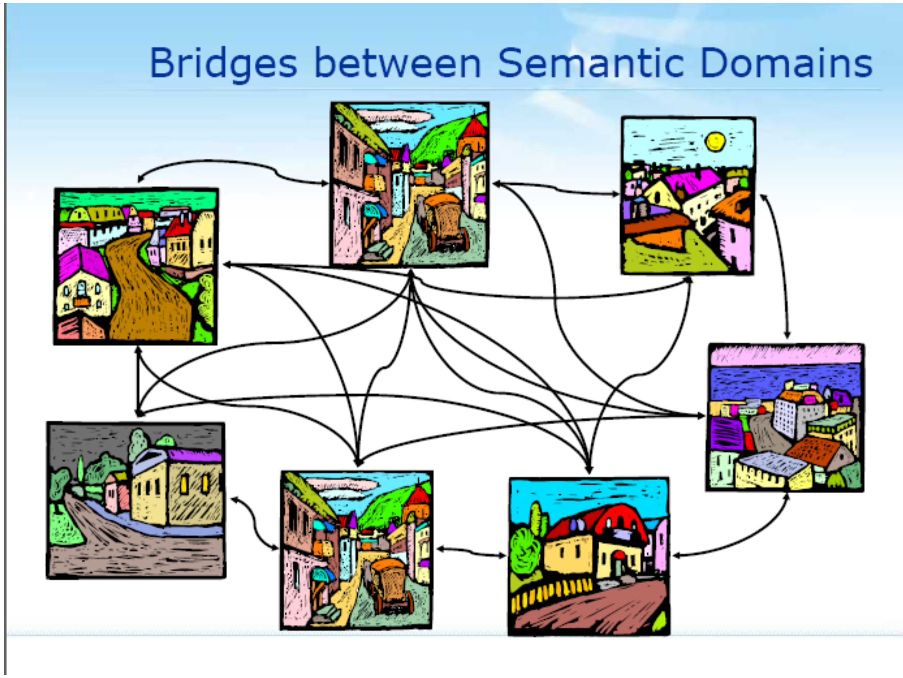
Comparing spaces

	XML	MDA	Grammarware	Ontologies
Executability	Poor	Poor	Excellent	Poor
Aspects	Good	Excellent	Poor	Fair
Formalization	Poor	Poor	Excellent	Fair
Specialization	Fair	Good	Poor	Fair
Modularity	Fair	Fair	Poor	Fair
Traceability	Good	Fair	Poor	Excellent
Transformability	Excellent	Fair	Fair	Fair

(NB: marks are indicative)

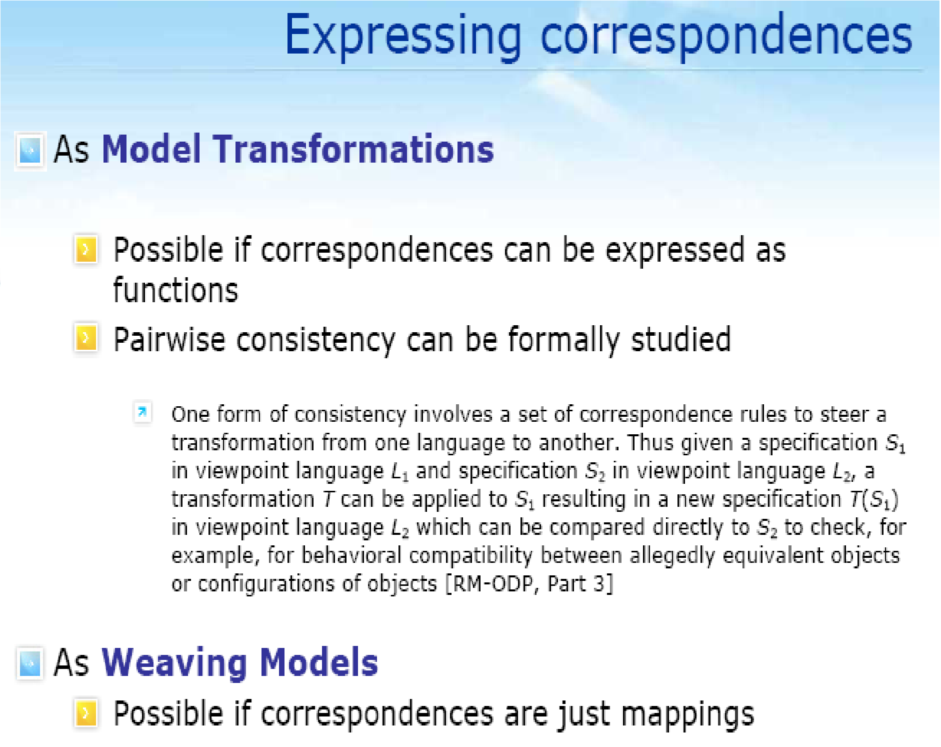
+ stability in time

The «Village metaphor» by Antonio Vallecillo



Bridges between Semantic Domains

The Prolog village
The Petri net village
The Coloured Petri Net Village
The Z village
The B village
The Maude village
The Coq village
etc.



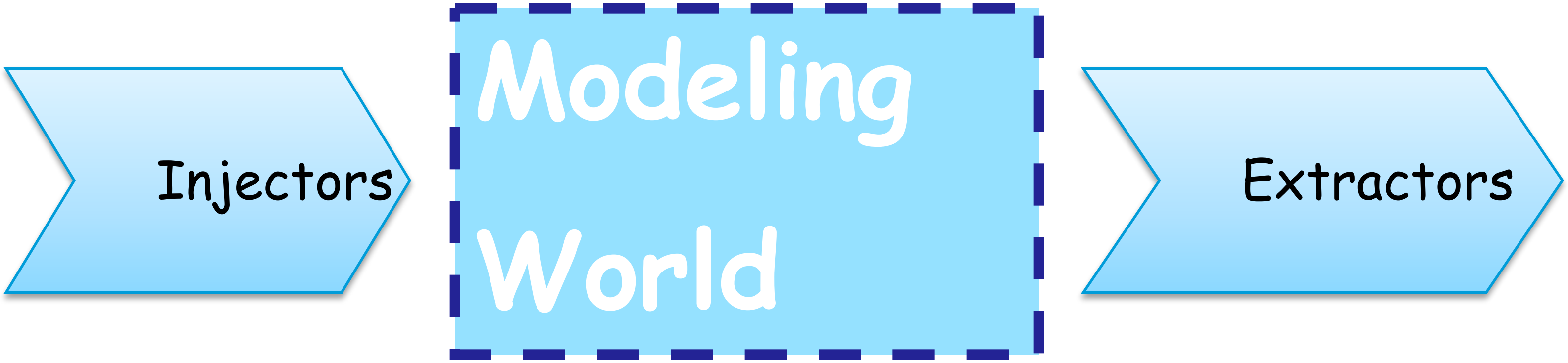
Expressing correspondences

- As **Model Transformations**
  - Possible if correspondences can be expressed as functions
  - Pairwise consistency can be formally studied
    - One form of consistency involves a set of correspondence rules to steer a transformation from one language to another. Thus given a specification S_1 in viewpoint language L_1 and specification S_2 in viewpoint language L_2, a transformation T can be applied to S_1 resulting in a new specification $T(S_1)$ in viewpoint language L_2 which can be compared directly to S_2 to check, for example, for behavioral compatibility between allegedly equivalent objects or configurations of objects [RM-ODP, Part 3]
- As **Weaving Models**
  - Possible if correspondences are just mappings

Communications between technical spaces

Any software artifact can be **injected** into a model

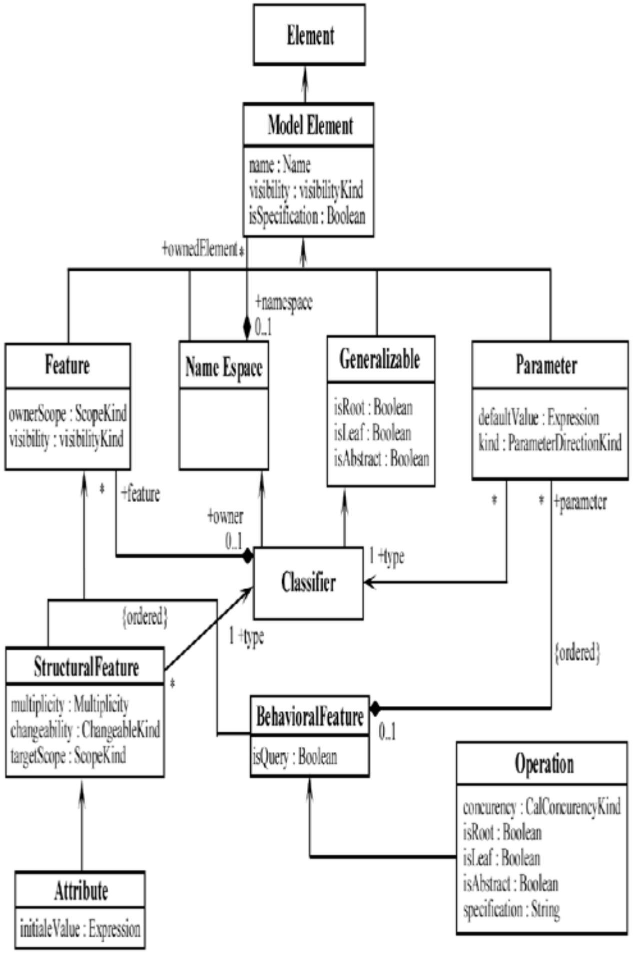
Any model can be **extracted** to a software artifact



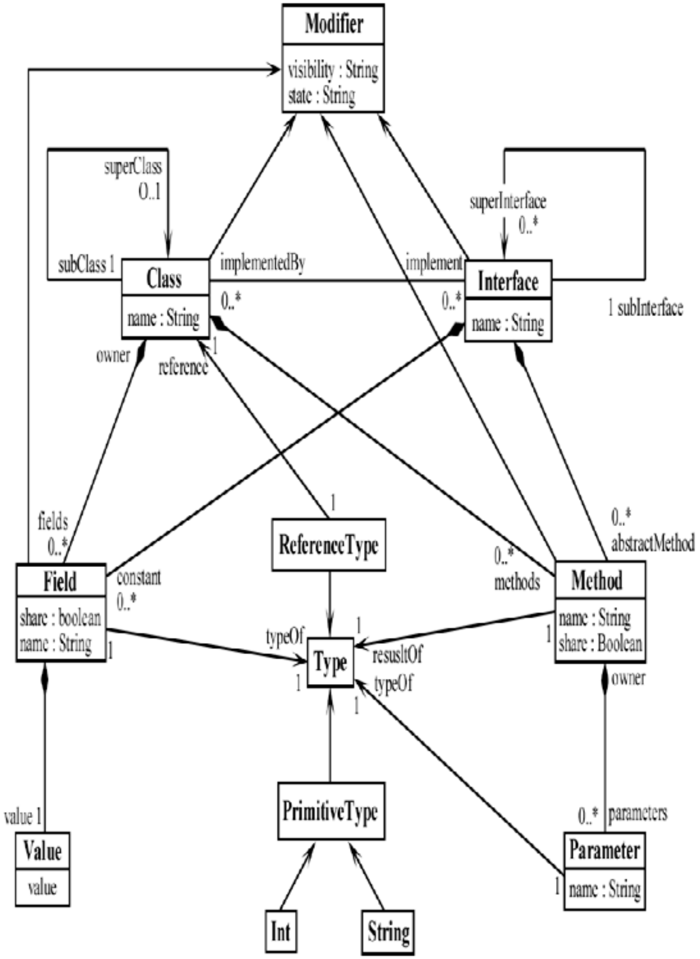
Cost of solving a problem inside a TS vs. Exporting it to another TS, solving it and importing back the solution.

APPLICATIONS

Example: UML to Java



UML Metamodel



Java Metamodel

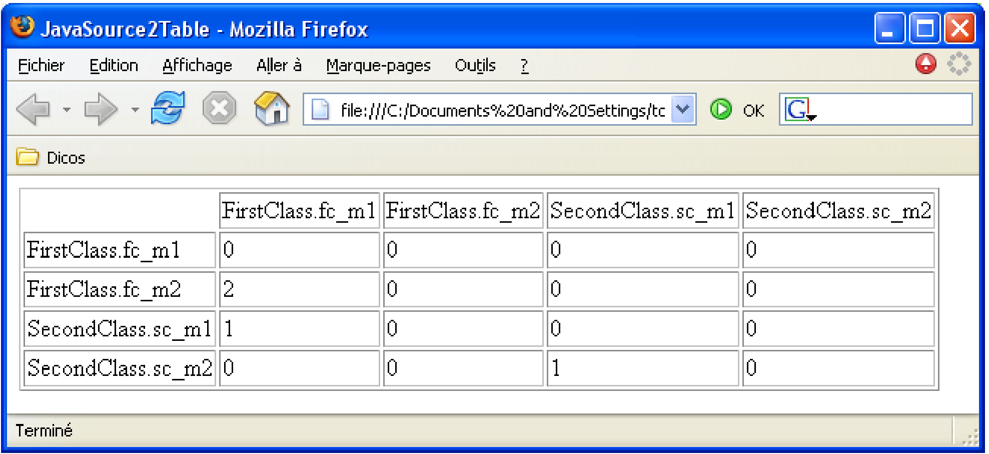
Example : Java to Excel

FirstClass.java

```
public class
    FirstClass {
  public void
    fc_m1(){
  }
  public void
    fc_m2(){
    this.fc_m1();
    this.fc_m1();
  }
}
```

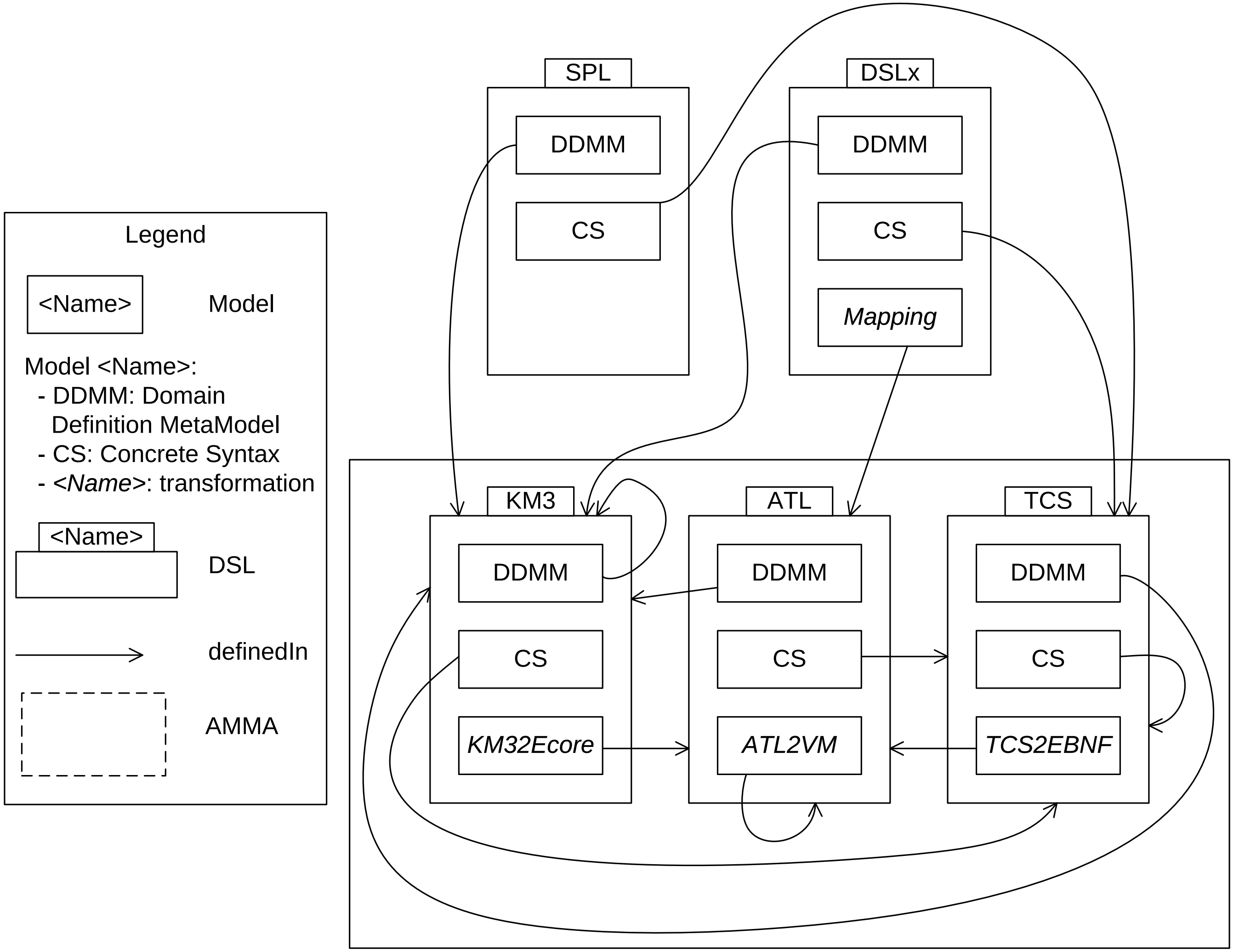
SecondClasss.java

```
public class SecondClass
    {
  public void sc_m1(){
    FirstClass a = new
    FirstClass();
    a.fc_m1();
  }
  public void sc_m2(){
    this.sc_m1();
  }
}
```


JavaSource2Table - Mozilla Firefox
Fichier
Edition
Affichage
Aller à
Marque-pages
Outils
?
file:///C:/Documents%20and%20Settings/tc
OK
Dicos
Terminé

	FirstClass.fc_m1	FirstClass.fc_m2	SecondClass.sc_m1	SecondClass.sc_m2
FirstClass.fc_m1	0	0	0	0
FirstClass.fc_m2	2	0	0	0
SecondClass.sc_m1	1	0	0	0
SecondClass.sc_m2	0	0	1	0

Example: SPL to CPL



Example: XSLT to XQuery

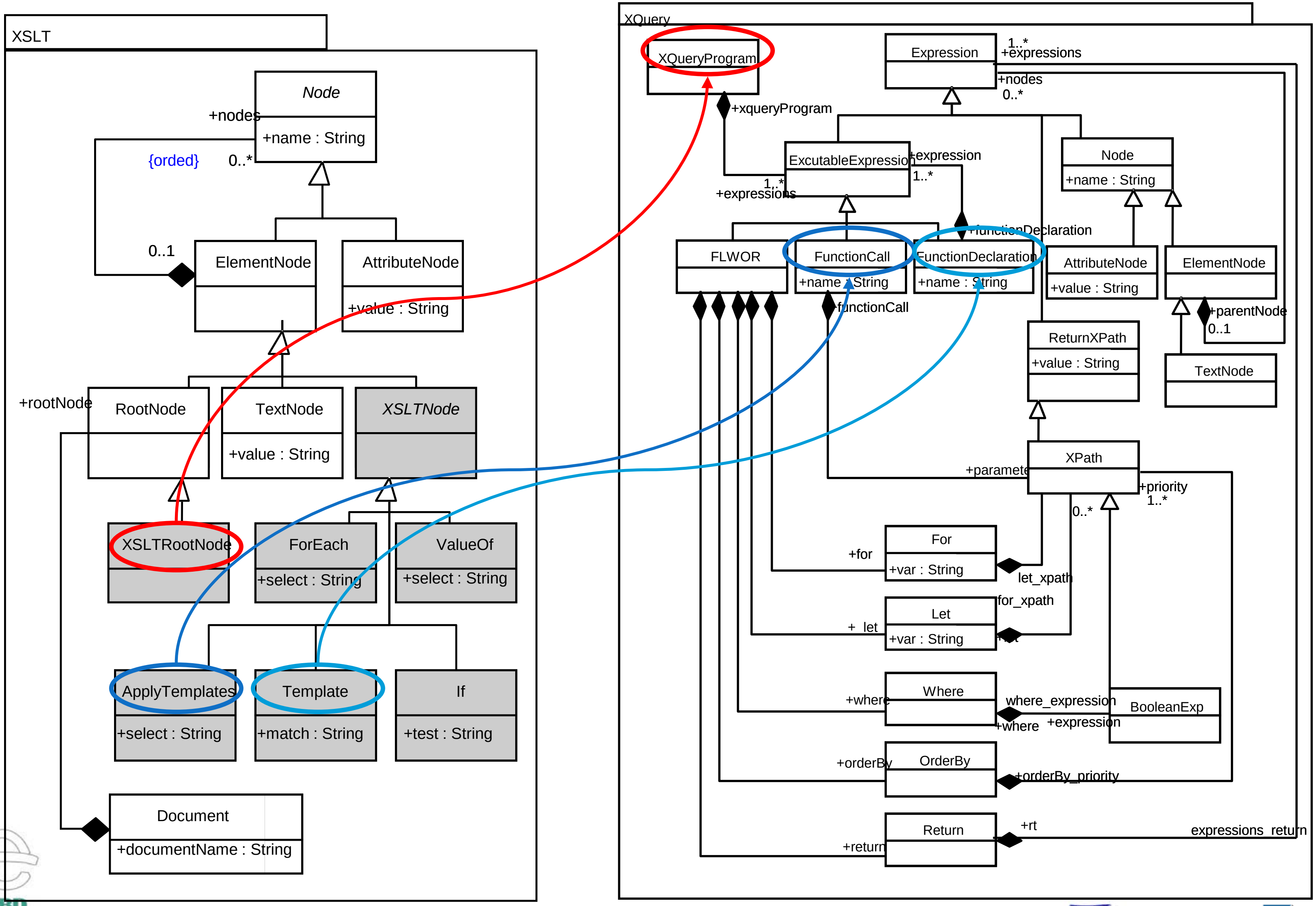
XSLT

```
<xsl:stylesheet […] >
    <xsl:template match="/">
          <emps>
            <xsl:apply-templates
    select="employee"/>
          </emps>
    </xsl:template>
    <xsl:template match="employee">
      <xsl:if test="salary&gt;2000">
        <emp>
          <xsl:value-of select="name"/>
          <xsl:value-of
    select="firstname"/>
        </emp>
      </xsl:if>
    </xsl:template>
</xsl:stylesheet>
```

XQuery

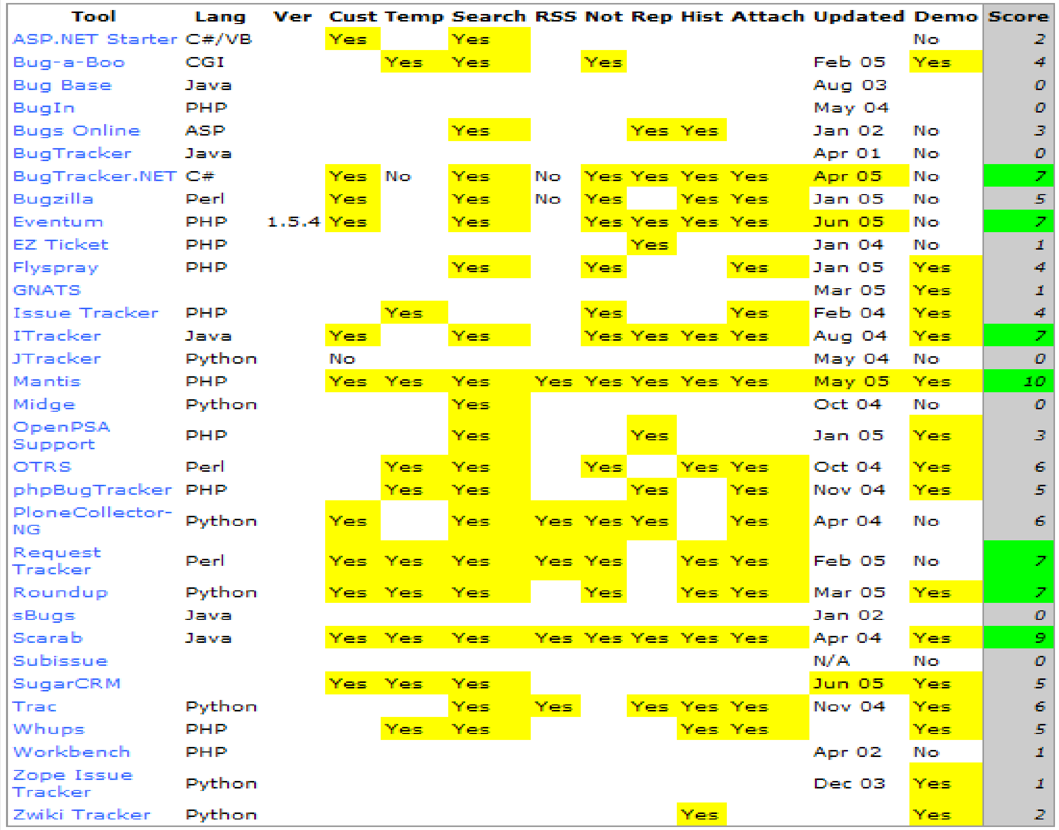
```
define function fctemployee($paramVar)
{
  for $var in $paramVar
  return
    let $var := $var
    where $var/salary>2000
    return

     <emp>{$var/name}{$var/firstname}</emp
     >
}
for $var in document("xmlFile.xml")/*
return

     <emps>{fctemployee($var/employee)}</e
     mps>
```

Metamodel-driven transformation in ATL: XSLT & XQuery metamodels



Example: Bugzilla to Mantis

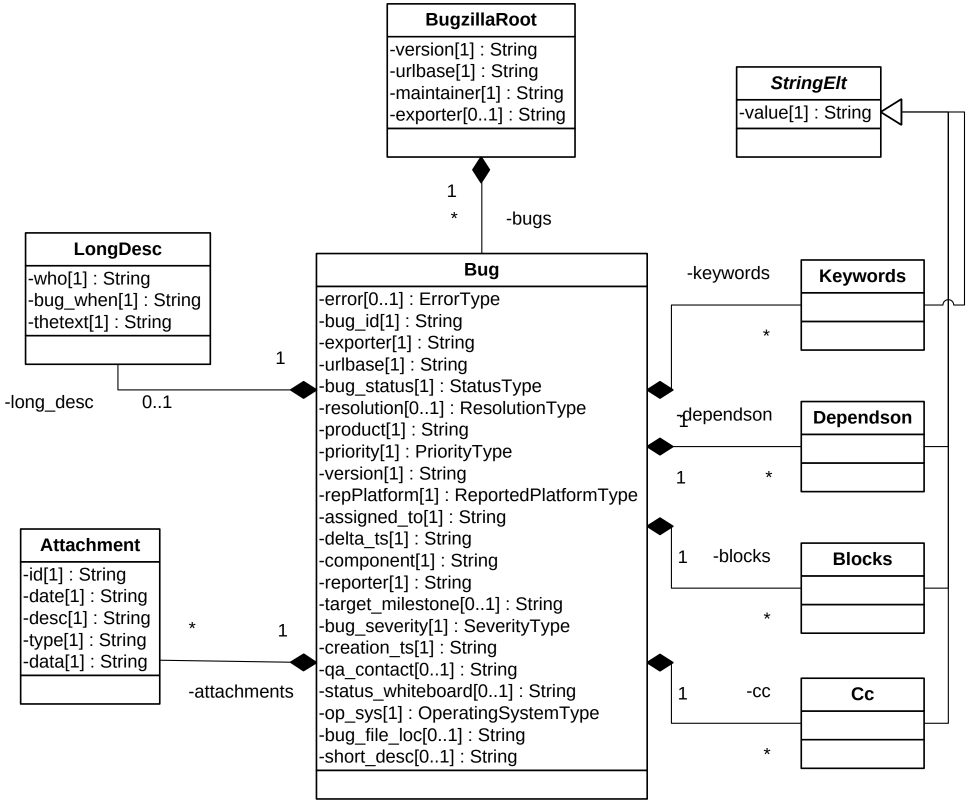
About 30 open source tools to choose among for bug tracking



Tool	Lang	Ver	Cust	Temp	Search	RSS	Not	Rep	Hist	Attach	Updated	Demo	Score
ASP.NET Starter	C#/VB		Yes		Yes							No	2
Bug-a-Boo	CGI			Yes	Yes		Yes				Feb 05	Yes	4
Bug Base	Java										Aug 03		0
BugIn	PHP										May 04		0
Bugs Online	ASP				Yes			Yes	Yes		Jan 02	No	3
BugTracker	Java										Apr 01	No	0
BugTracker.NET	C#		Yes	No	Yes	No	Yes	Yes	Yes	Yes	Apr 05	No	7
Bugzilla	Perl		Yes		Yes	No	Yes		Yes	Yes	Jan 05	No	5
Eventum	PHP	1.5.4	Yes		Yes		Yes	Yes	Yes	Yes	Jun 05	No	7
EZ Ticket	PHP							Yes			Jan 04	No	1
Flyspray	PHP				Yes		Yes			Yes	Jan 05	Yes	4
GNATS											Mar 05	Yes	1
Issue Tracker	PHP			Yes			Yes			Yes	Feb 04	Yes	4
ITracker	Java		Yes		Yes		Yes	Yes	Yes	Yes	Aug 04	Yes	7
JTracker	Python		No								May 04	No	0
Mantis	PHP		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	May 05	Yes	10
Midge	Python				Yes						Oct 04	No	0
OpenPSA Support	PHP				Yes			Yes			Jan 05	Yes	3
OTRS	Perl			Yes	Yes		Yes		Yes	Yes	Oct 04	Yes	6
phpBugTracker	PHP			Yes	Yes			Yes		Yes	Nov 04	Yes	5
PloneCollector-NG	Python		Yes		Yes	Yes	Yes	Yes		Yes	Apr 04	No	6
Request Tracker	Perl		Yes	Yes	Yes	Yes	Yes		Yes	Yes	Feb 05	No	7
Roundup	Python		Yes	Yes	Yes		Yes		Yes	Yes	Mar 05	Yes	7
sBugs	Java										Jan 02		0
Scarab	Java		Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Apr 04	Yes	9
Subissue											N/A	No	0
SugarCRM			Yes	Yes	Yes						Jun 05	Yes	5
Trac	Python				Yes	Yes		Yes	Yes	Yes	Nov 04	Yes	6
Whups	PHP			Yes	Yes				Yes	Yes		Yes	5
Workbench	PHP										Apr 02	No	1
Zope Issue Tracker	Python										Dec 03	Yes	1
Zwiki Tracker	Python								Yes			Yes	2

Different data models and functionalities

Bugzilla metamodel (simplified)



BugzillaRoot
-version[1] : String
-urlbase[1] : String
-maintainer[1] : String
-exporter[0..1] : String

StringElt
-value[1] : String

LongDesc
-who[1] : String
-bug_when[1] : String
-thetext[1] : String

Bug
-error[0..1] : ErrorType
-bug_id[1] : String
-exporter[1] : String
-urlbase[1] : String
-bug_status[1] : StatusType
-resolution[0..1] : ResolutionType
-product[1] : String
-priority[1] : PriorityType
-version[1] : String
-repPlatform[1] : ReportedPlatformType
-assigned_to[1] : String
-delta_ts[1] : String
-component[1] : String
-reporter[1] : String
-target_milestone[0..1] : String
-bug_severity[1] : SeverityType
-creation_ts[1] : String
-qa_contact[0..1] : String
-status_whiteboard[0..1] : String
-op_sys[1] : OperatingSystemType
-bug_file_loc[0..1] : String
-short_desc[0..1] : String

Attachment
-id[1] : String
-date[1] : String
-desc[1] : String
-type[1] : String
-data[1] : String

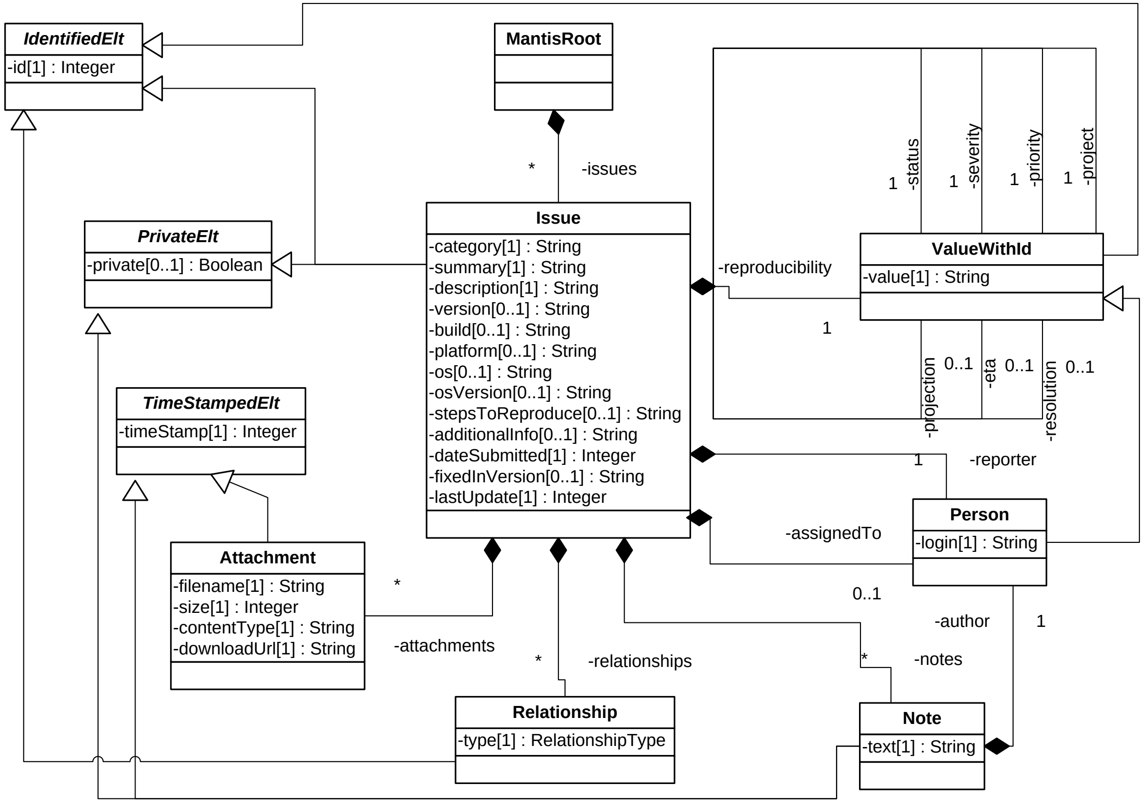
Keywords

Dependson

Blocks

Cc

Mantis metamodel (simplified)



IdentifiedElt
-id[1] : Integer

MantisRoot

PrivateElt
-private[0..1] : Boolean

TimeStampedElt
-timeStamp[1] : Integer

Issue
-category[1] : String
-summary[1] : String
-description[1] : String
-version[0..1] : String
-build[0..1] : String
-platform[0..1] : String
-os[0..1] : String
-osVersion[0..1] : String
-stepsToReproduce[0..1] : String
-additionalInfo[0..1] : String
-dateSubmitted[1] : Integer
-fixedInVersion[0..1] : String
-lastUpdate[1] : Integer

ValueWithId
-value[1] : String

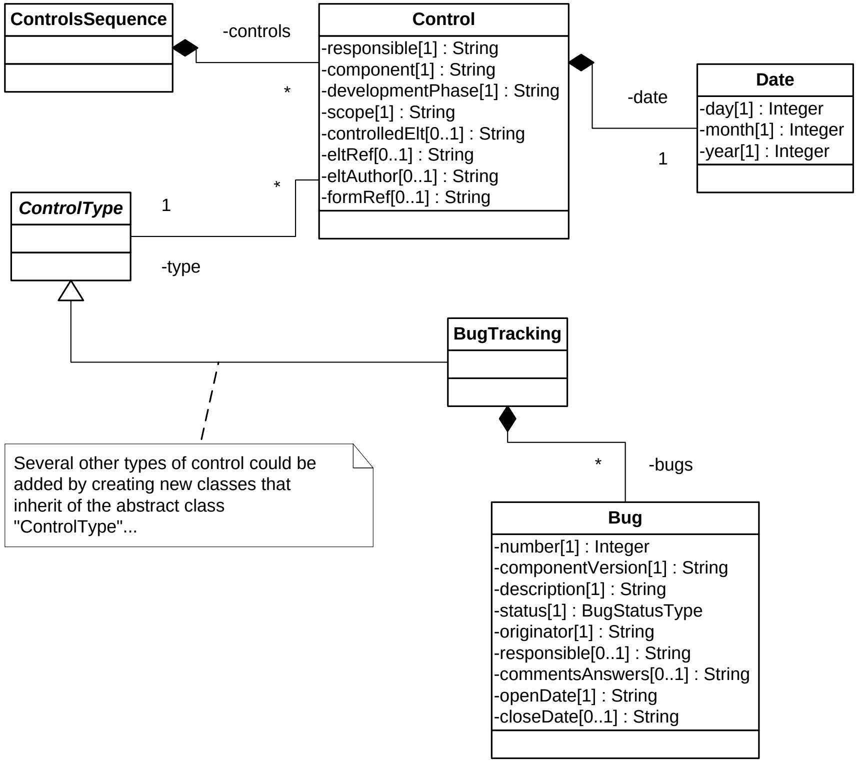
Attachment
-filename[1] : String
-size[1] : Integer
-contentType[1] : String
-downloadUrl[1] : String

Relationship
-type[1] : RelationshipType

Person
-login[1] : String

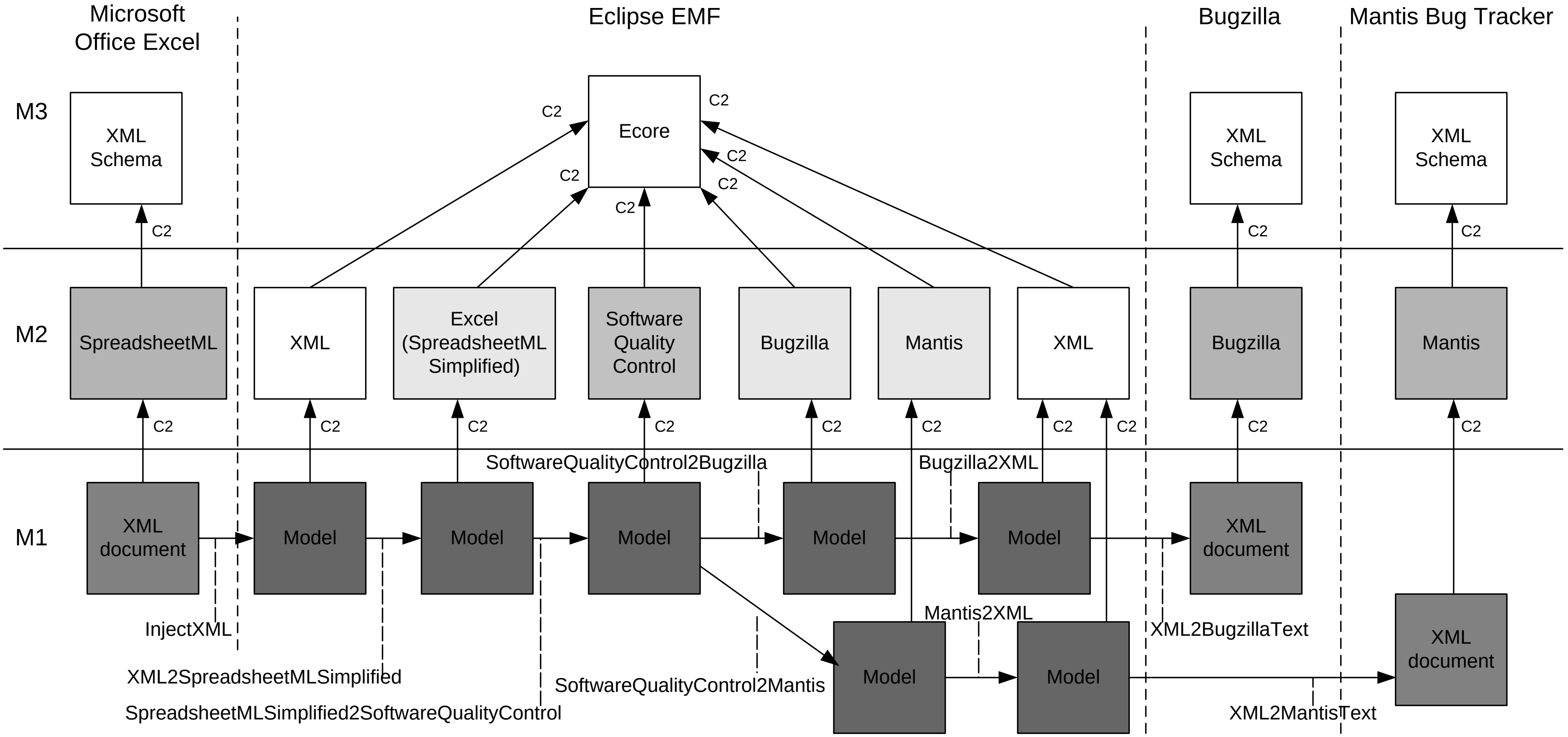
Note
-text[1] : String

Bug control metamodel (pivot)



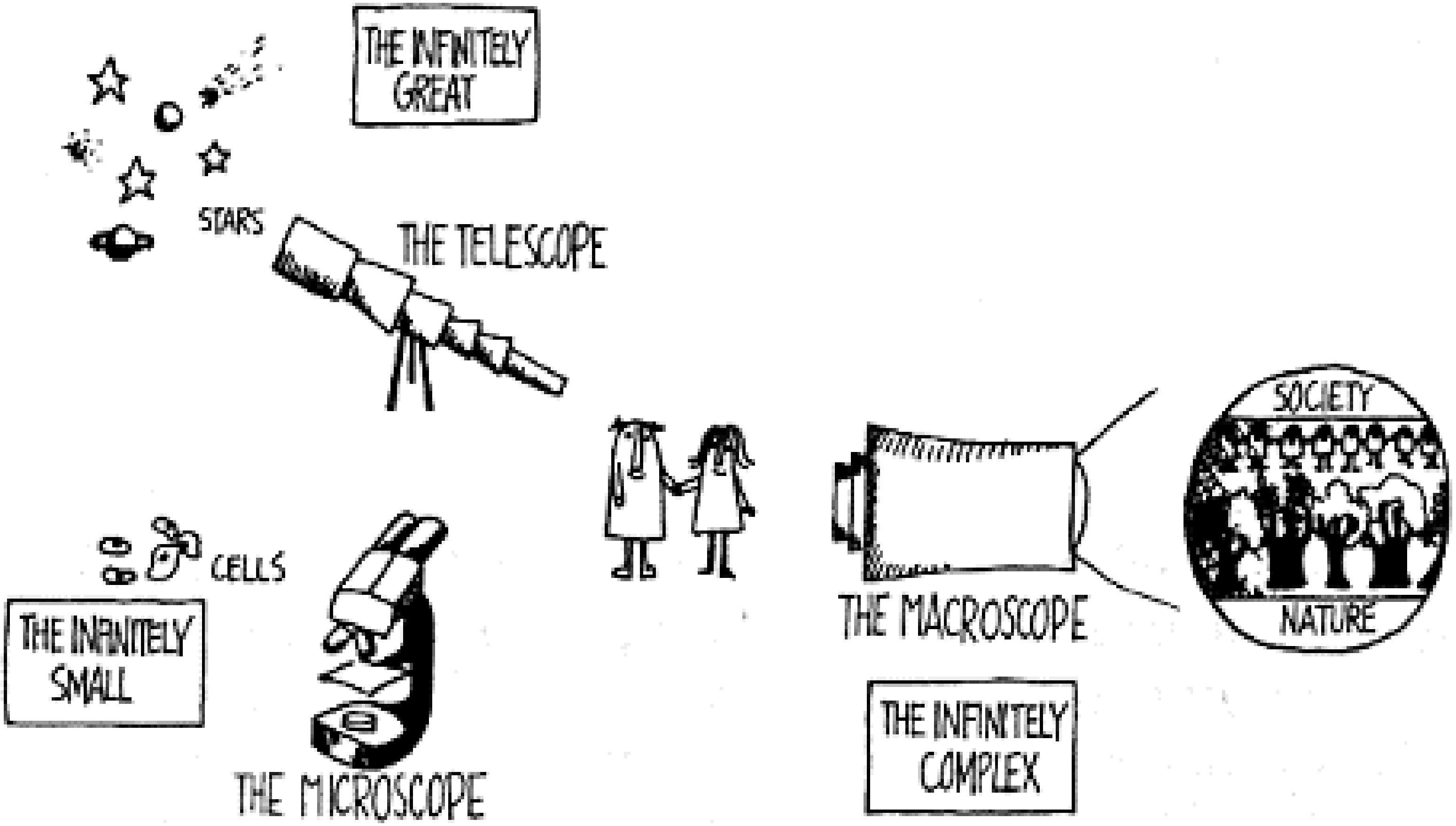
Several other types of control could be added by creating new classes that inherit of the abstract class "ControlType"...

Excel-to-Bugzilla and Excel-to-Mantis ATL bridges



Harnessing the additional complexity (accidental): global model management

Example : Complex to Simple

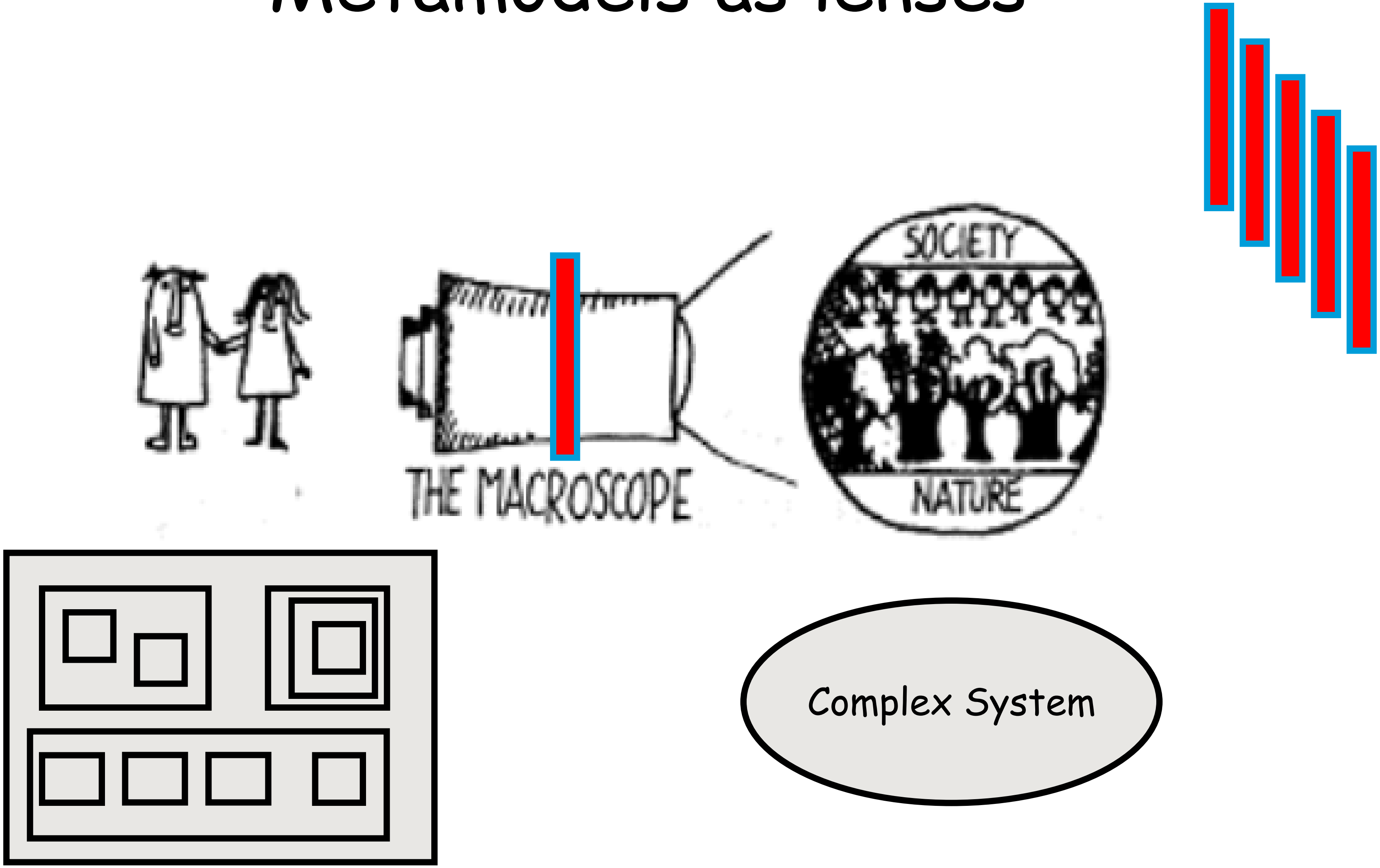


De Rosnay, J: The macroscope, Harper & Row, New York, 1979.

What is a Complex System?

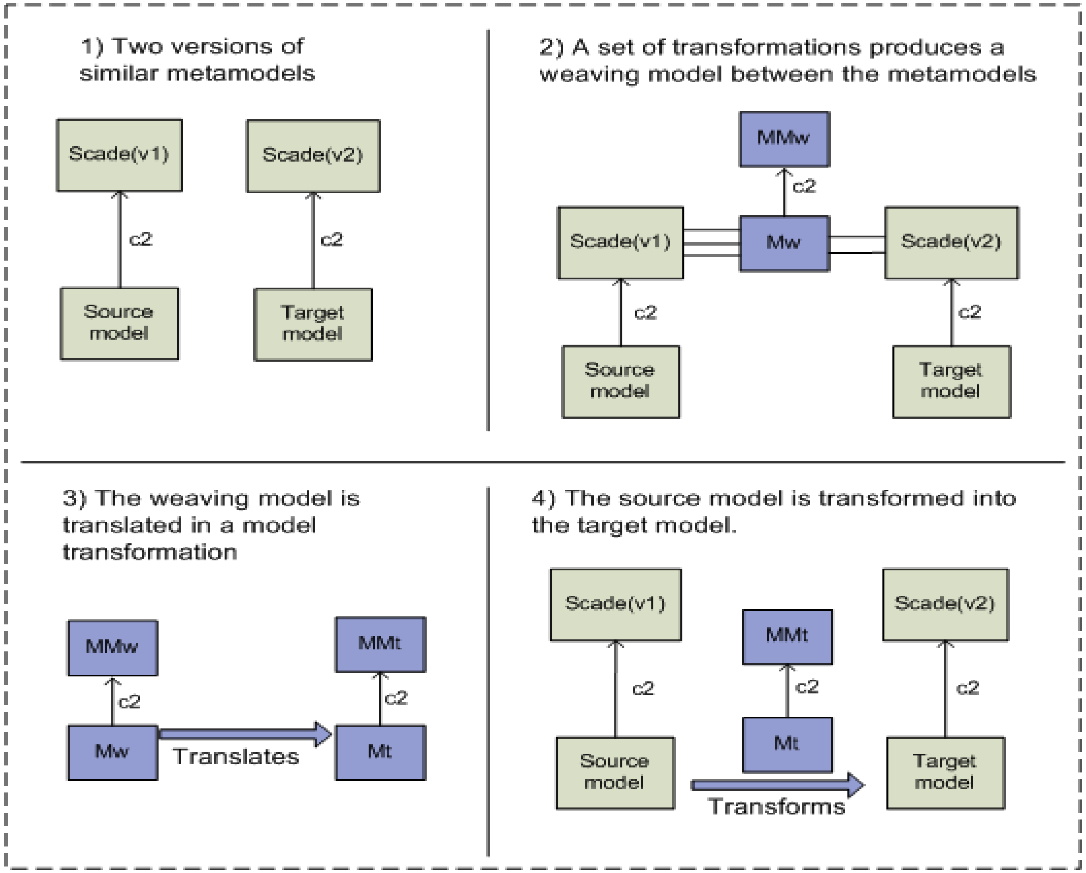
- CBCS: Computer-Based Complex System
  - ✓ A complex system with a significant number of hardware/software components
  - ✓ Compare with de Rosnay's biological or ecological complex systems
- A CBCS is composed of a large number of components
- A CBCS is constantly in evolution
  - ✓ Past, present, future
  - ✓ No stops when parts are added, removed or under maintenance
- A CBCS has a structure (static architecture) and a dynamic behavior
- A CBCS is composed of components that may be also CBCSs (no limit in nesting)
- A CBCS has a goal defining its purpose in the context in which it is operating
  - ✓ The goal of a CBCS is part of its metadata
- A CBCS has a heterogeneous-based engineering
- A CBCS is a distributed system
- A CBCS may not be understood by one unique human operator
- The interactions between different parts of a CBCS follow specific patterns , implicit or explicit
- Other properties of CBCS like behavior emergence

Metamodels as lenses



Complex System

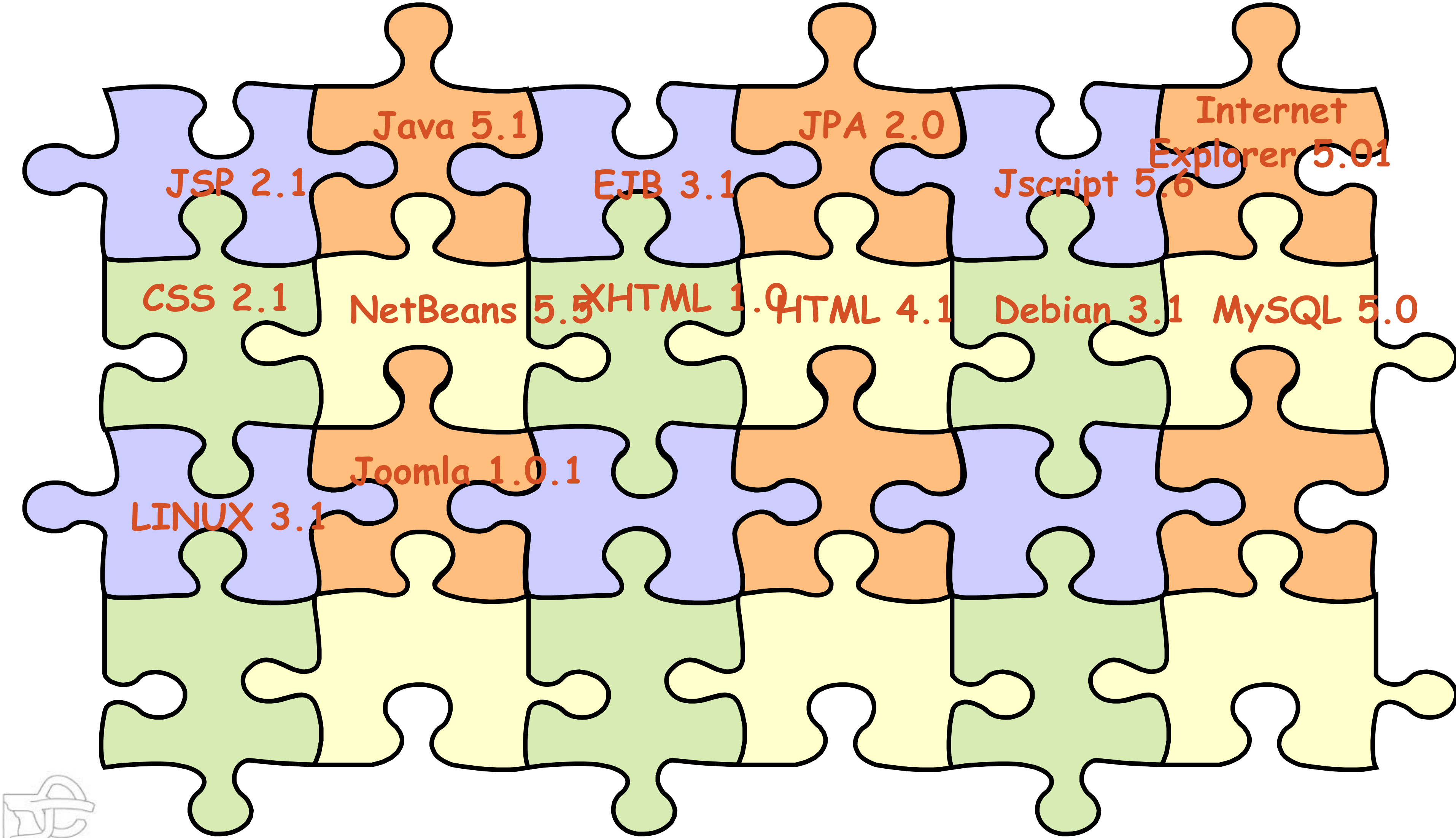
Example: Autosar 2.0 toAutosar 2.1


1) Two versions of similar metamodels
Scade(v1)
Scade(v2)
c2
c2
Source model
Target model
2) A set of transformations produces a weaving model between the metamodels
MMw
c2
Scade(v1)
Mw
Scade(v2)
c2
c2
Source model
Target model
3) The weaving model is translated in a model transformation
MMw
MMt
c2
c2
Mw
Translates
Mt
4) The source model is transformed into the target model.
Scade(v1)
MMt
Scade(v2)
c2
c2
c2
Mt
Source model
Target model
Transforms

"Metamodel comparison" Use Case's Overview

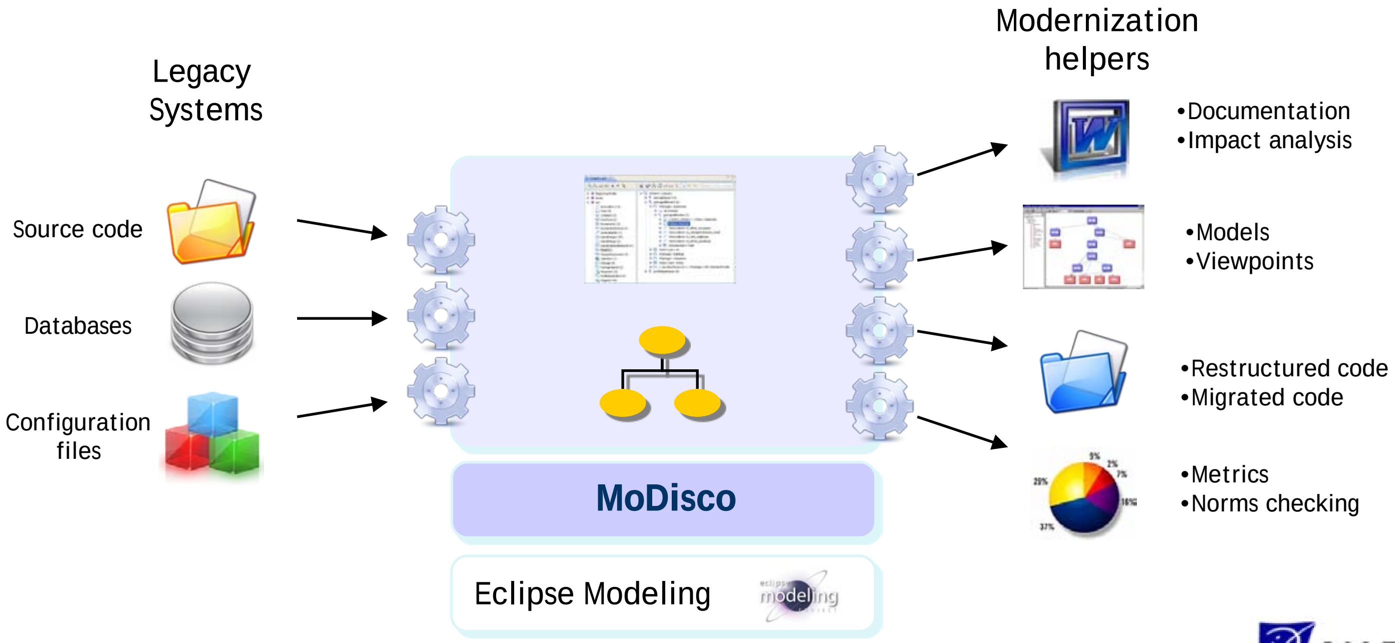
The comparison weaving model conforms to a weaving metamodel that is an extension of the core weaving metamodel. The metamodel extension contains an *Equivalent* link. This link contains a *similarity* attribute that saves the similarity estimation between a *left* and a *right* element. This link is extended by different kinds of links, depending on the type of elements that are being compared, for example *AttributeEqual* and *ReferenceEqual*. The relations between *ElementEqual* and *AttributeEqual* links are created according to the containment relations between Classes and Attributes. For the elements that have similarity value lower than a given threshold, the *Equivalent* links are rewritten into *NotEquivalent* links.

A typical problem: Version evolution in the information system jigsaw



Example Old to New

- MoDisco for "**Model Discovery**"
- Mining legacy to discover EMF/Ecore based models
- Extraction of models from legacy systems
  - ✓ Multiple types of such legacy systems
- *A* generic and extensible metamodel driven approach to model discovery



Legacy Systems

Source code

Databases

Configuration files

MoDisco

Eclipse Modeling

Modernization helpers

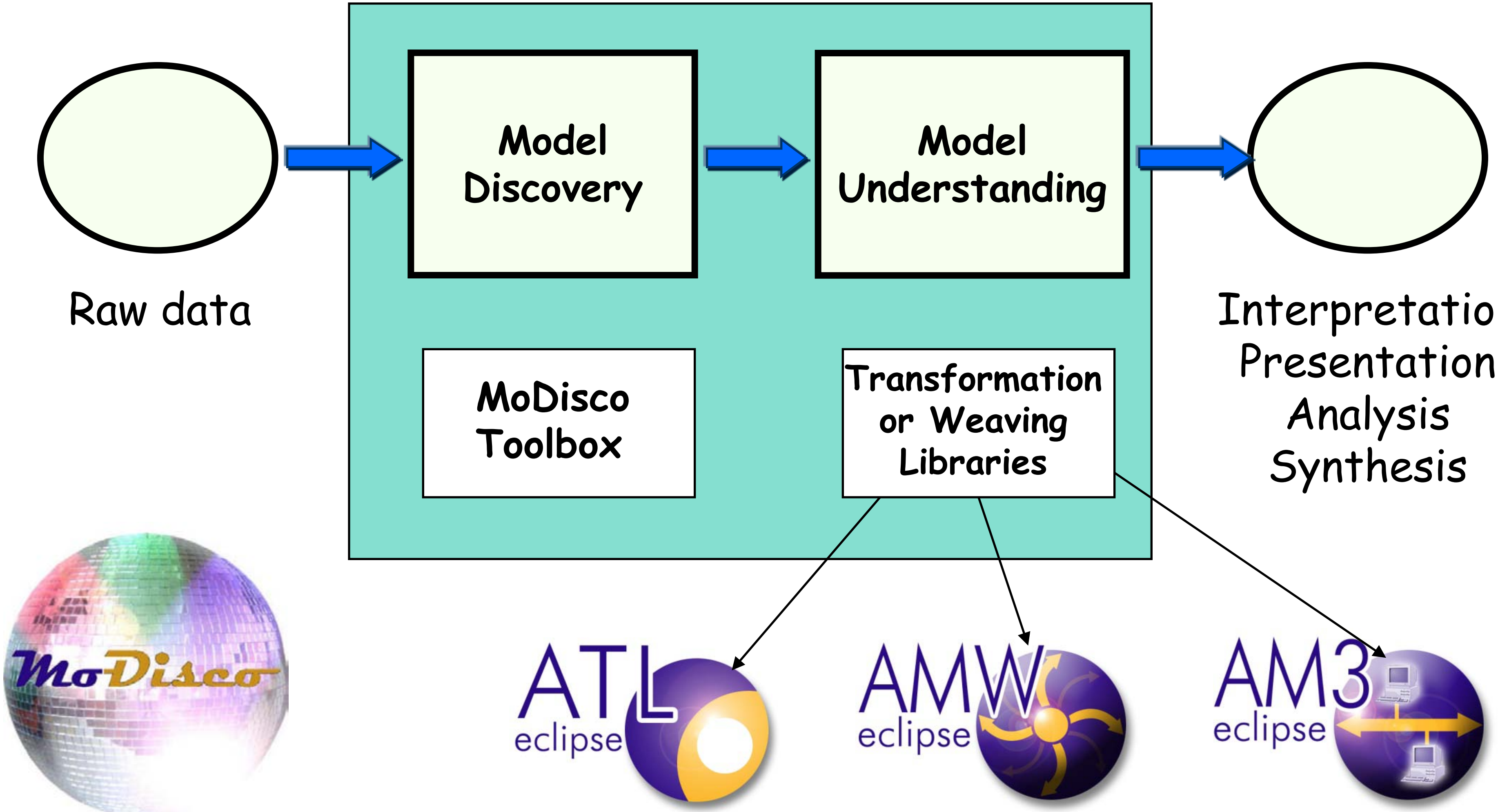
•Documentation
•Impact analysis

•Models
•Viewpoints

•Restructured code
•Migrated code

•Metrics
•Norms checking

The Modisco Component (Model Discovery)



Raw data

Model Discovery

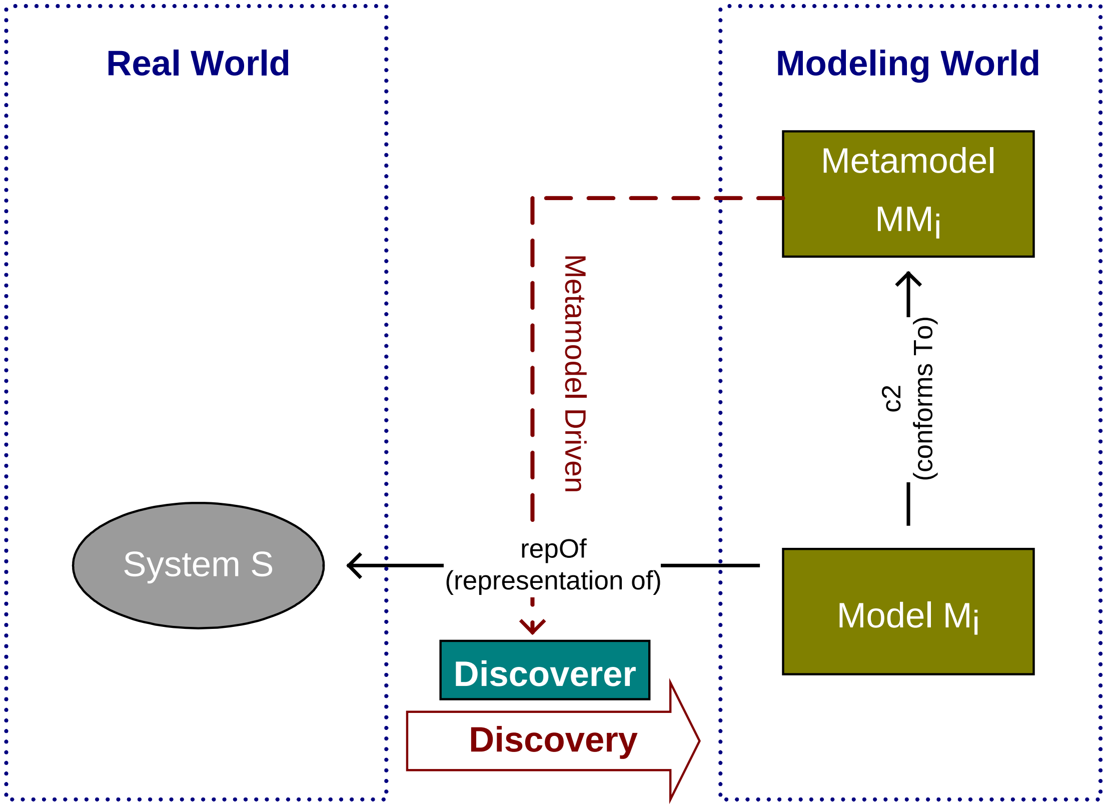
Model Understanding

Interpretation
Presentation
Analysis
Synthesis

MoDisco Toolbox

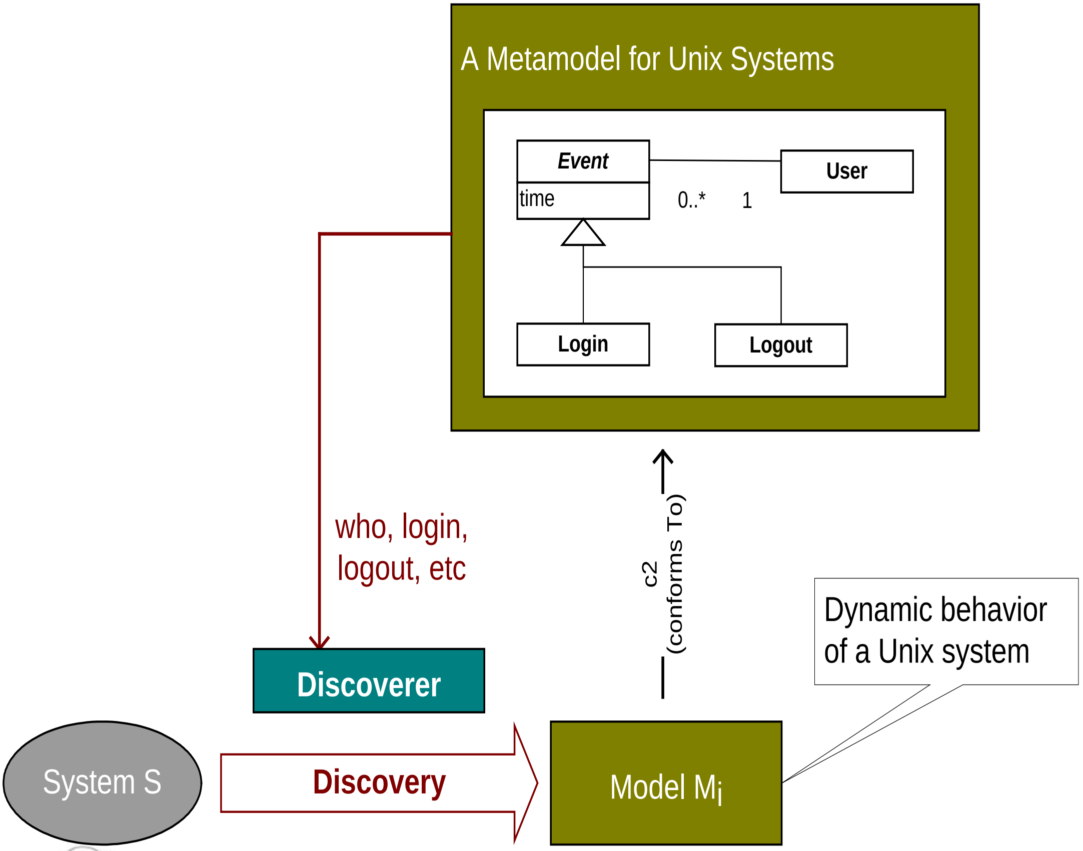
Transformation or Weaving Libraries

Discovery Principles



- Step 1:
  - Define the metamodel
- Step 2:
  - Create the "discoverer"
- Step 3:
  - Run the discoverer to extract model M_i from system S

Example



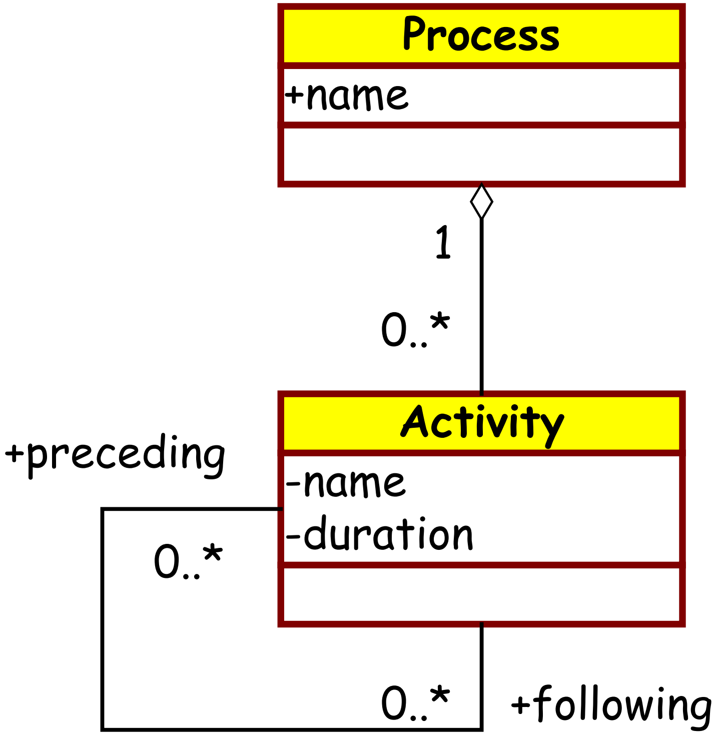
- Example of the Unix users' actions
- Study of the **dynamic** behavior of the system
  - Execution trace of the system

Will MDE scale up?

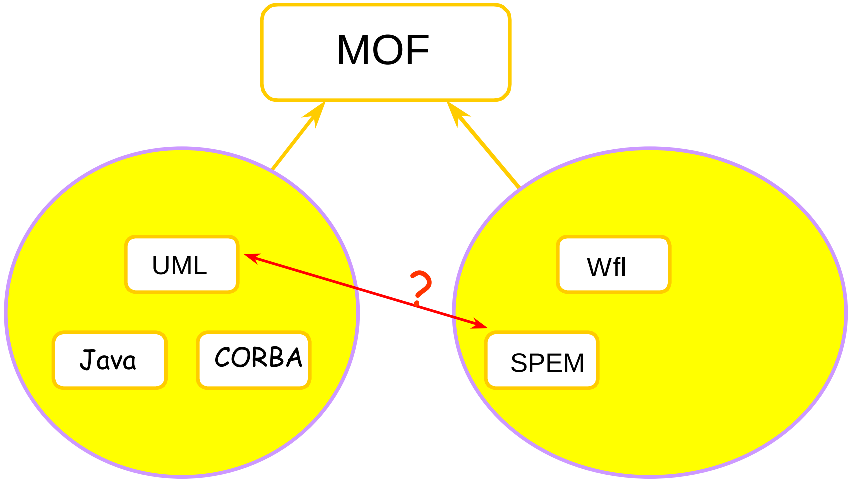
	Terminal Models	Metamodels	Others (e.g. transf.)
Size	<5 000 el. <50 000 el. <500 000 el.	<50 el. <5 000 el. <50 000 el.	<500 rules <5 000 rules
Evolutivity	Every second Every minute Every hour Every day Every year Never	Every second Every minute Every hour Every day Every year Never	Every second Every minute Every hour Every day Every year Never
Heterogeneity	XMI, native data, etc.	MOF, DSL, KM3, etc.	QVT, Viatra, ATL, XSLT, etc.
Quantity	1 10 100 1 000 10 000	1 10 100 1 000 10 000	1 10 100 1 000 10 000

MODEL TAXONOMY

Process and Product Models



Product and process explicit models



Who's doing what, when, how and why?

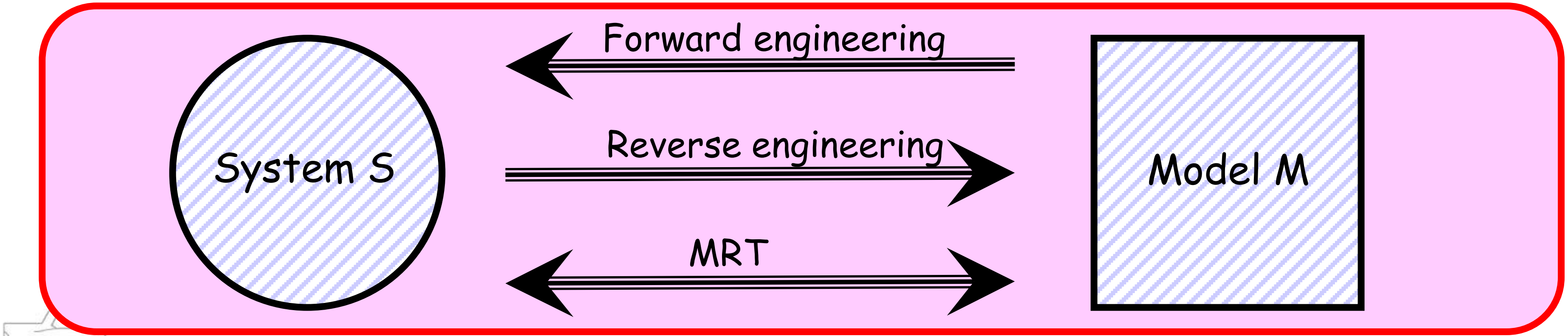
Static and Dynamic Systems/Models

- Most systems are dynamic
  - ✓ They evolve in time
  - ✓ Example : a washing machine
- Most models are static
  - ✓ They don't evolve in time
  - ✓ Example: a statechart of a washing machine
- Counter examples (rare)
  - ✓ Static system : Census results
  - ✓ Dynamic model : Simulation program

System ⇩ Model ⇨	Static	Dynamic
Static	Rare	Impossible
Dynamic	Frequent	Simulation

Code and Data Models

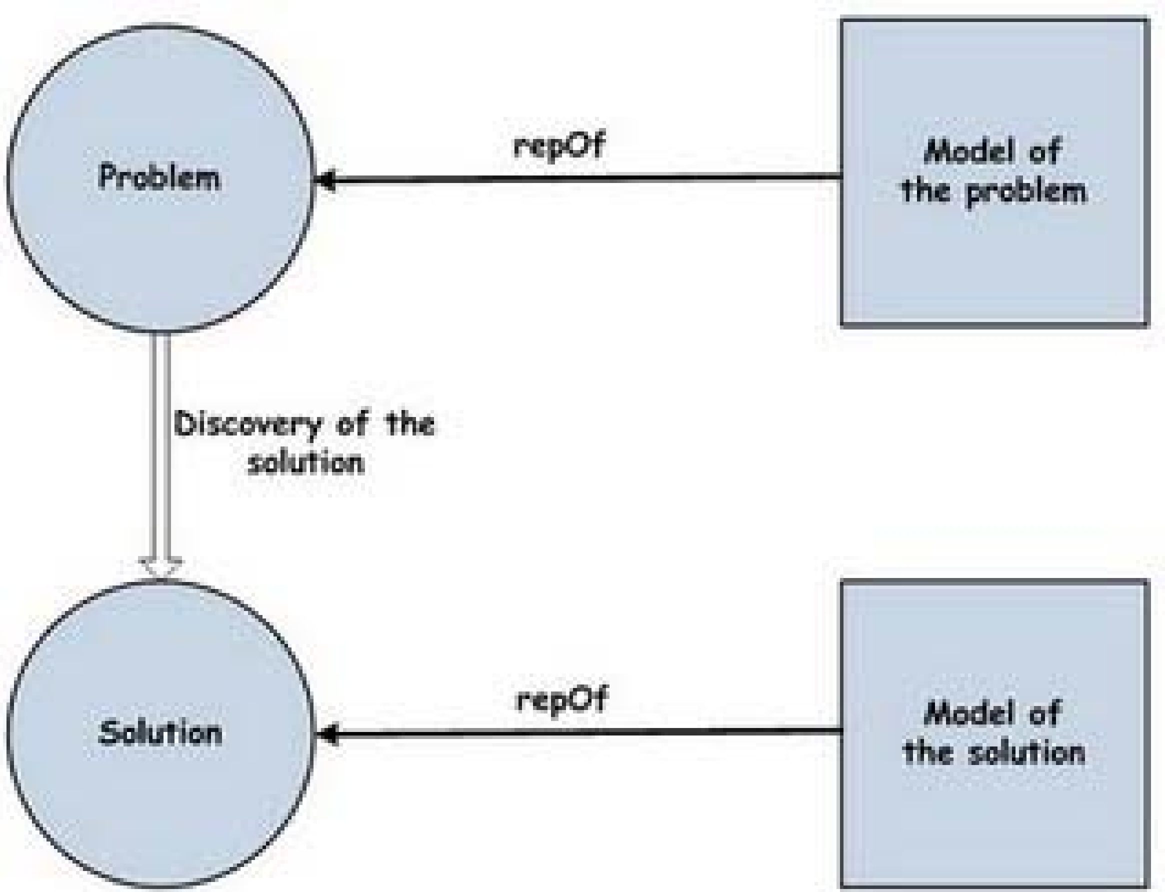
- Generating code from hand-coded models
- Generating models from legacy code (code as data)
- Models of data



Problem and Solution Models

- Once upon a time, there was a team leader that was going on holidays. Before leaving, (s)he made the last recommendation to his/her small team of three young engineers. For the ongoing project, do not start coding in Java before the UML model is completely finished and that you all agree on this model.
- On the Monday morning, as soon a s/he left, one of the engineers told the others of a wonderful discovery he made while twittering in the week end : a very powerful tool to generate UML diagrams from UML code.
- The decision was rapidly taken and all three of them started coding the problem in Java.
- Some days before the end of the holidays of their leader, all the Java code was used to generate UML diagrams and both the code and the UML diagrams were handled to the group leader.
- S/He was quite impressed at the level of detail of the UML model and the narrow correspondence between the code and the model.

Problem and Solution Models



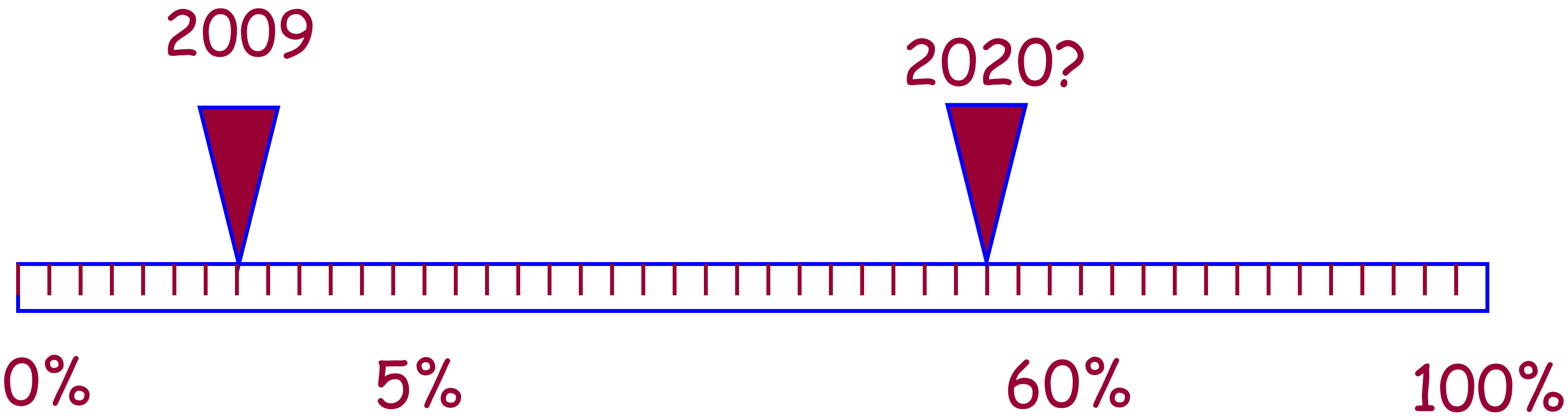
*MDE is a unique chance to achieve the goal of separation of the **what** and of the **how**, for example in the educational context.*

CONCLUSIONS

Main messages of the presentation

- What is a model?
  - ✓ A model is a representation of a system
  - ✓ A model is written in the language of its metamodel
  - ✓ A metamodel is written in the language of its metametamodel
  - ✓ A model is a typed graph
- Where do models come from?
- What are the various kinds of models?

MDE fulfilling the promises?



But 100% of what?
Of code generation?
But then of code generation from which model? Design? Requirements?
If MDE is the solution, then what is the problem?
We still need to make precise the MDE promises.

Main messages of the presentation

- Model Driven Engineering (i.e. the consideration of models as first class entities and the representation of all types of artifacts by models) and its various variants are changing the landscape of software engineering, system engineering and data engineering.

- Foundations are important: Principles are beginning to be consensually identified (relations of representation and conformance)

- MDE has evolved a lot since its inception in 2000
  - ✓ PSM generation (Java) from a PIM (UML)
  - ✓ Generating tests or other software artifacts (conf. files)
  - ✓ Separation and combination of aspects
  - ✓ Interoperability (tool, data, enterprise, etc.)
  - ✓ Using MDE to manage complex systems: a big challenge

Main messages of the presentation

- MDE is about working with a lot of open an explicit metamodels (precise metamodeling).
- Model transformation, model weaving and global model management are the typical facilities needed in a practical MDE framework
- Three levels of complexity
  - ✓ **S ⇐ M** (MD Software Development)
  - ✓ **S ⇒ M** (MD Reverse Engineering)
  - ✓ **S ⇔ M** (Run Time Modeling)
    - ➢ External reflection
    - ➢ Model-Based introspection and intercession
    - ➢ Descriptive/Predictive/Prescriptive

Main messages of the presentation

1. A process is a model
2. A platform is a model
3. A transformation is a model
4. A metamodel is a model
5. A model-element is a model
6. A program is a model
7. An execution trace is a model
8. A measure is a model
9. A test is a model
10. A decoration is a model
11. An aspect is a model
12. A pattern is a model
13. A legacy system is a model
14. An event trace is a model
15. Any data is a model
16. etc.

Main messages of the presentation

- Model Driven Testing
- Model Driven Validation and Verification
- Model Driven Requirement Engineering
- Model Driven Web Engineering
- Model Driven Process Engineering
- Model Driven Interoperability
- Model Driven Integration of Product Lines
- Model Driven Reusability
- Model Driven Traceability
- Model Driven Data Stream Processing
- Model Driven System Architecture
- Model Driven Data Engineering
- Model Driven Reverse Engineering
- Model Driven Software Modernization
- Model Driven Software Evolution
- Model Driven System Administration
- etc.

The "Towers of Models" Grand Challenge (Robin Milner)

A more thorough science-based approach to informatics and ubiquitous computing is both necessary and possible. We often think in terms of models, whether formal or not. These models, each involving a subset of the immense range of concepts needed for ubiquitous computer systems, should form the structure of our science.

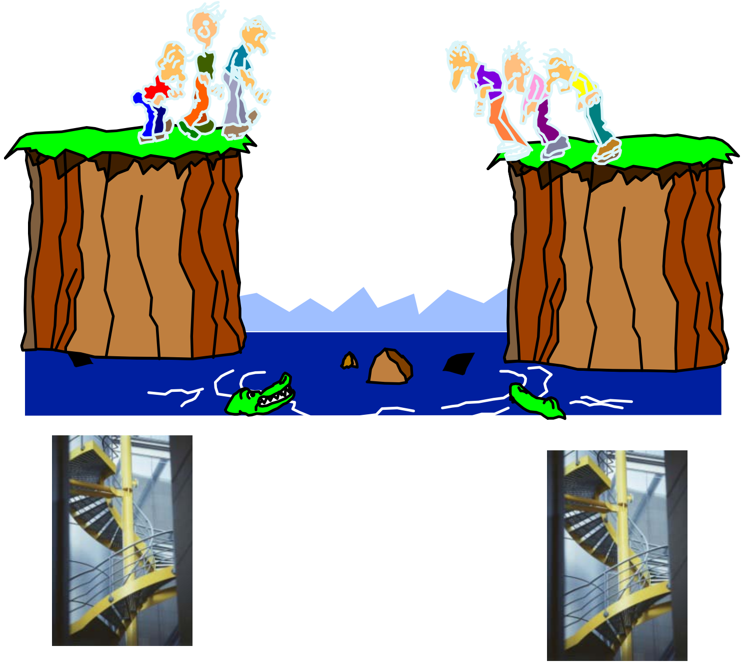
- Even more importantly, the relationships (either formal or informal) among them are the cement that will hold our towers of models together. For example, how do we derive a model for senior executives from one used by engineers in designing a platform for business processes, or by theoreticians in analyzing it?

- The essence of software engineering and informatics is formulating, managing, and realizing models.

The importance of teaching

"Teaching reduces the gap and research increases it again"

(C.A.R. Hoare, ICSE-18)



The MDE Diploma

Model Driven Engineering for Software Management

International Post-graduate Specialization Diploma
Awarded by the French Ministry of Industry.

A Comprehensive Course
on Advanced Software Production, Operation, and Maintenance
Based on Model Driven Engineering.

Ecole des Mines de Nantes

- La Chantrerie
- 4, rue Alfred Kastler

B.P. 20722
F-44307 Nantes Cedex 3
Phone: (+33) 2 51 85 81 00
Fax: (+33) 2 51 85 81 99

The MDE Diploma

- **Module 1: Prerequisites (60h)**
  - ✓ Software Development with Eclipse
  - ✓ Free and Open Source Models for software development
  - ✓ Software Modeling
- **Module 2: Fundamentals (120h)**
  - ✓ Fundamentals of Metamodeling and DSLs
  - ✓ Theory and Practice of Model Transformation
  - ✓ Advanced Model Management: repositories & collaborative development
  - ✓ Basic Model Driven Software Development
- **Module 3: Applications of MDE (120h)**
  - ✓ Information Systems
  - ✓ Embedded Systems
  - ✓ Data Engineering
  - ✓ Web Engineering
  - ✓ Graphical User Interfaces
  - ✓ Legacy Reverse Engineering
  - ✓ Process Engineering
  - ✓ System Engineering
- **Module 4: Management (60h)**
  - ✓ Management of MDE Projects
  - ✓ Alignment of Business Needs with Technical Platforms
  - ✓ Cartography of Information Systems
  - ✓ Strategies for Information System Evolution and Modernization
  - ✓ Human and Organizational Factors in Transitioning from Previous Technologies
- **Module 5: Internship (6 months)**
  - ✓ A co-op stay in a company or in a lab to work on a MDE project.

- Move from object technology to model engineering:
  - Probably much more difficult than the migration from procedural technology to object technology in the 80's

Thanks

- ✓ Questions?
- ✓ Comments?

http://www.emn.fr/x-info/atlanmod/

Jean Bézivin
Jean.Bezivin{noSpamAt}inria.fr
AtlanMod research team, INRIA & EMN, Nantes, France