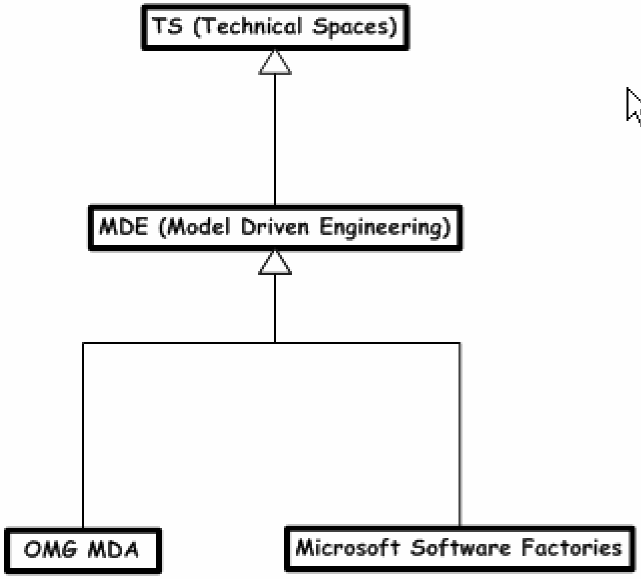
Model Engineering: from Principles to Platforms

Jean Bézivin
Jean.Bezivin{noSpamAt}lina.univ-nantes.fr
ATLAS Group (INRIA & LINA),
University of Nantes, France
http://www.sciences.univ-nantes.fr/lina/atl/

Agenda

- The industrial evolution (OMG MDA™, IBM EMF, Microsoft Software factories) and the MDE trend
- The need for sound principles (models as first class entities)
- Technology spaces or why MDE is not sufficient
- Why OT was not sufficient and what did we learn?
- Towards an ideal open MDE platform
- The future of model engineering: Challenges & Perspectives



Agenda

The industrial evolution (OMG, IBM, Microsoft)

MDA™: OMG's new vision

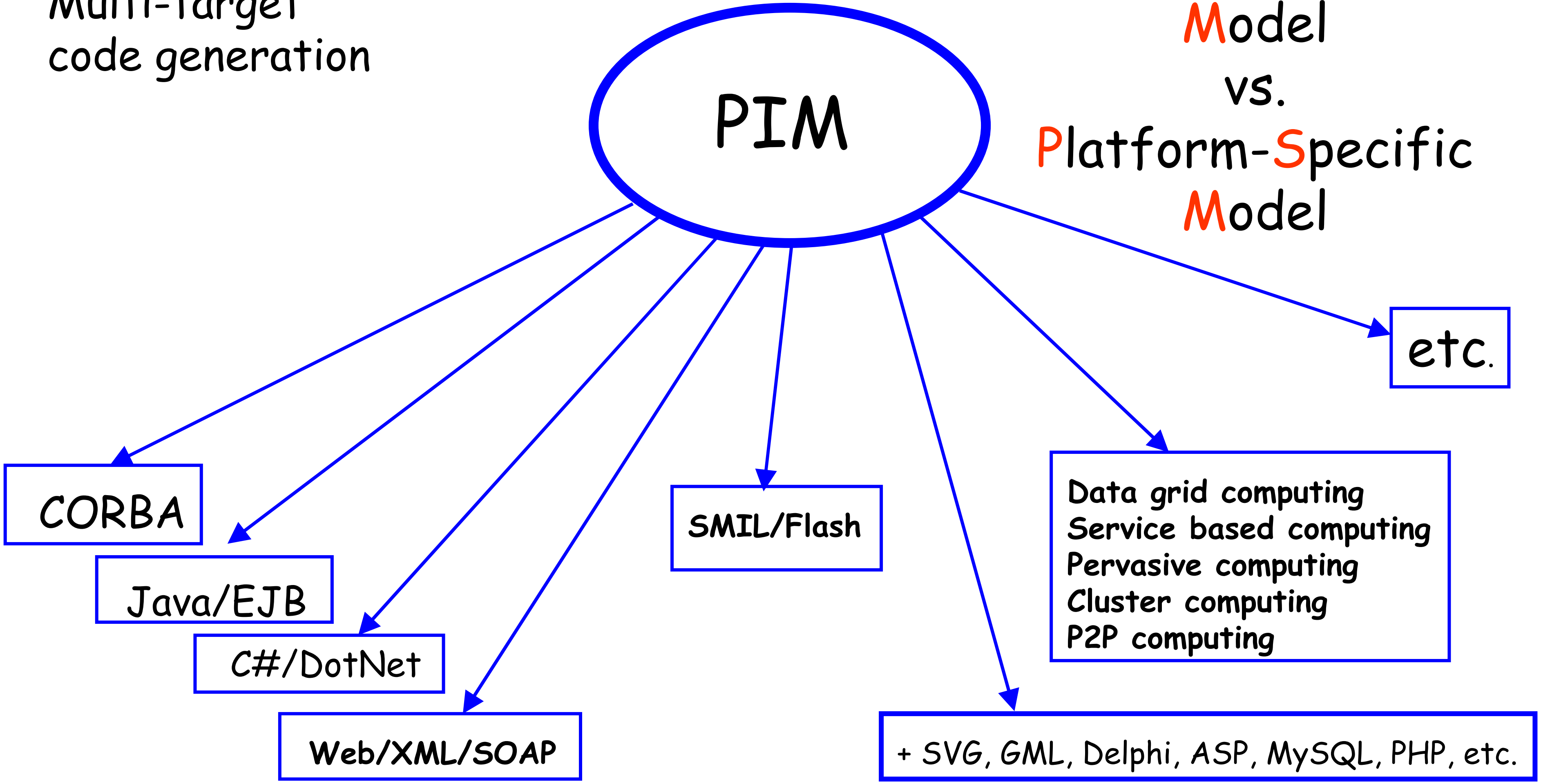
"OMG is in the ideal position to provide the model-based standards that are necessary to extend integration beyond the middleware approach… Now is the time to put this plan into effect. Now is the time for the Model Driven Architecture."

Richard Soley and the OMG staff,
MDA Whitepaper Draft 3.2
November 27, 2000

~~Write Once, Run Anywhere~~
Model Once, Generate Anywhere

Multi-target
code generation

Platform-Independent
Model
vs.
Platform-Specific
Model



PIM

CORBA

Java/EJB

C#/DotNet

Web/XML/SOAP

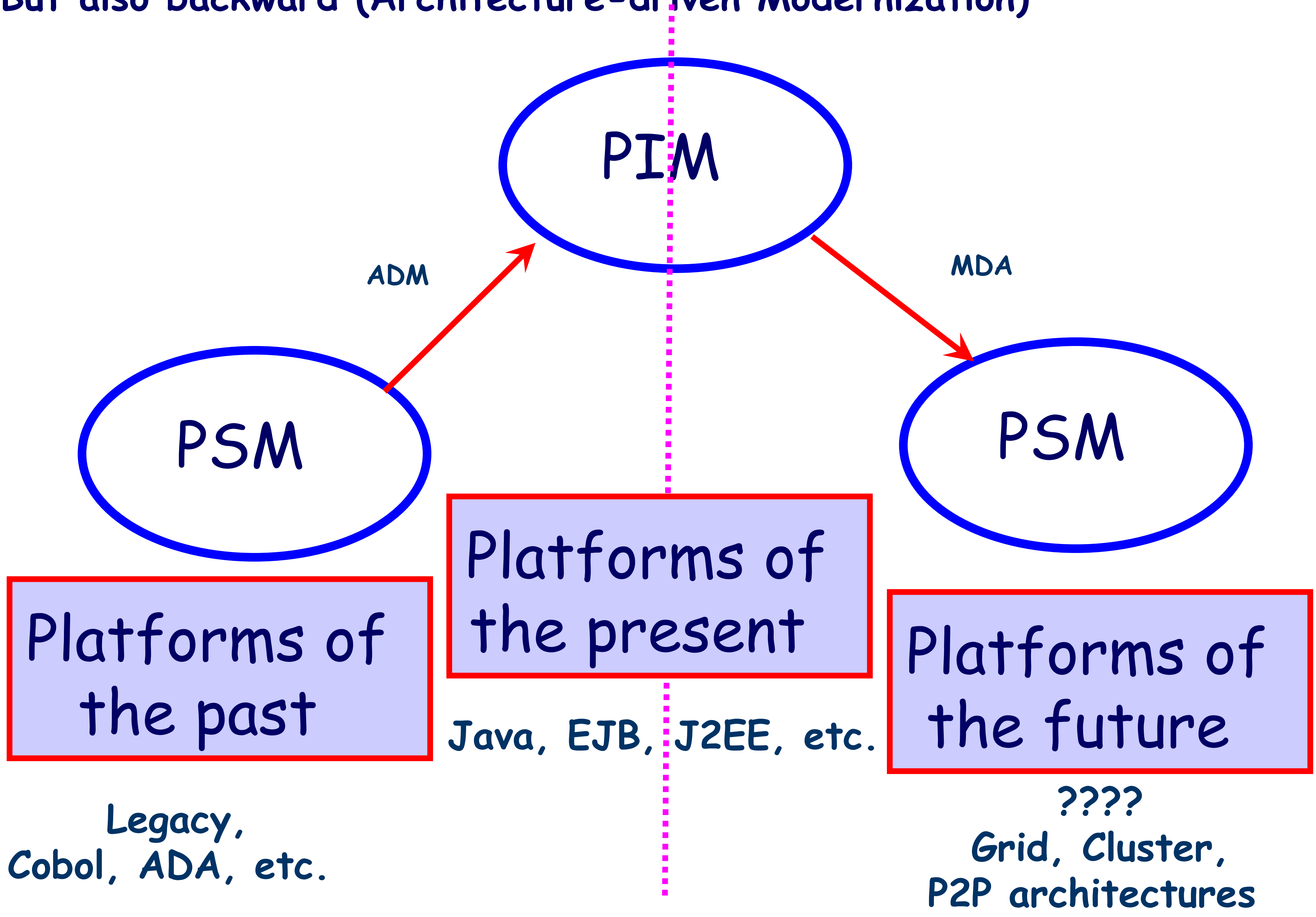
SMIL/Flash

+ SVG, GML, Delphi, ASP, MySQL, PHP, etc.

Data grid computing
Service based computing
Pervasive computing
Cluster computing
P2P computing

etc.

But also backward (Architecture-driven Modernization)



PIM

ADM

MDA

PSM

PSM

Platforms of the past

Legacy, Cobol, ADA, etc.

Platforms of the present

Java, EJB, J2EE, etc.

Platforms of the future

????
Grid, Cluster,
P2P architectures

Steve Cook (OOPSLA 2004 panel)

Suggests that MDA proponents fall into the following three camps:

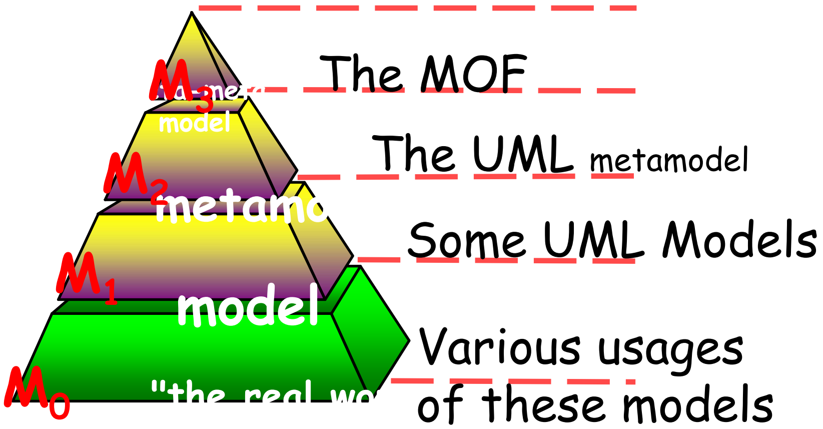
1. The UML PIM camp: MDA involves the use of UML to build Platform Independent Models (PIMs) which are transformed into Platform Specific Models (PSMs) from which code is generated.

2. The MOF camp: MDA does not involve the use of UML, but instead the crucial technology is MOF, and the definition of modelling languages and language transformations using MOF.

3. The Executable UML camp: MDA involves building a UML compiler, making it a first class programming language.

Ref: Steve Cook Blog @: http://blogs.msdn.com/stevecook

Coordinated metamodels (conformance)

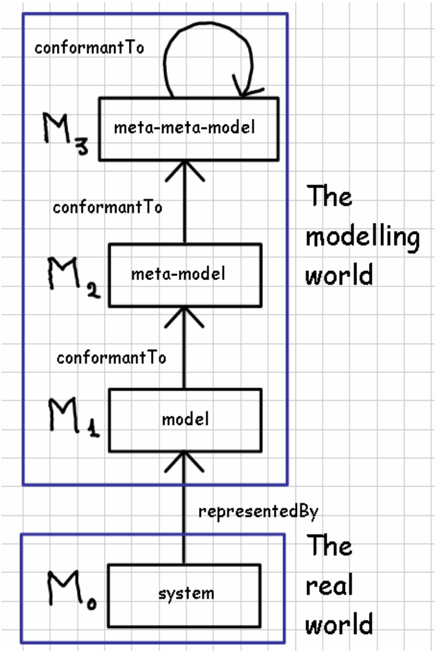


The MOF

The UML metamodel

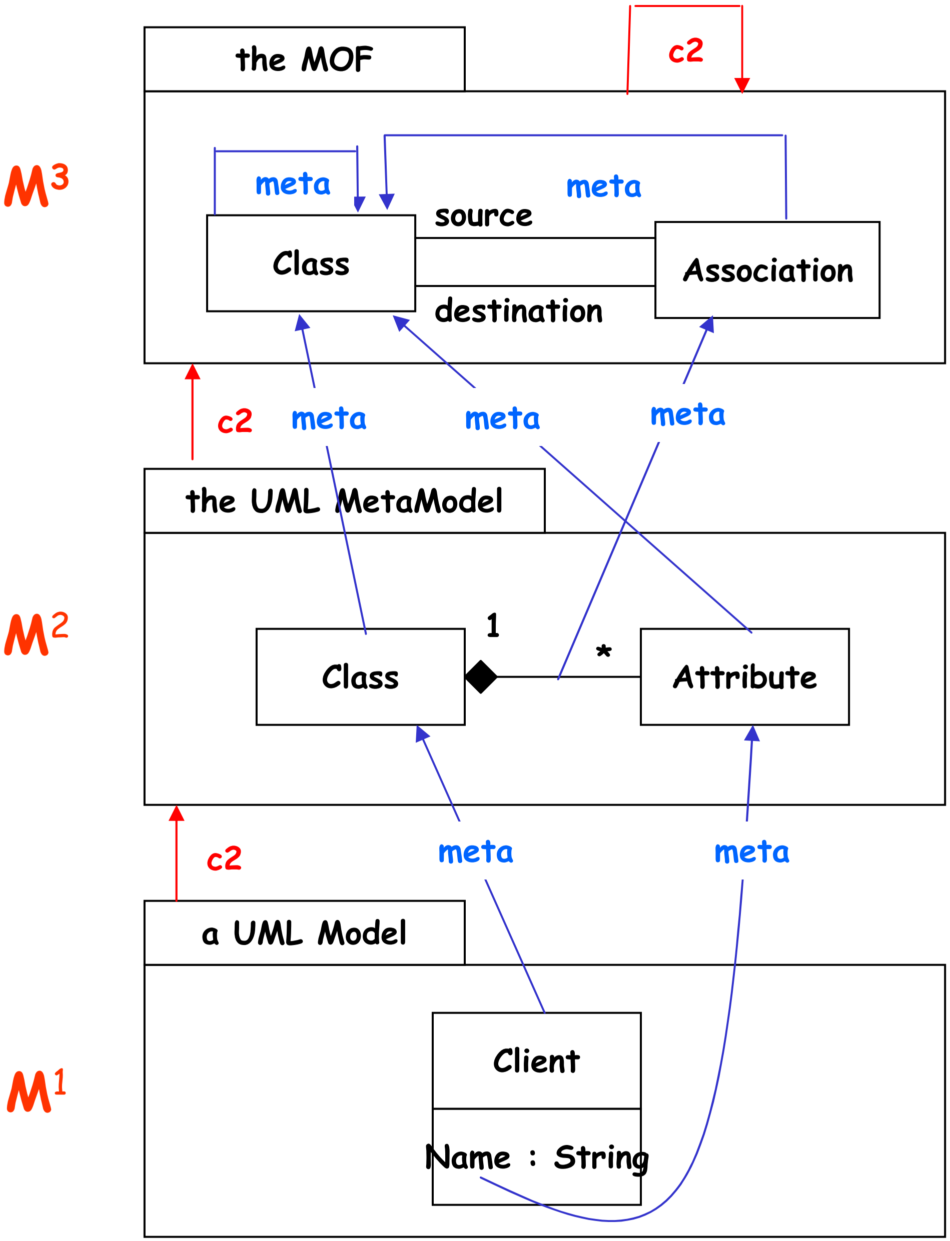
Some UML Models

Various usages of these models



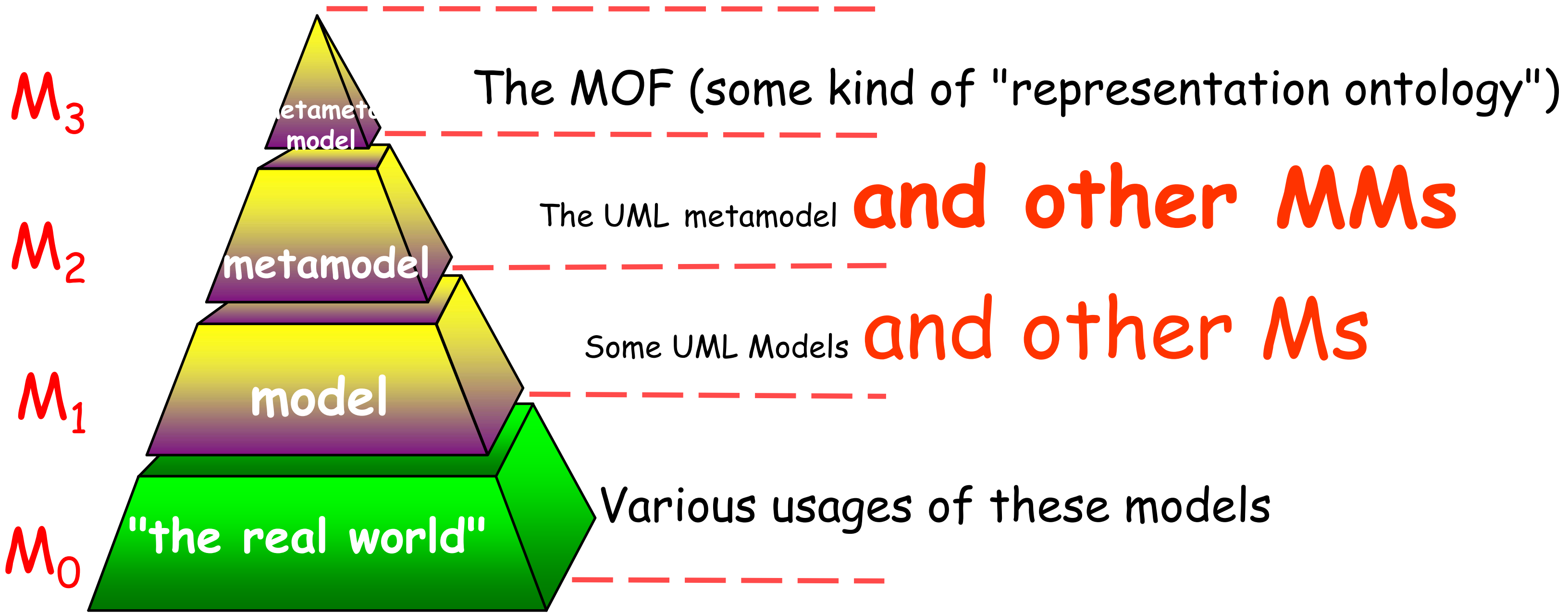
The modelling world

The real world



The metamodelling stack

- UML
- CWM
- SPEM
- EDOC
- QVT
- MDA is based on a coordinated set of metamodels



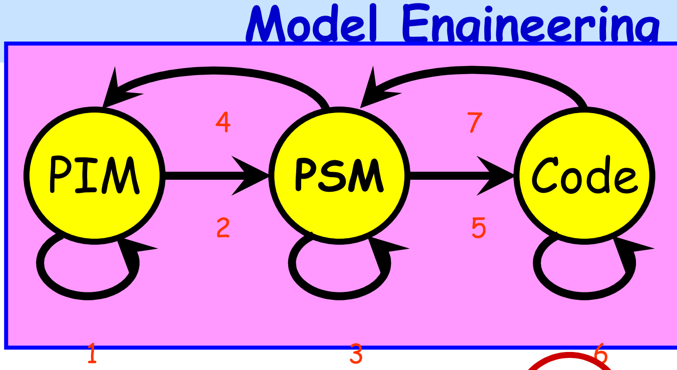
The MOF (some kind of "representation ontology")

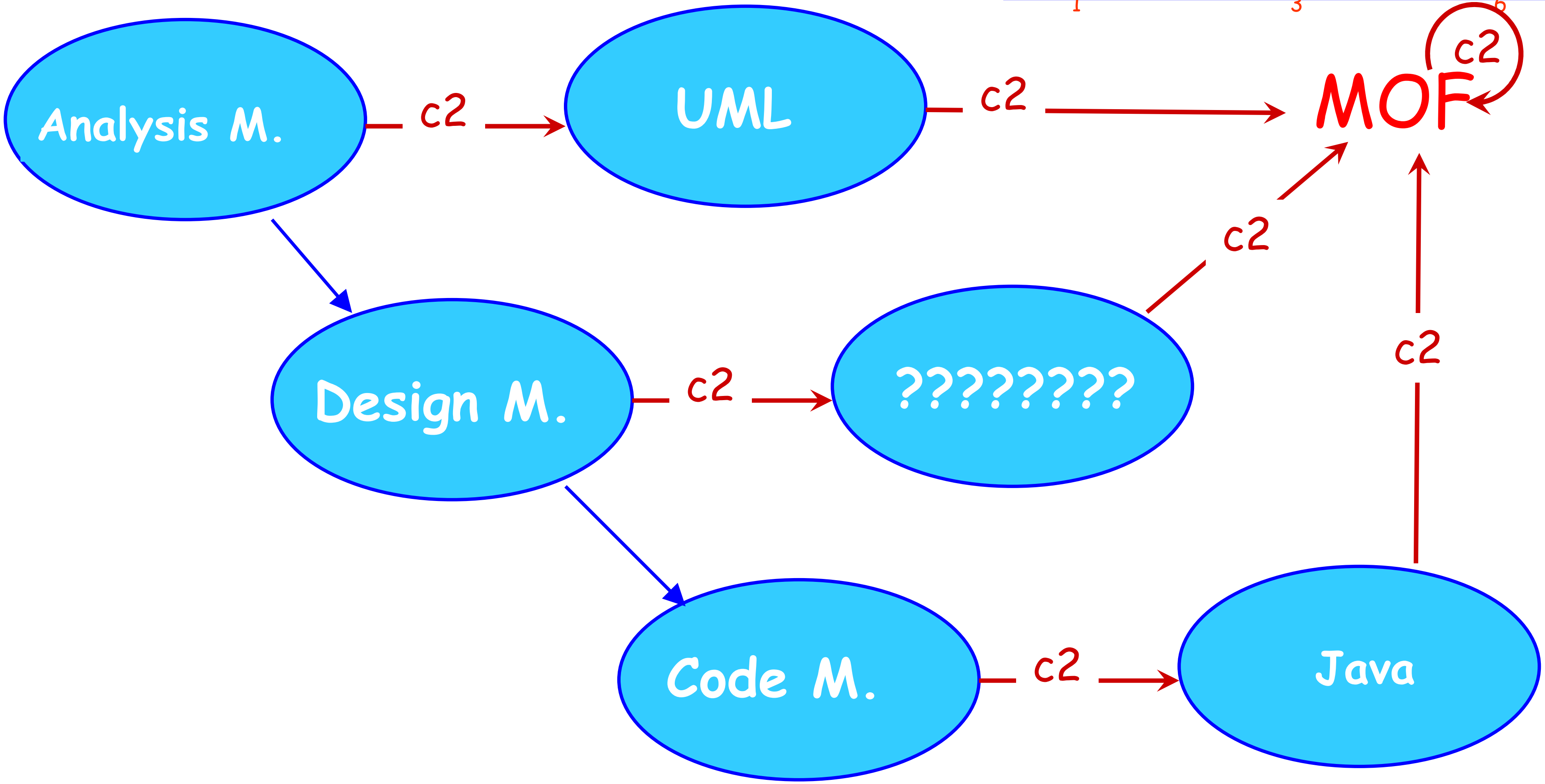
The UML metamodel **and other MMs**

Some UML Models **and other Ms**

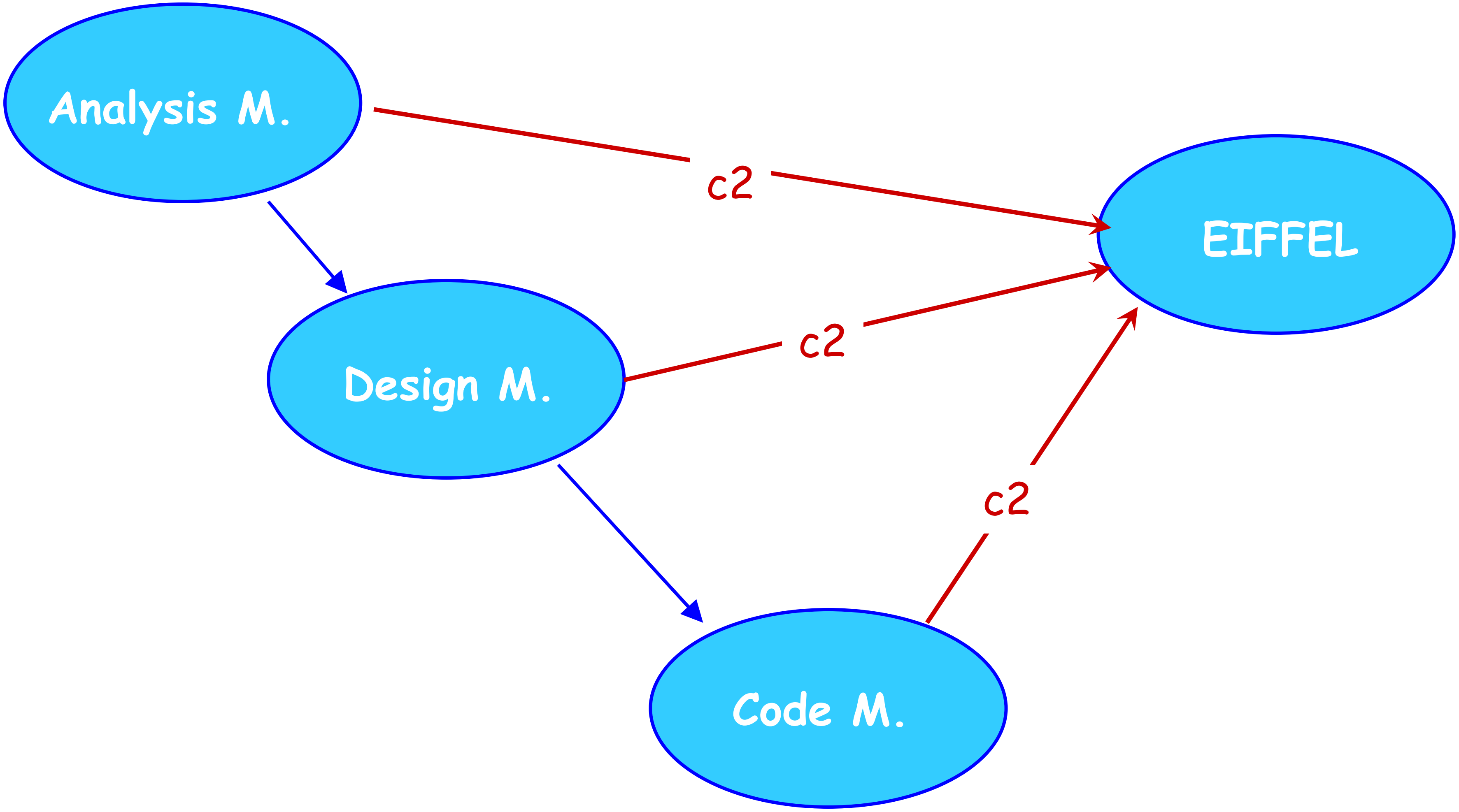
Various usages of these models

MDA general organization

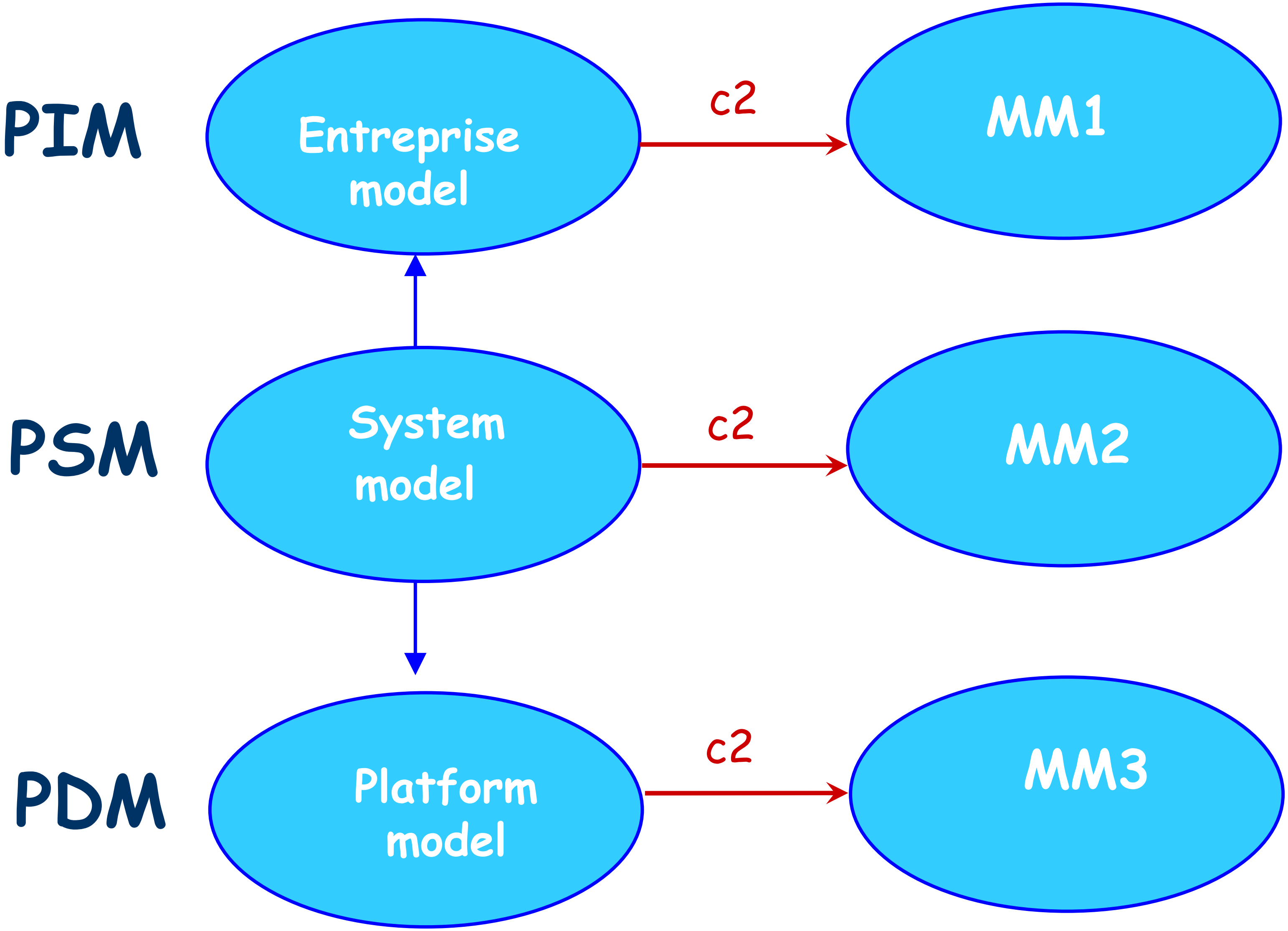




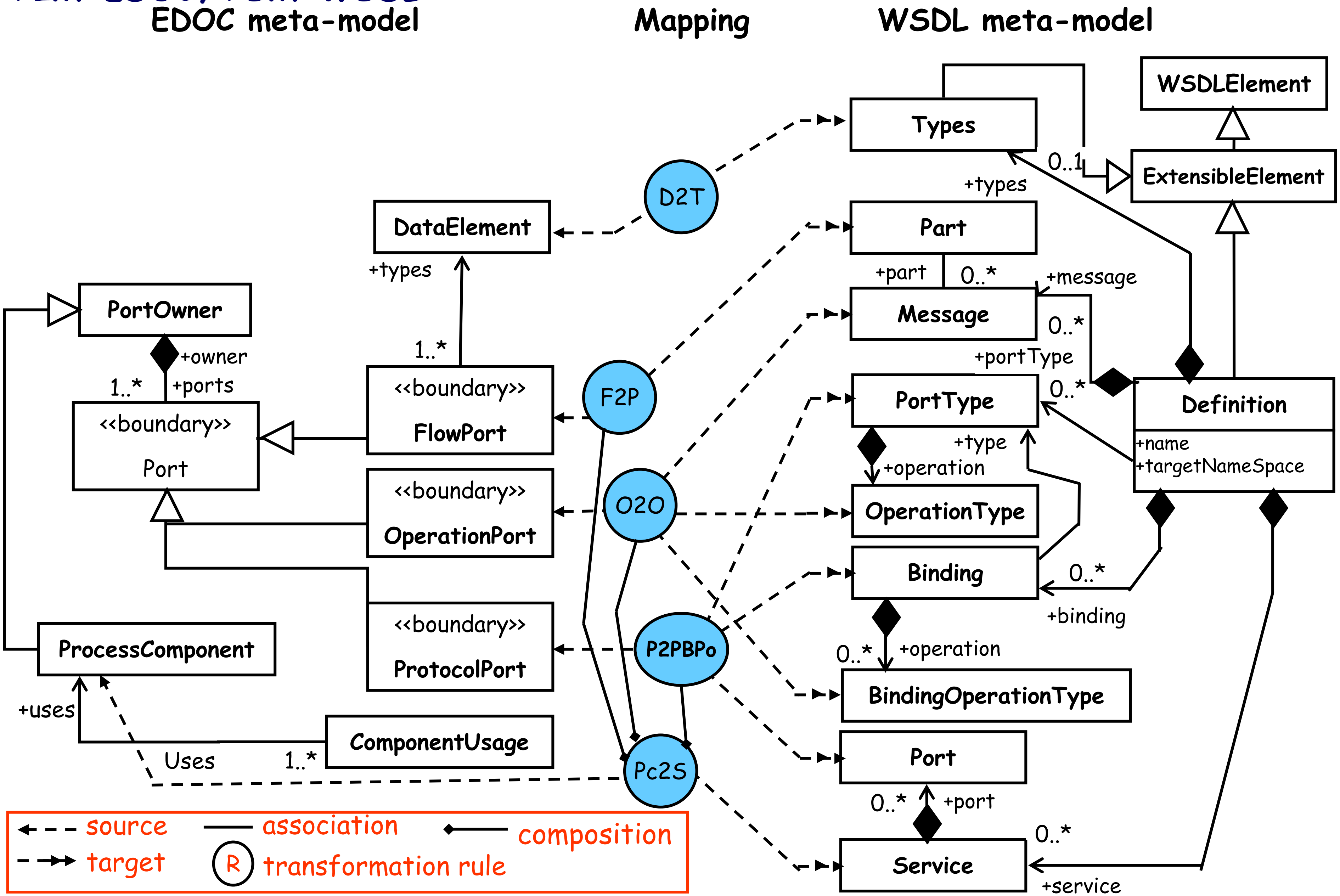
Simplified vision of the world



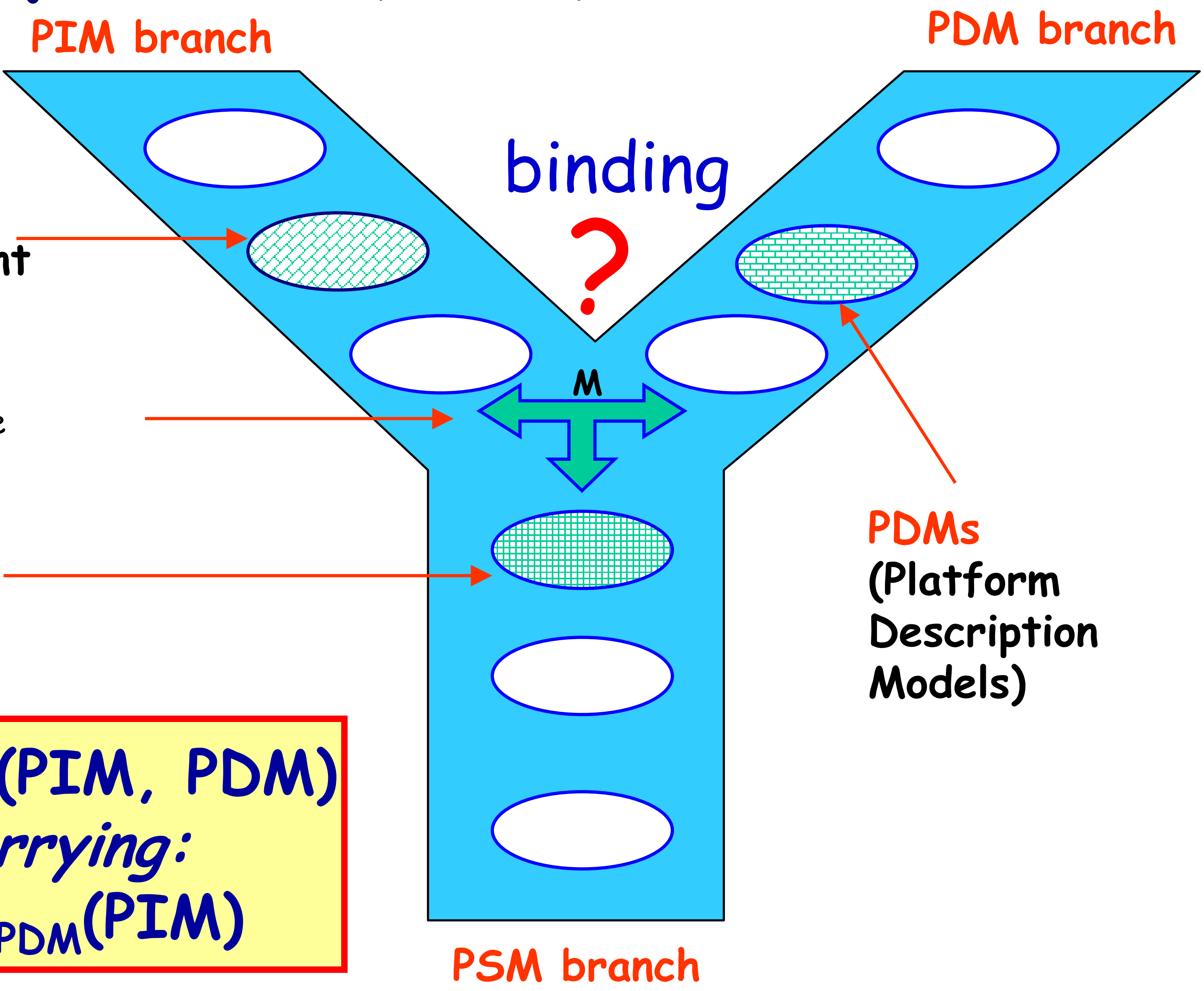
General case



PIM=EDOC; PSM=WSDL



The initial Y conjecture : PSM = f(PIM, PDM)



PIMs (Platform Independent Models)

Merging phase

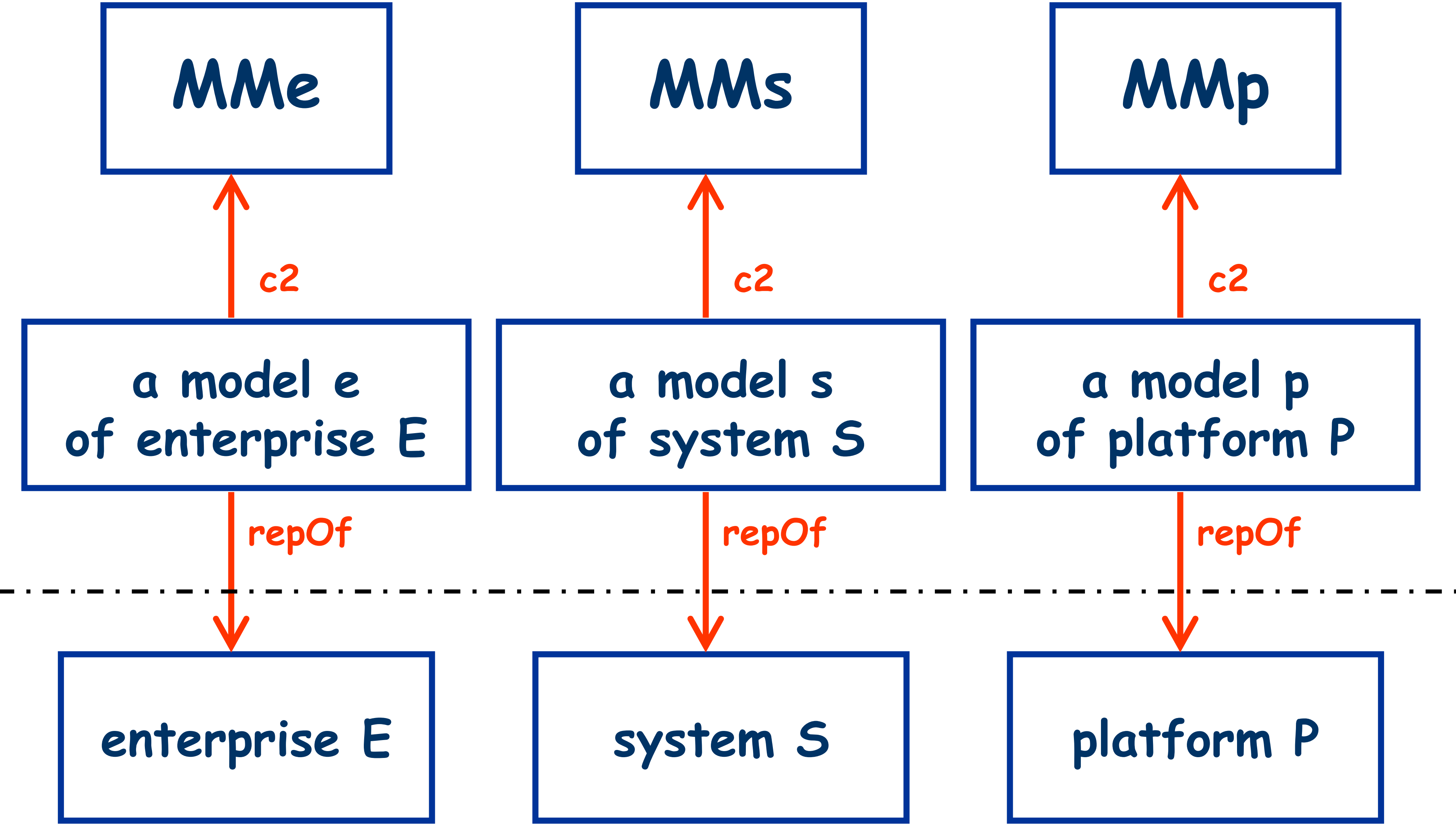
PSMs (Platform Specific Models)

PSM = f(PIM, PDM)

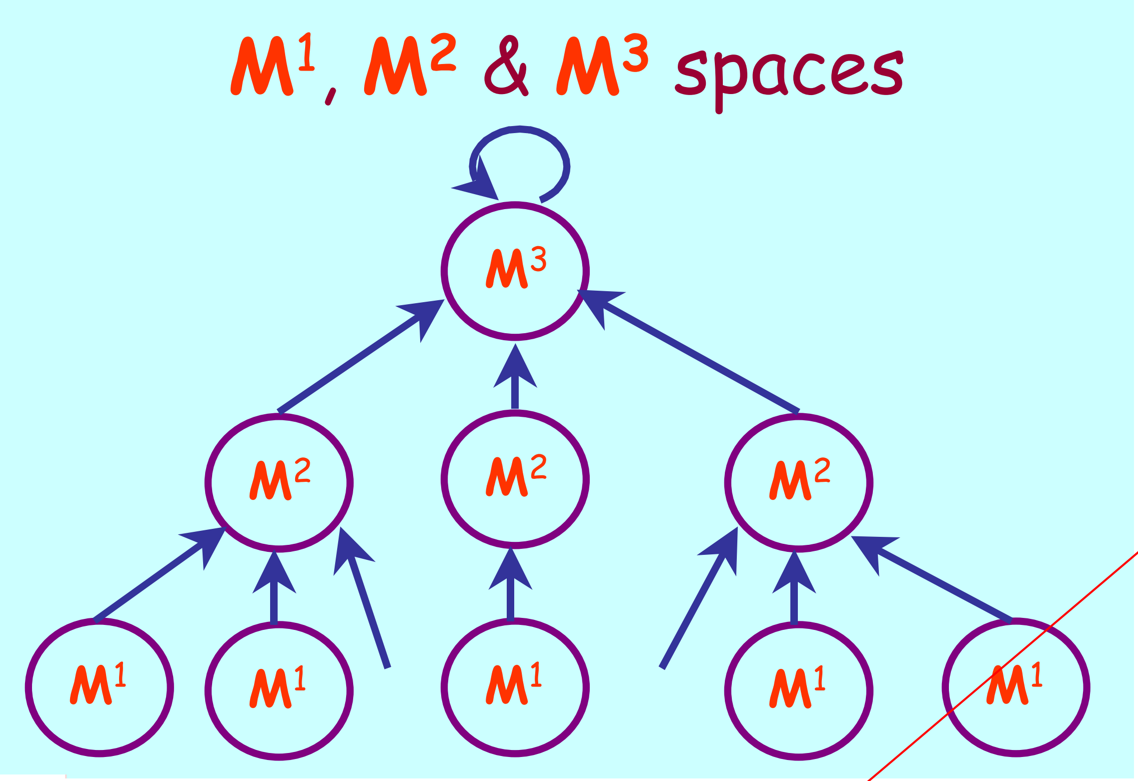
or by currying:

PSM = f_{PDM}(PIM)

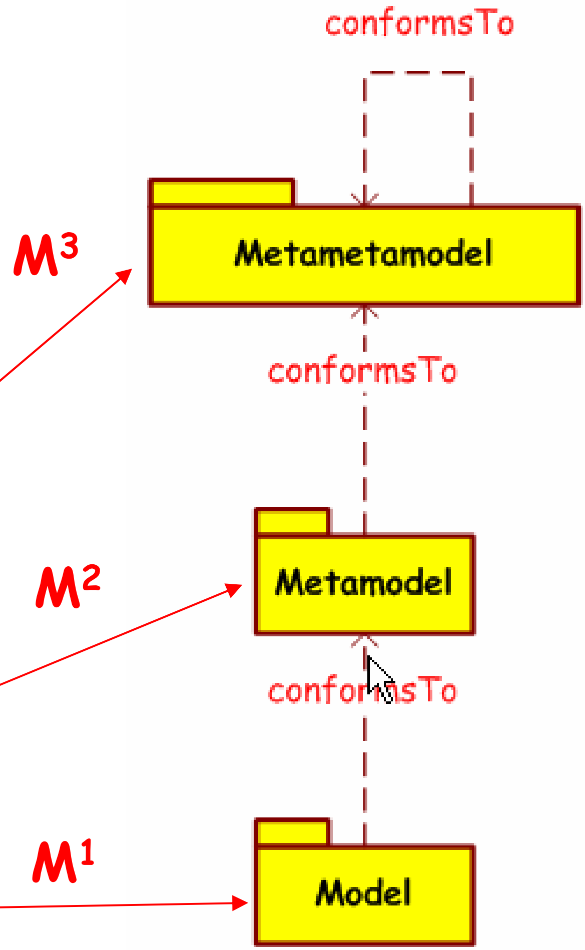
Abstract models of Platforms and Enterprises



MDA in a nutshell



M^1, M^2 & M^3 spaces



- One unique Metametamodel (the MOF)
- An important library of compatible Metamodels, each defining a DSL
- Each of the models is defined in the language of its unique metamodel

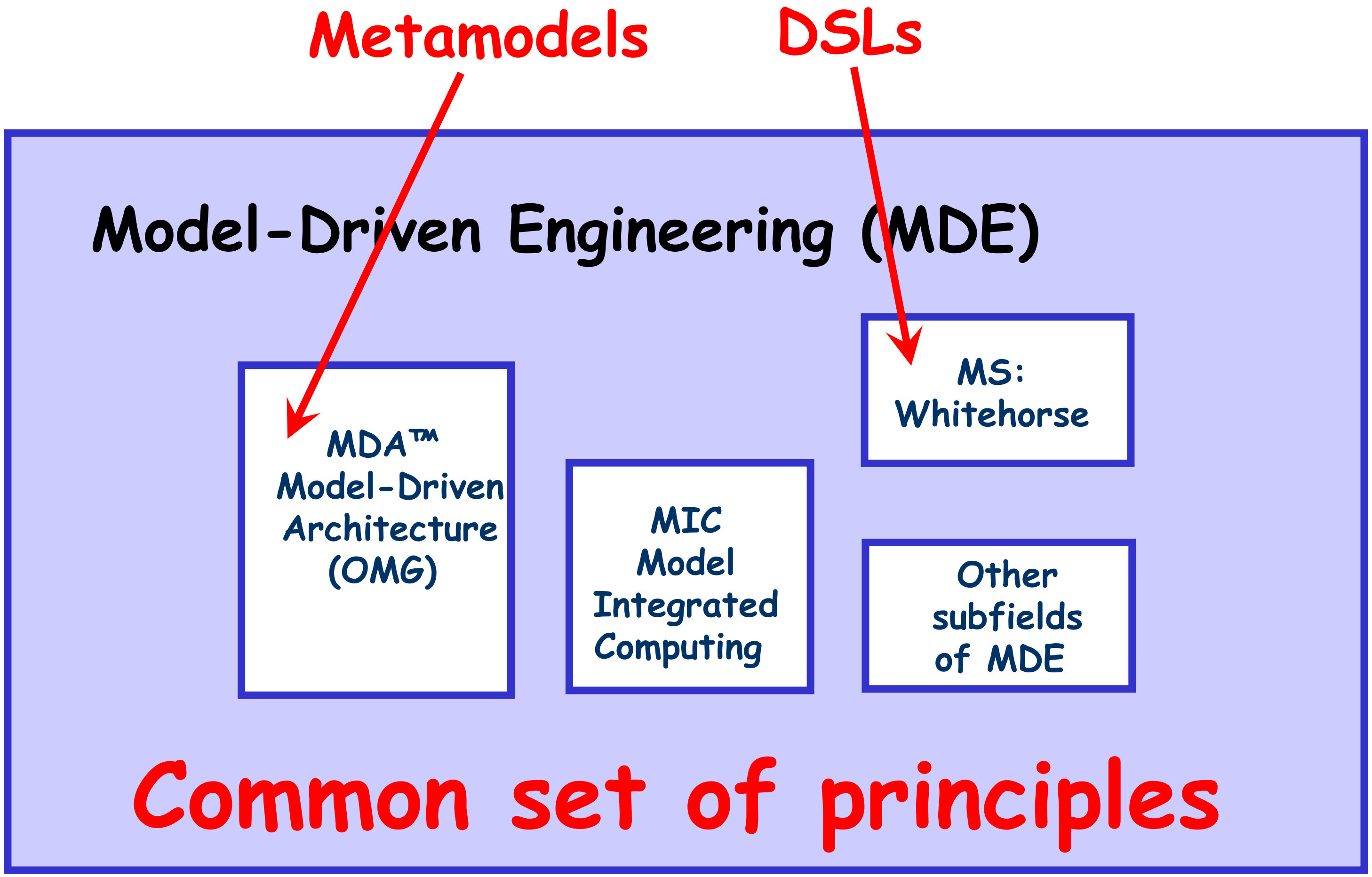
A definition of MDA

OMG/ORMSC/2004-06-01 (The OMG MDA Guide): A Definition of MDA (The following was approved unanimously by 17 participants 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. Not all of these technologies will directly concern the transformations involved in MDA.
- 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.

MDE and the MDA™

"MDA™ is a specific MDD™ deployment effort around such industrial standards as MOF, UML, CWM, QVT, etc." (from OMG/MDA guide).



Metamodels

DSLs

Model-Driven Engineering (MDE)

MDA™
Model-Driven
Architecture
(OMG)

MIC
Model
Integrated
Computing

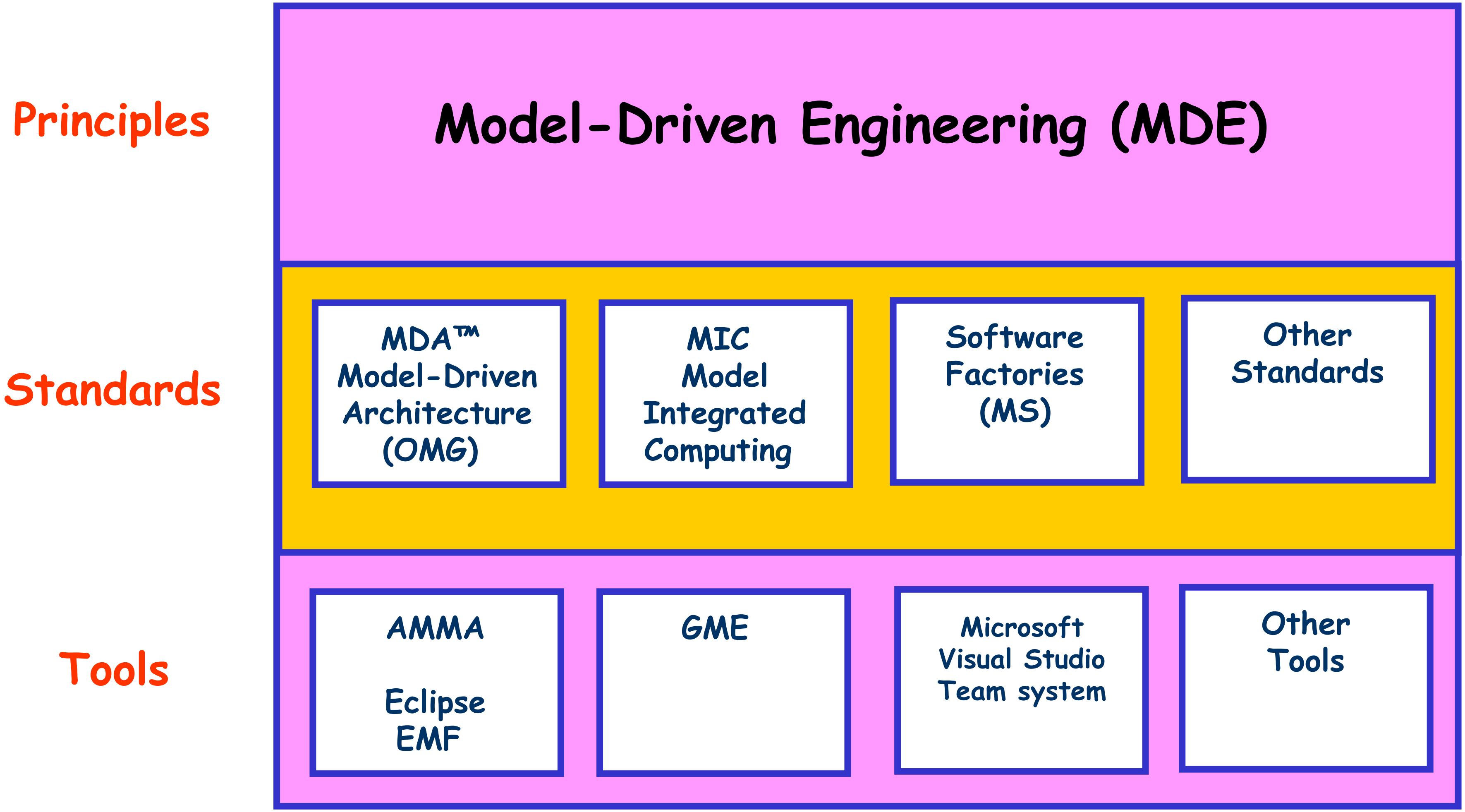
MS:
Whitehorse

Other
subfields
of MDE

Common set of principles

Terminological note: MDA and MDD are trademarks of OMG; MDE is not.

Principles, standards and tools



Principles

Model-Driven Engineering (MDE)

Standards

MDA™ Model-Driven Architecture (OMG)

MIC Model Integrated Computing

Software Factories (MS)

Other Standards

Tools

AMMA

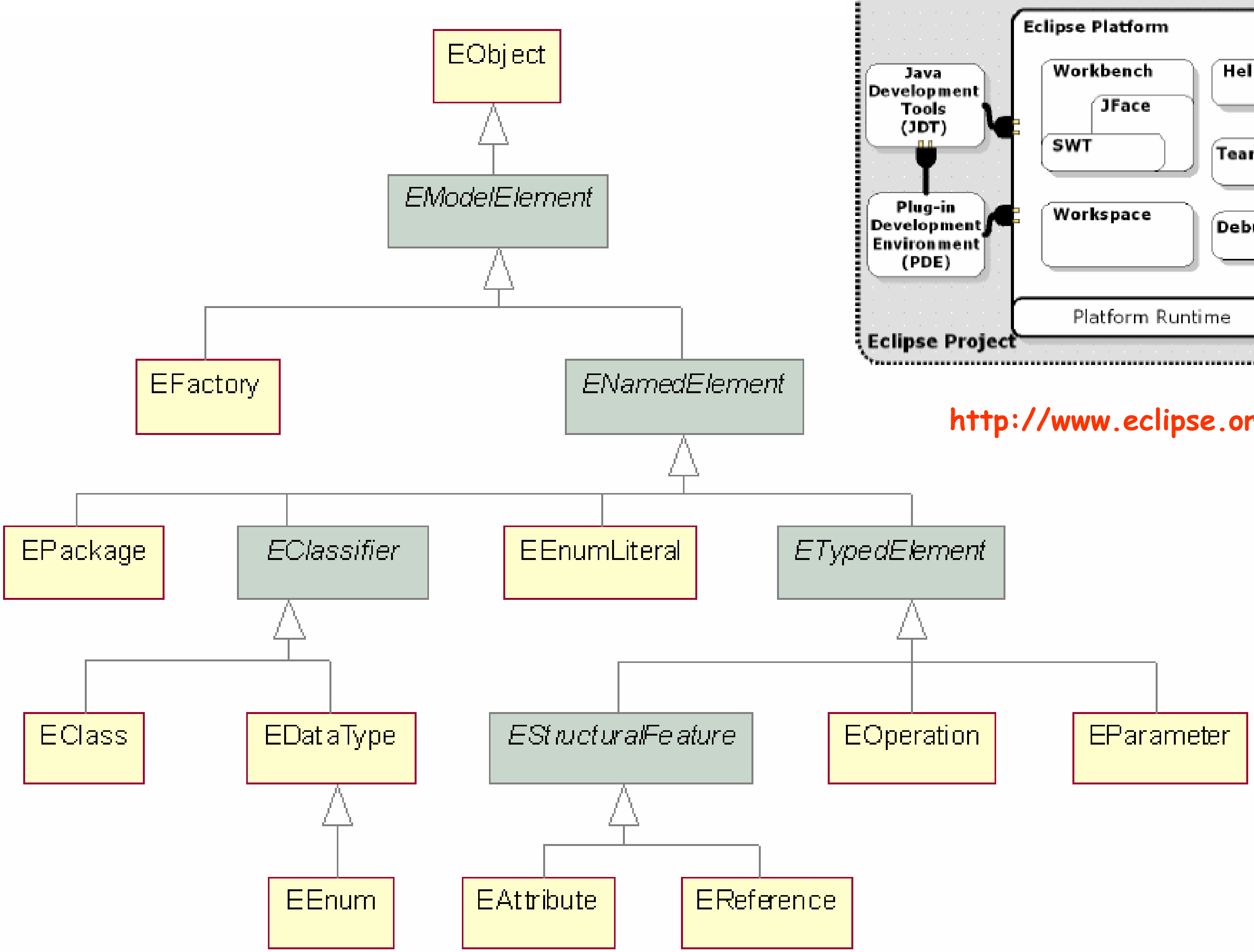
Eclipse EMF

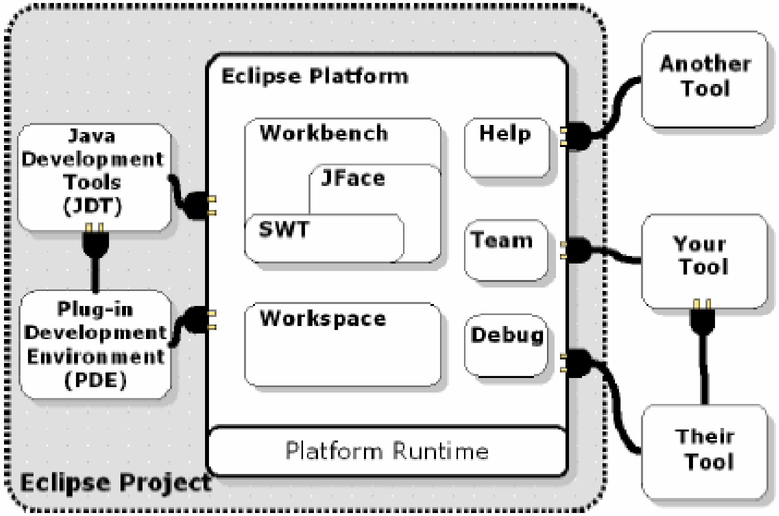
GME

Microsoft Visual Studio Team system

Other Tools

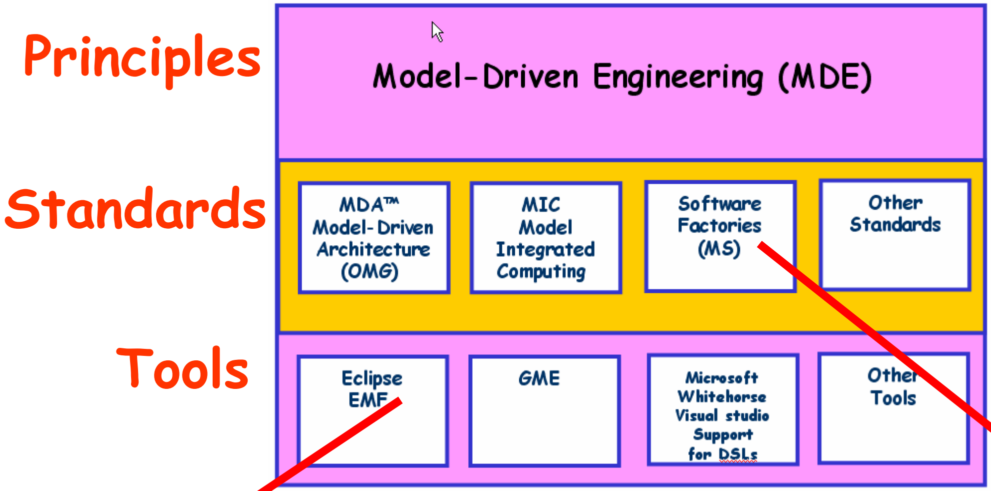
EMF: The ECORE metametamodel





http://www.eclipse.org/emf/

A very active industrial development area



Principles

Model-Driven Engineering (MDE)

Standards

MDA™ Model-Driven Architecture (OMG)

MIC Model Integrated Computing

Software Factories (MS)

Other Standards

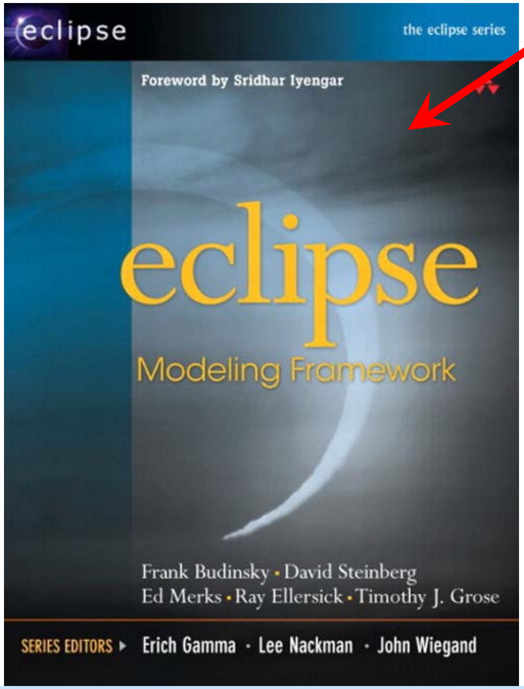
Tools

Eclipse EMF

GME

Microsoft Whitehorse Visual studio Support for DSLs

Other Tools



eclipse

the eclipse series

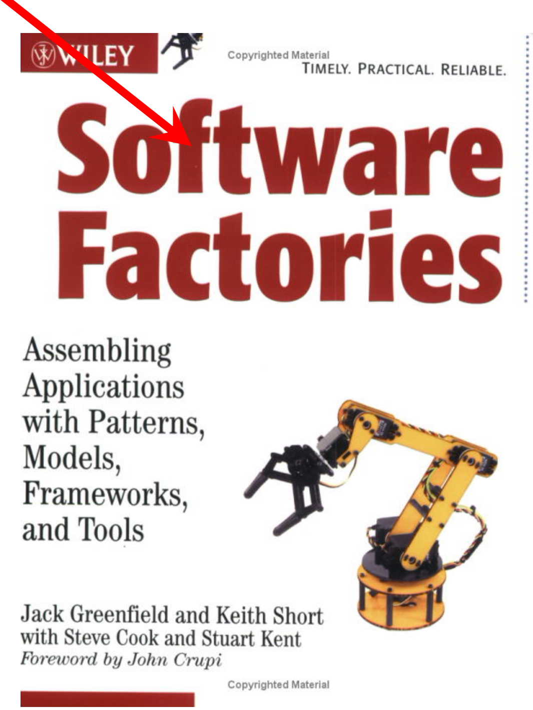
Foreword by Sridhar Iyengar

eclipse

Modeling Framework

Frank Budinsky • David Steinberg
Ed Merks • Ray Ellersick • Timothy J. Grose

SERIES EDITORS ▸ Erich Gamma • Lee Nackman • John Wiegand



WILEY

Copyrighted Material

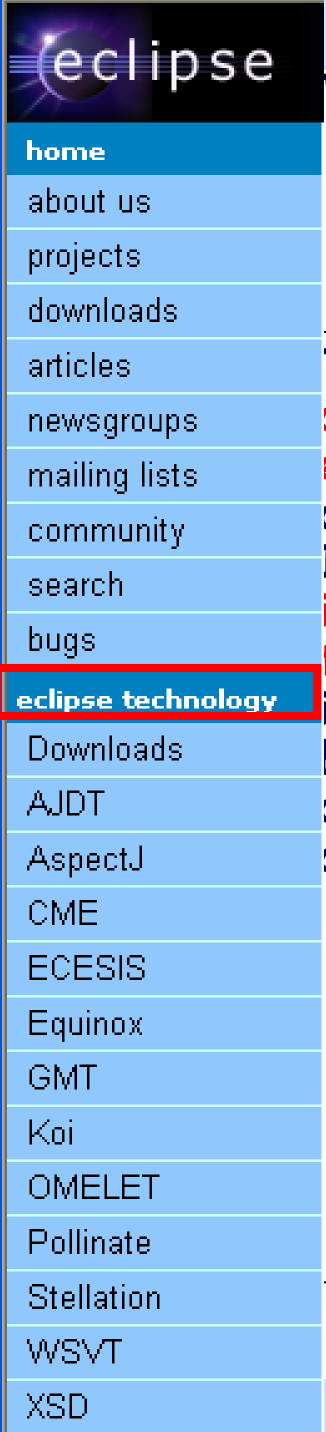
TIMELY. PRACTICAL. RELIABLE.

Software Factories

Assembling Applications with Patterns, Models, Frameworks, and Tools

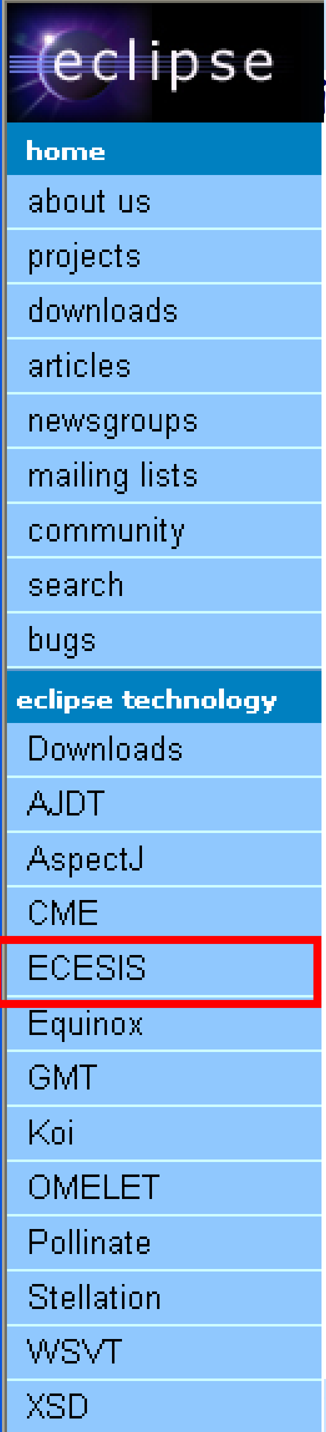
Jack Greenfield and Keith Short
with Steve Cook and Stuart Kent
Foreword by John Crupi

Copyrighted Material



Technology Project

The mission of the Eclipse Technology Project is to provide new channels for open source developers, researchers, academics and educators to participate in the on-going evolution of Eclipse. It is organized as three related project streams, namely Research, Incubators and Education. Research projects explore research issues in Eclipse-relevant domains such as programming languages, tools, and development environments. Incubators are small, informally structured projects which add new capabilities to the Eclipse software base. Education projects focus on the development of educational materials, teaching aids and courseware.



eclipse

home

about us

projects

downloads

articles

newsgroups

mailing lists

community

search

bugs

eclipse technology

Downloads

AJDT

AspectJ

CME

ECESIS

Equinox

GMT

Koi

OMELET

Pollinate

Stellation

WSVT

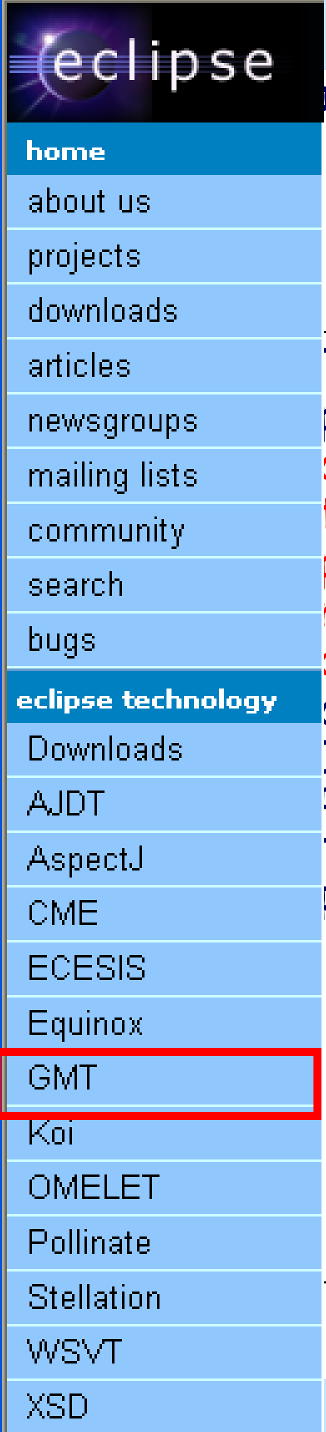
XSD

ipse Community Education Project)

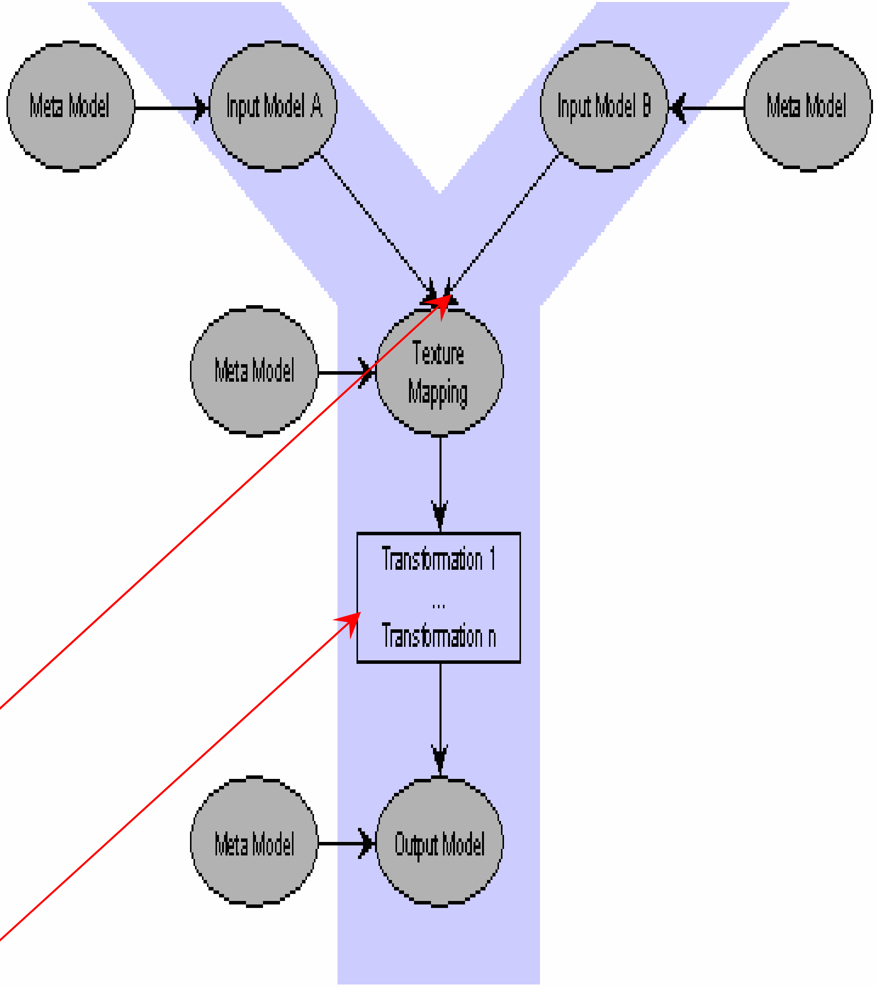
✓ The goal of the Eclipse Community Education Project (ECESIS) is to **promote the creation, improvement and distribution of commercial and academic quality Eclipse courseware, education and training technologies, and resource material**. ECESIS's basic objectives are as follows:

- ❑ To develop a starting set of high quality Eclipse courseware, tools, and non-course educational resources
- ❑ To distribute freely all educational material under an open source license
- ❑ To encourage the production and development of further material
- ❑ To encourage and promote the widest possible use of such material

ative Model Transformer)



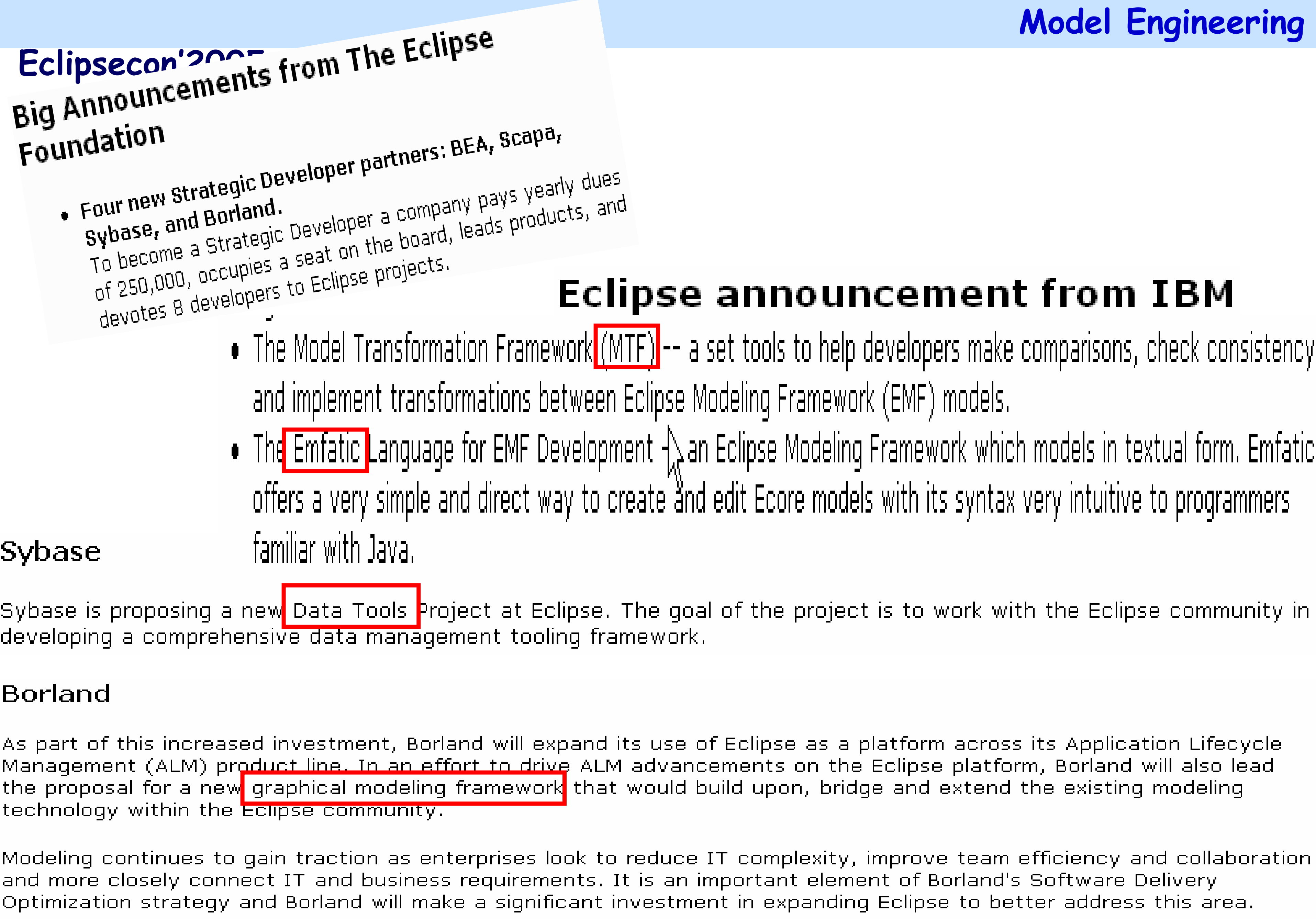
The goal of this technology project is to construct/assemble a set of tools for the Eclipse platform which support model driven software development with fully customisable Platform Independent Models, Platform Description Models, Texture Mappings, and Refinement Transformations.



model Weaving

model Transformation

Eclipsecon'2005



Big Announcements from The Eclipse Foundation

- **Four new Strategic Developer partners: BEA, Scapa, Sybase, and Borland.**
  To become a Strategic Developer a company pays yearly dues of 250,000, occupies a seat on the board, leads products, and devotes 8 developers to Eclipse projects.

Eclipse announcement from IBM

- The Model Transformation Framework (MTF) -- a set tools to help developers make comparisons, check consistency and implement transformations between Eclipse Modeling Framework (EMF) models.
- The Emfatic Language for EMF Development -- an Eclipse Modeling Framework which models in textual form. Emfatic offers a very simple and direct way to create and edit Ecore models with its syntax very intuitive to programmers familiar with Java.

Sybase

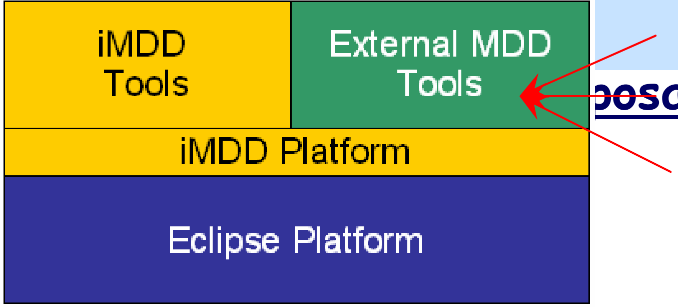
Sybase is proposing a new Data Tools Project at Eclipse. The goal of the project is to work with the Eclipse community in developing a comprehensive data management tooling framework.

Borland

As part of this increased investment, Borland will expand its use of Eclipse as a platform across its Application Lifecycle Management (ALM) product line. In an effort to drive ALM advancements on the Eclipse platform, Borland will also lead the proposal for a new graphical modeling framework that would build upon, bridge and extend the existing modeling technology within the Eclipse community.

Modeling continues to gain traction as enterprises look to reduce IT complexity, improve team efficiency and collaboration and more closely connect IT and business requirements. It is an important element of Borland's Software Delivery Optimization strategy and Borland will make a significant investment in expanding Eclipse to better address this area.

posals



iMDD is supported by: ModelWare

- **iMDD (integrated Model Driven Development)**
  - The Eclipse Integrated Model Driven Development (iMDD) project is dedicated to the realization of a platform offering facilities needed for applying a Model Driven Development (MDD) approach. This platform will provide a consistent set of tools for modeling Domain Specific Languages (DSL) and for generating a dedicated environment (GUI, checks, generators, compilers, simulators…) for these specific DSL.
  - The Eclipse platform is an excellent basis on which to build and integrate Model Driven Development tools. The iMDD platform will extend the Eclipse platform with an additional level of integration for MDD tools providing and requiring services that operate over models or model elements.
- **GMF (Graphical Modeling Framework) (GMF =**
  - Borland is taking a bigger stake in the organization it helped found, joining the Eclipse Foundation's board of directors as a strategic developer, officials will announce in a press conference Monday at EclipseCon 2005 in Burlingame, California
  - Raaj Shinde, Borland vice president of product strategy and architecture, said the proposal, which hasn't been formally submitted to the Eclipse Foundation, builds on an area they have some expertise with, he said, as Together has been running on Eclipse since 2002. The project has a good chance of looking like a port of its commercial product.
  - Skip McGaughey, an Eclipse spokesman, said Borland's installment on the board of directors will add significant expertise and energy to the Eclipse Foundation, and doesn't expect the company to have any difficulties getting its project accepted within the organization. It's too early to tell, he said, whether the project will remain a sub-project or become a top-level project down the road.

Microsoft Whitehorse, etc.

- SDK (Team System) released in late 2004
  - First presentation at OOPSLA, Vancouver, Oct. 2004
  - See S. Cook, S. Kent and K. Short Blogs
- Aims to be closer to a metaCASE tool than Eclipse
- Not UML-based
- Models strongly tied to code
  - Reverse engineering/synchronization
  - Reliance on Microsoft's platforms (Visual studio)

… Modeling is the future …

Bill Gates

Whitehorse : Bill Gates on models

… Modeling is the future …

You know UML [Unified Modeling Language] made the meta-models a little complex, so I don't think UML alone is the answer …

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 …

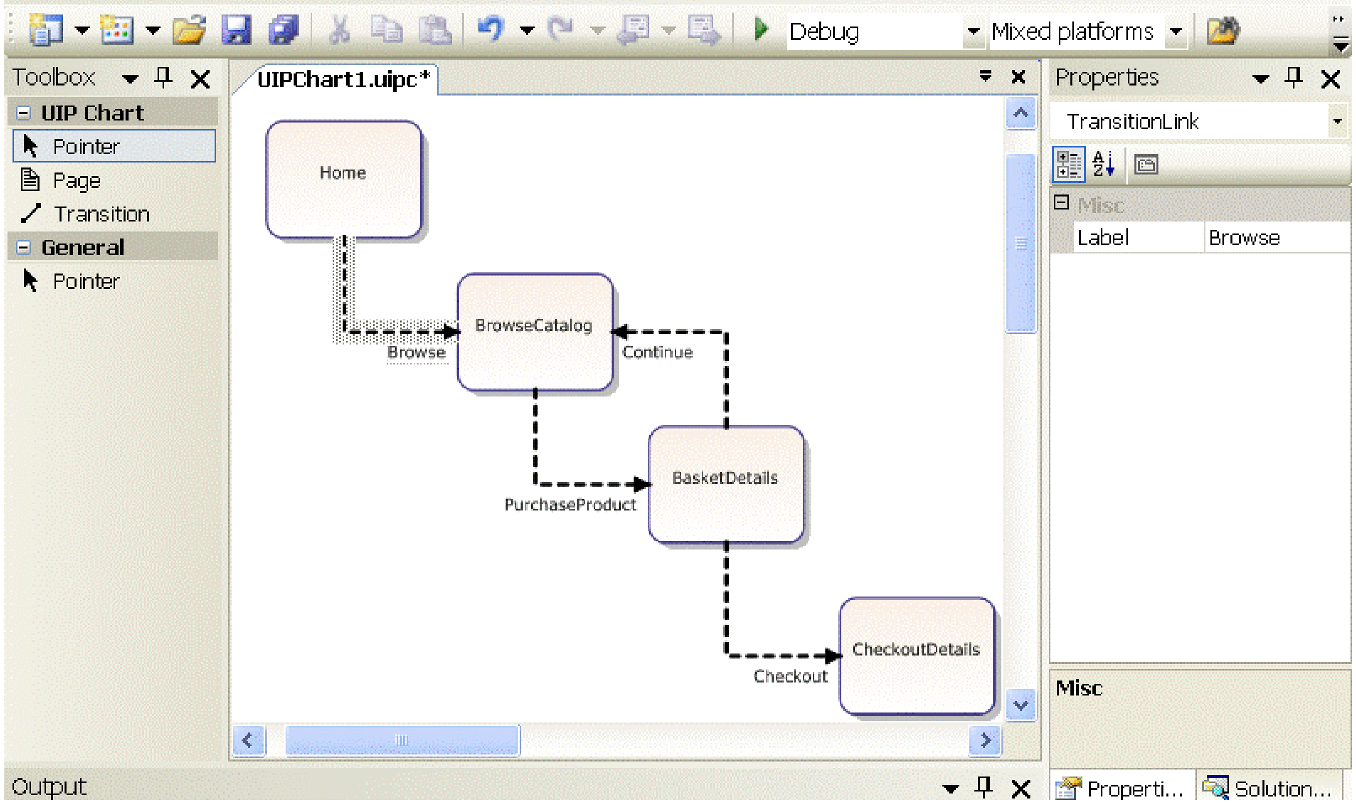
So even a business could express in a formal, modeled way, not just scribbling on paper, how the business process is changing over time or how it's different from other companies. So instead of having lots of code behind that, you just have visual, essentially model, customization …

So, I think we believe that. There are certainly some people from IBM who have that same vision, and I think it'll be healthy competition between the two of us because today's modeling products fall short. That's one part of Visual Studio 2005, that we do have some neat things coming along that will be part of it that we haven't shown completely …

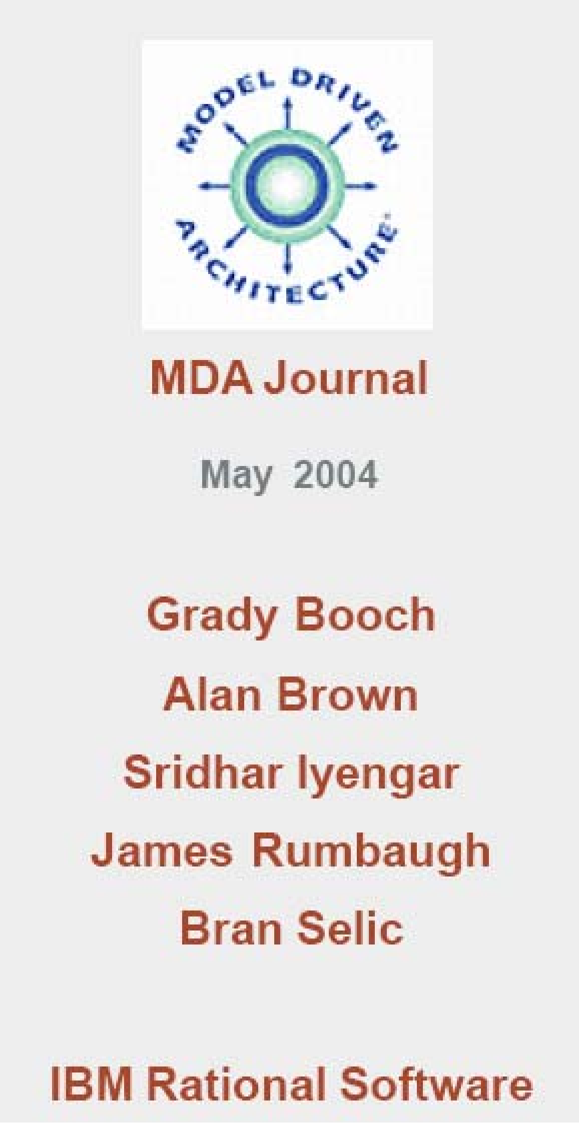
So, modeling is pretty magic stuff, whether it's management problems or business customization problems or work-flow problems, visual modeling. Even the Office group now really gets that for document life-cycle rights management, that this visual modeling will be key to them. Business intelligence, where you let people navigate through things, is another area where modeling could be used. It's probably the biggest thing going on. And both Visual Studio and Office need to be on top of that …

MDE@Microsoft

- Microsoft is releasing a suite of tools to make it easy to construct graphical designers hosted in Visual Studio for editing domain specific languages (DSL).



IBM on MDA : Three complementary ideas



MDA Journal

May 2004

Grady Booch

Alan Brown

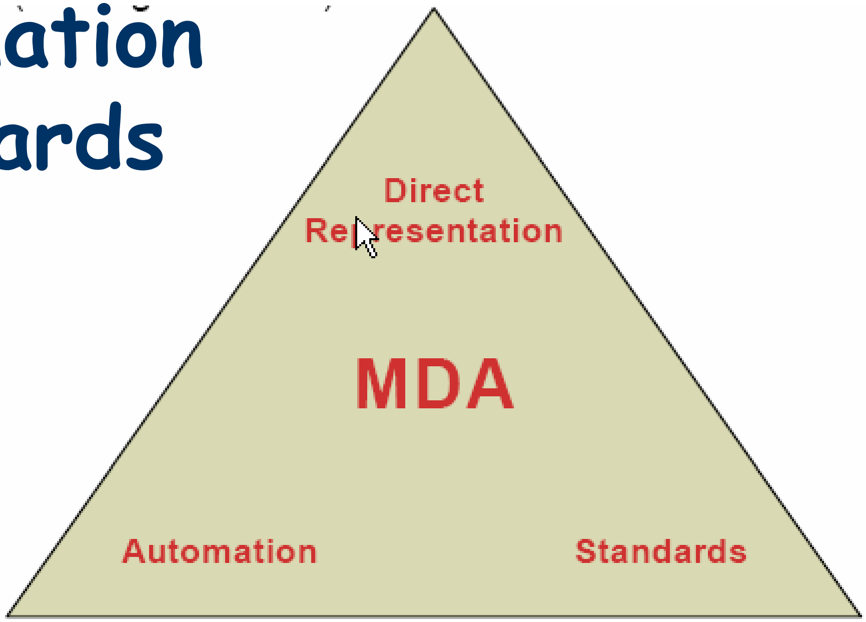
Sridhar Iyengar

James Rumbaugh

Bran Selic

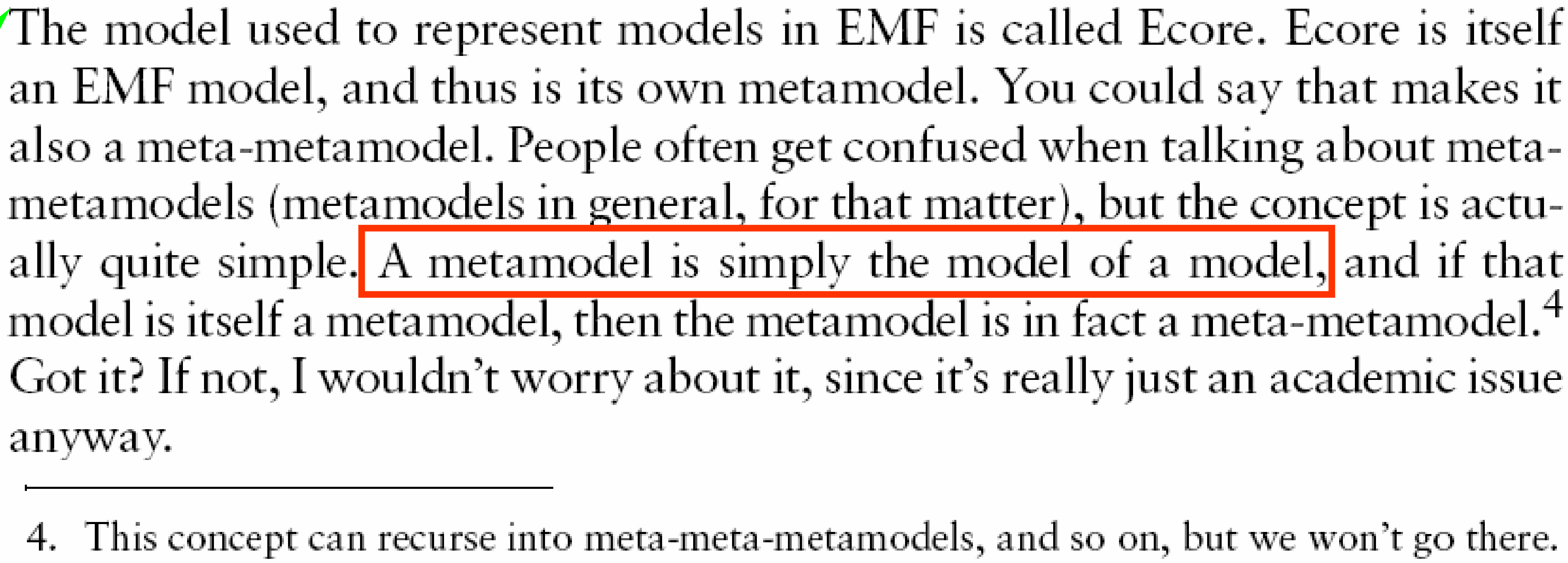
IBM Rational Software

1. Direct representation
2. Automation
3. Standards



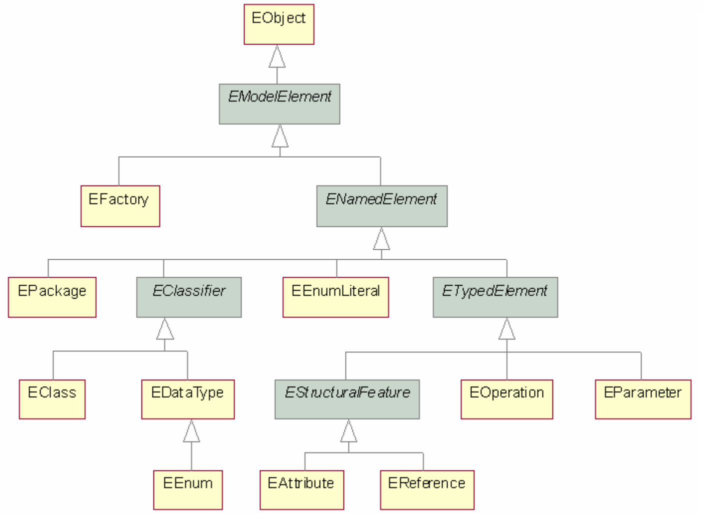
Direct representation => multiple languages
Danger of fragmentation
Need coordination
How to coordinate?
Short answer: metametamodel
But what is a metametamodel?

Just an academic issue anyway?

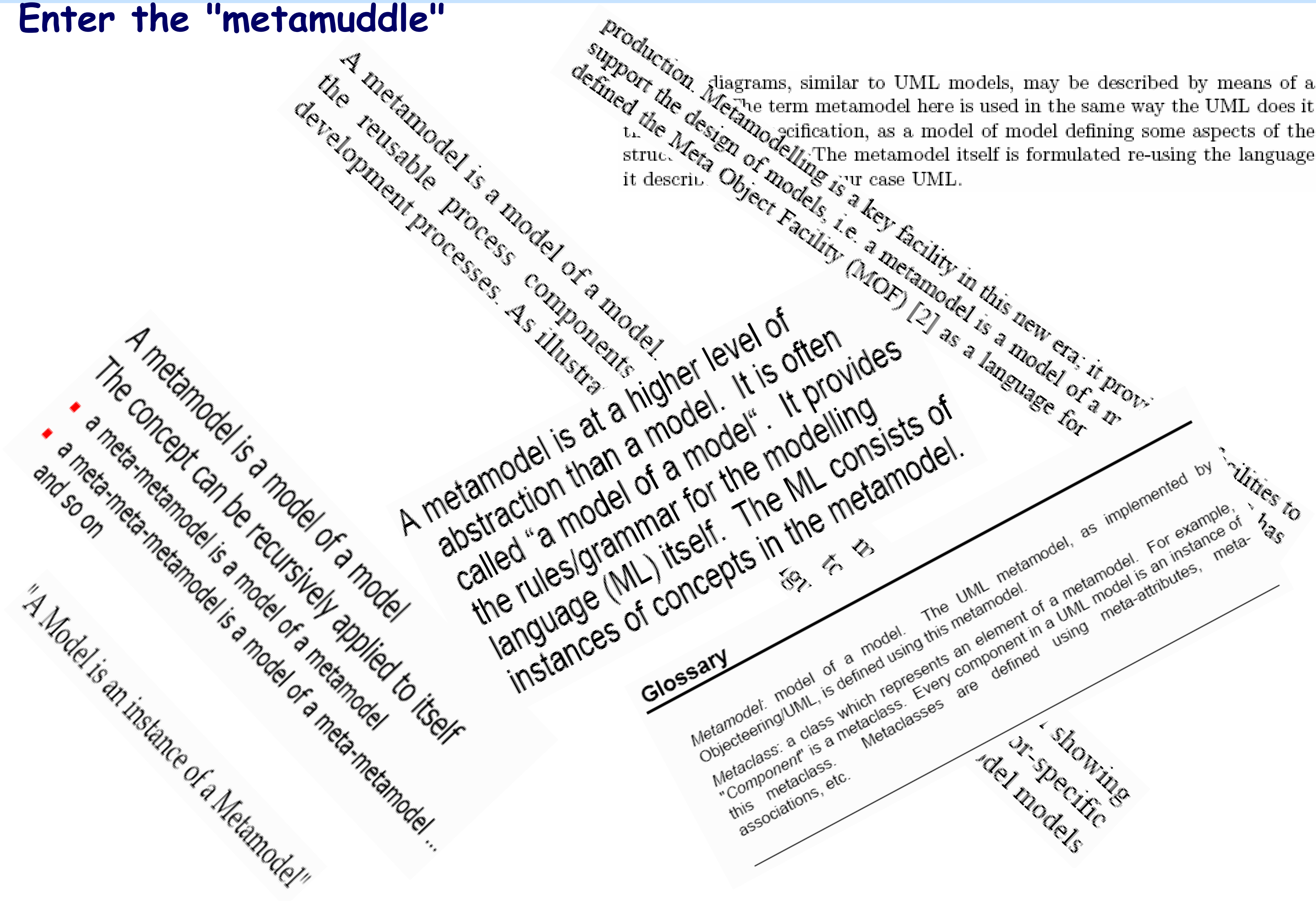


The model used to represent models in EMF is called Ecore. Ecore is itself an EMF model, and thus is its own metamodel. You could say that makes it also a meta-metamodel. People often get confused when talking about meta-metamodels (metamodels in general, for that matter), but the concept is actually quite simple. A metamodel is simply the model of a model, and if that model is itself a metamodel, then the metamodel is in fact a meta-metamodel.[4] Got it? If not, I wouldn't worry about it, since it's really just an academic issue anyway.

[4] This concept can recurse into meta-meta-metamodels, and so on, but we won't go there.



Enter the "metamuddle"



production. diagrams, similar to UML models, may be described by means of a ... The term metamodel here is used in the same way the UML does it ... specification, as a model of model defining some aspects of the struc... The metamodel itself is formulated re-using the language it describ... our case UML.

production. Metamodelling is a key facility in this new era; it provi... support the design of models, i.e. a metamodel is a model of a m... defined the Meta Object Facility (MOF) [2] as a language for

A metamodel is a model of a model. the reusable process components development processes. As illustra

A metamodel is at a higher level of abstraction than a model. It is often called "a model of a model". It provides the rules/grammar for the modelling language (ML) itself. The ML consists of instances of concepts in the metamodel.

A metamodel is a model of a model

The concept can be recursively applied to itself

- a meta-metamodel is a model of a metamodel
- a meta-meta-metamodel is a model of a meta-metamodel ...

and so on

"A Model is an instance of a Metamodel"

Glossary

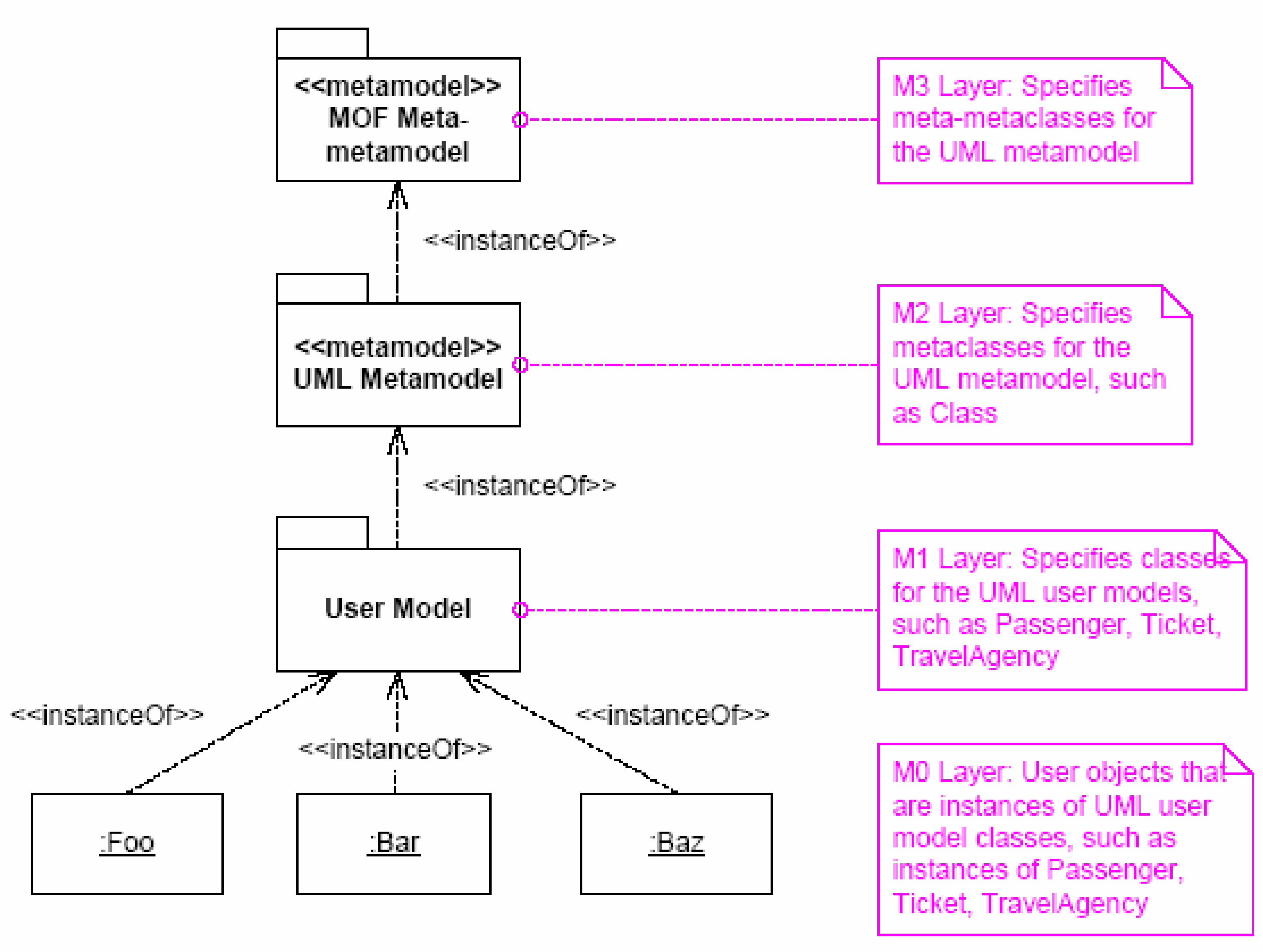
Metamodel: model of a model. The UML metamodel, as implemented by Objecteering/UML, is defined using this metamodel.

Metaclass: a class which represents an element of a metamodel. For example, "*Component*" is a metaclass. Every component in a UML model is an instance of this metaclass. Metaclasses are defined using meta-attributes, meta-associations, etc.

...ilities to ... has

... showing ...or-specific ...del models

The standard "official" OMG stack (mimicking OT)



M3 Layer: Specifies meta-metaclasses for the UML metamodel

M2 Layer: Specifies metaclasses for the UML metamodel, such as Class

M1 Layer: Specifies classes for the UML user models, such as Passenger, Ticket, TravelAgency

M0 Layer: User objects that are instances of UML user model classes, such as instances of Passenger, Ticket, TravelAgency

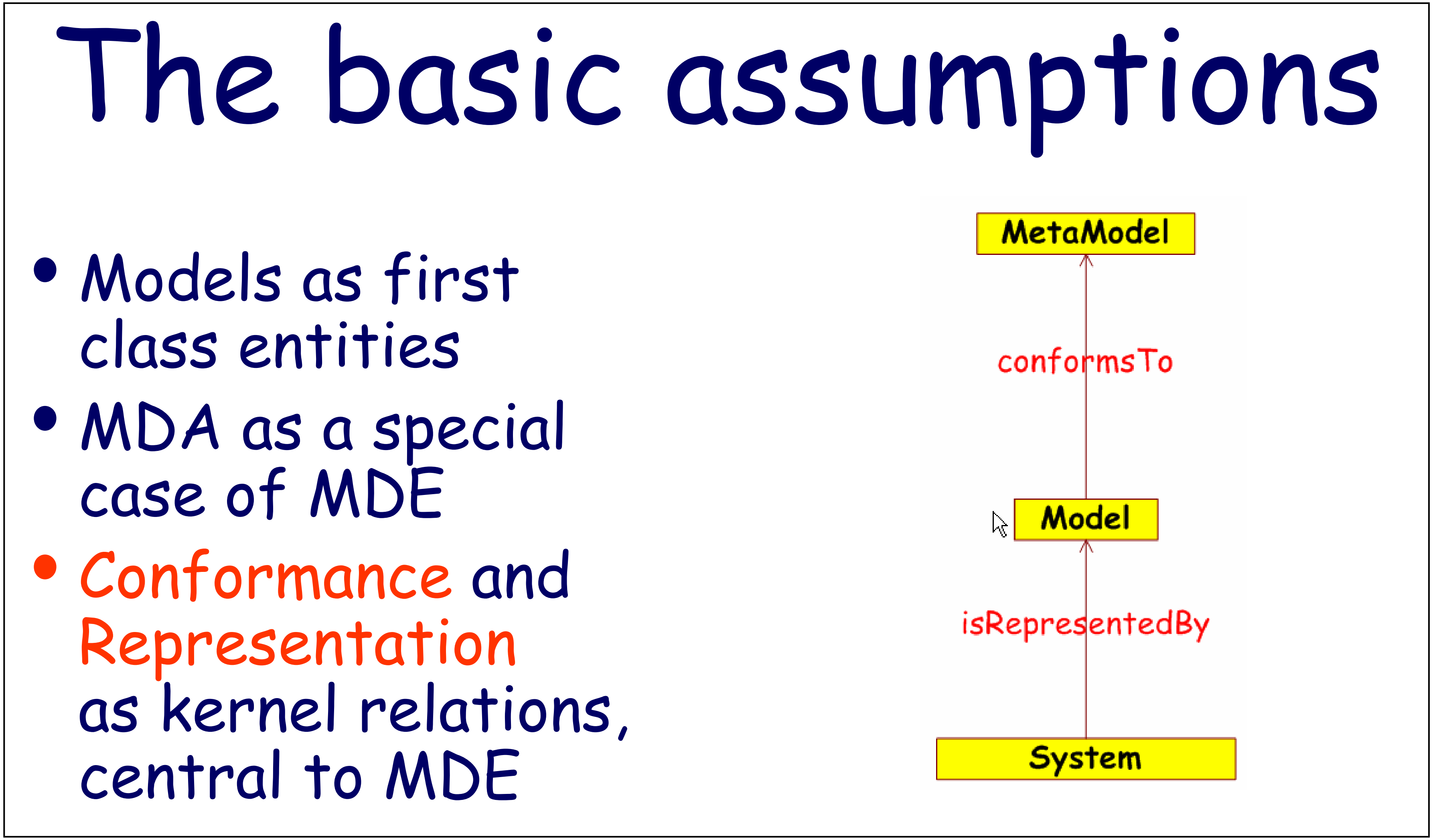
The metamuddle

- A very rapidly growing industrial application field since november 2000,
- … but …
- We badly need a unifying theory of models

Agenda

In search of sound principles for MDE

The basic assumptions

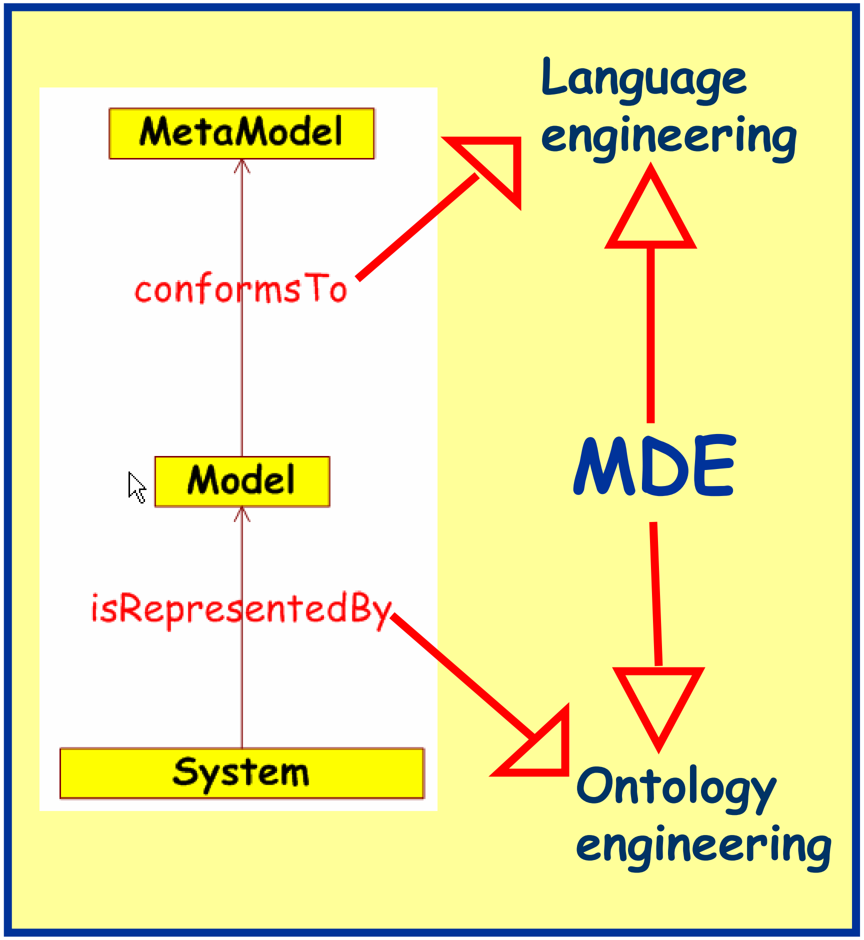


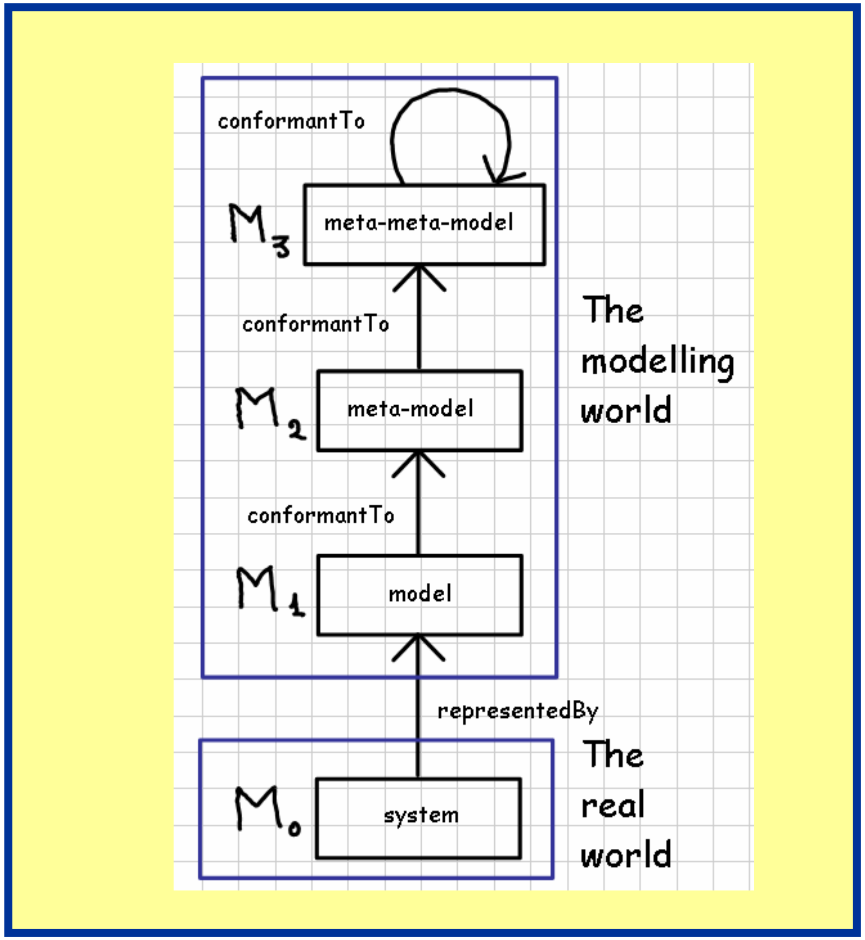
The basic assumptions

- Models as first class entities
- MDA as a special case of MDE
- Conformance and Representation as kernel relations, central to MDE

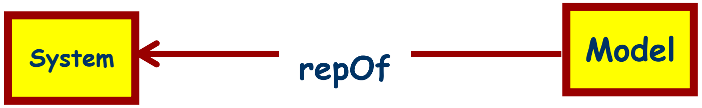
MetaModel
conformsTo
Model
isRepresentedBy
System

Credits and MDA compliance

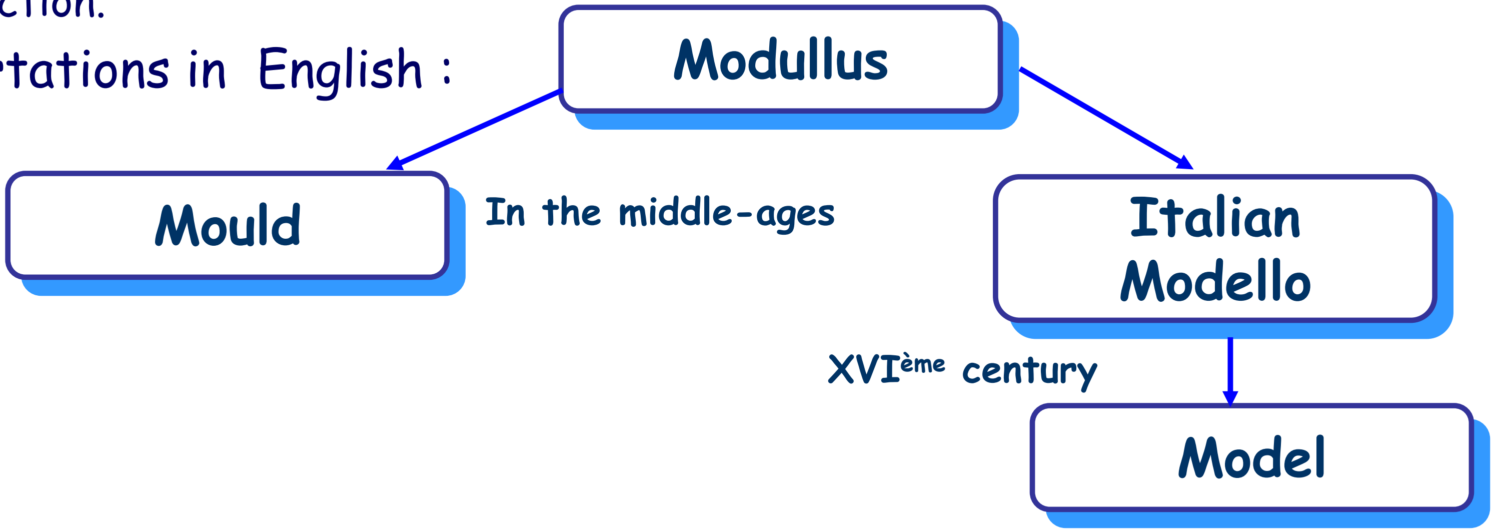




Definitions



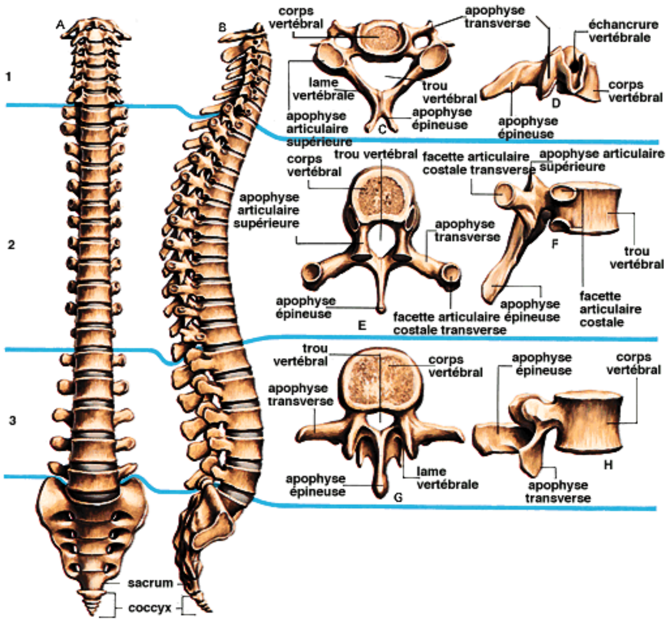
- A model is the simplified image of a system
  - This short definition should be completed
- What is a system ?
  - "A system is a set of elements in interaction " (von Bertalanffy)
  - The word system comes from the Greek "sun-istémi" (I compose)
- Model comes from the Latin "modullus", diminutive of "modus" (measure)
  - Initially it was an architectural term meaning an arbitrary measure used for establishing various ratios between different parts of a building in construction.
- Two importations in English :

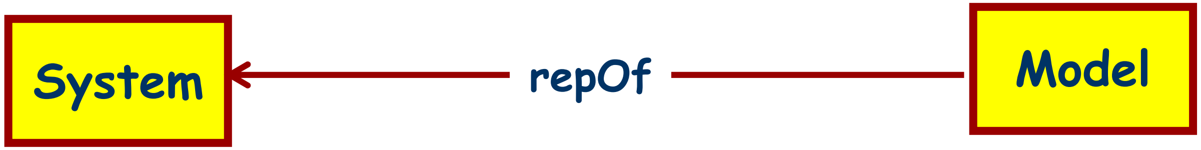


The word is recent, the idea is old

Plato (427-347 before JC), in *Timeus* compares vertebras to door hinges (74a) or blood vessels to irrigation channels.

This idea will be used again later by the english physiologist William Harvey (1578-1657) who will discover the blood circulation principle:
"de ce que, dans le cœur des vivants,
les valvules semblent être des soupapes ou des portes d'écluse".





Model: multiple definitions

MSN Encarta

mod·el [móddʼl] noun (plural **mod·els**)

1. **copy of an object:** a copy of an object, especially one made on a smaller scale than the original (*often used before a noun*)

2. **particular version of manufactured article:** a particular version of a manufactured article
had traded in her car for the latest model

3. **something copied:** something that is copied or used as the basis for a related idea, process, or system

4. **somebody paid to wear clothes:** somebody who is paid to wear clothes and demonstrate merchandise as a profession, for example, in fashion shows and photographs for magazines and catalogues

5. **simplified version:** a simplified version of something complex used, for example, to analyze and solve problems or make predictions *a financial model*

6. **perfect example:** an excellent example that deserves to be imitated

7. **artist's subject:** somebody who poses for a painter, sculptor, photographer, or other artist

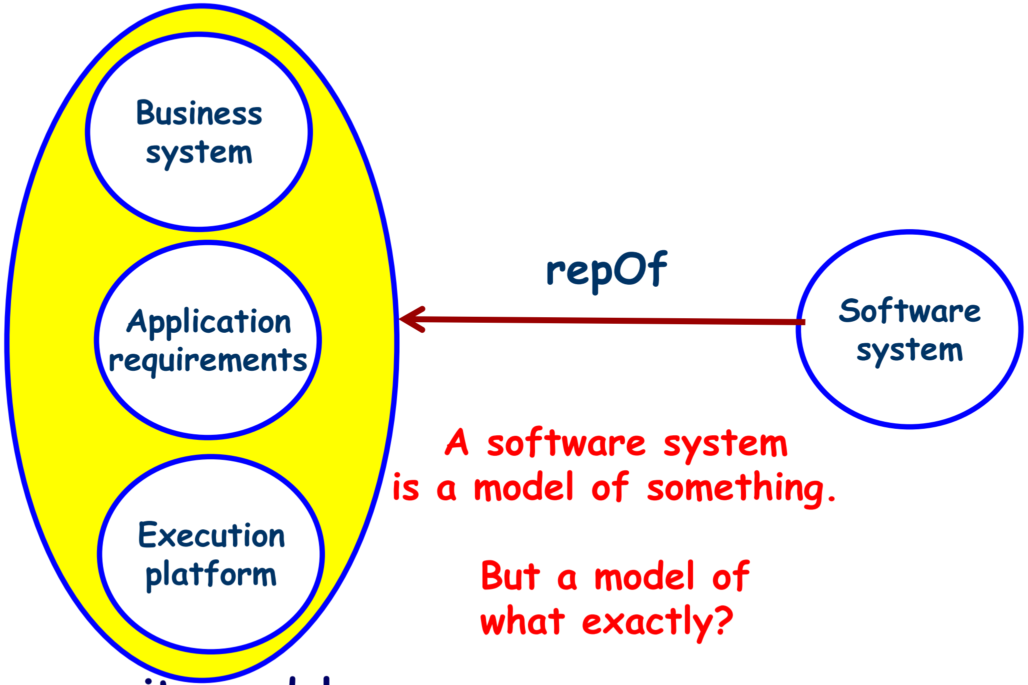
8. zoology **animal copied by another animal:** an animal species repellent to predators which another animal mimics for protection

9. logic **interpretation:** an interpretation of a theory arrived at by assigning referents in such a way as to make the theory true

10. *U.K.* fashion **exclusive garment:** the first sewn example of a couturier's or clothing manufacturer's design, from which a new line of garments is produced

Differents kinds of models

- Numerous examples
  - Mathematical models
  - Hydrological models
  - Biological models
  - Ecological models
  - Economical models
  - Meteorological models
  - Simulation models
  - Descriptive or predictive models
  - etc.
- Software are models:
  - **A software is a complex and composite model**
  - **But a model of what?**
    - Of the organization where it is supposed to function?
    - Of the architecture (hardware/software platform) on top of which it is supposed to function?
    - Of the applicative requirements it is supposed to satisfy ?
    - Of the team that elaborated the software in a given process?
  - **Expressed in which language?**
    - Until now software was mainly written in so-called programming languages like C# or Java
    - … but things are rapidly changing (code-centric to model-centric, DSLs)



A software system is a model of something.

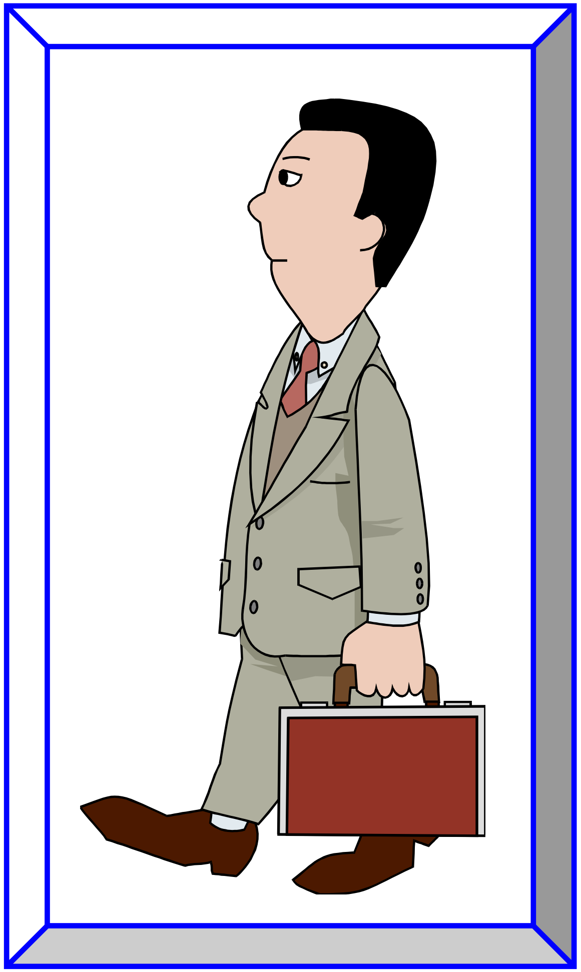
But a model of what exactly?

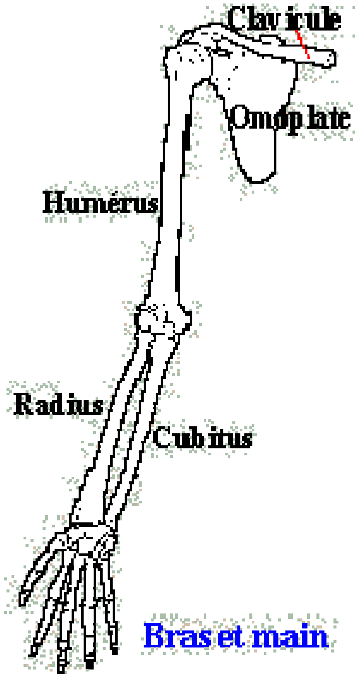
A model is a view on a system

A system

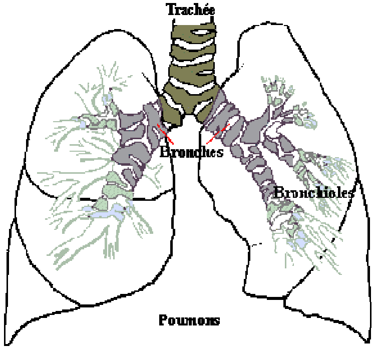
repOf

Several models of this system (partial views)





Skeleton model



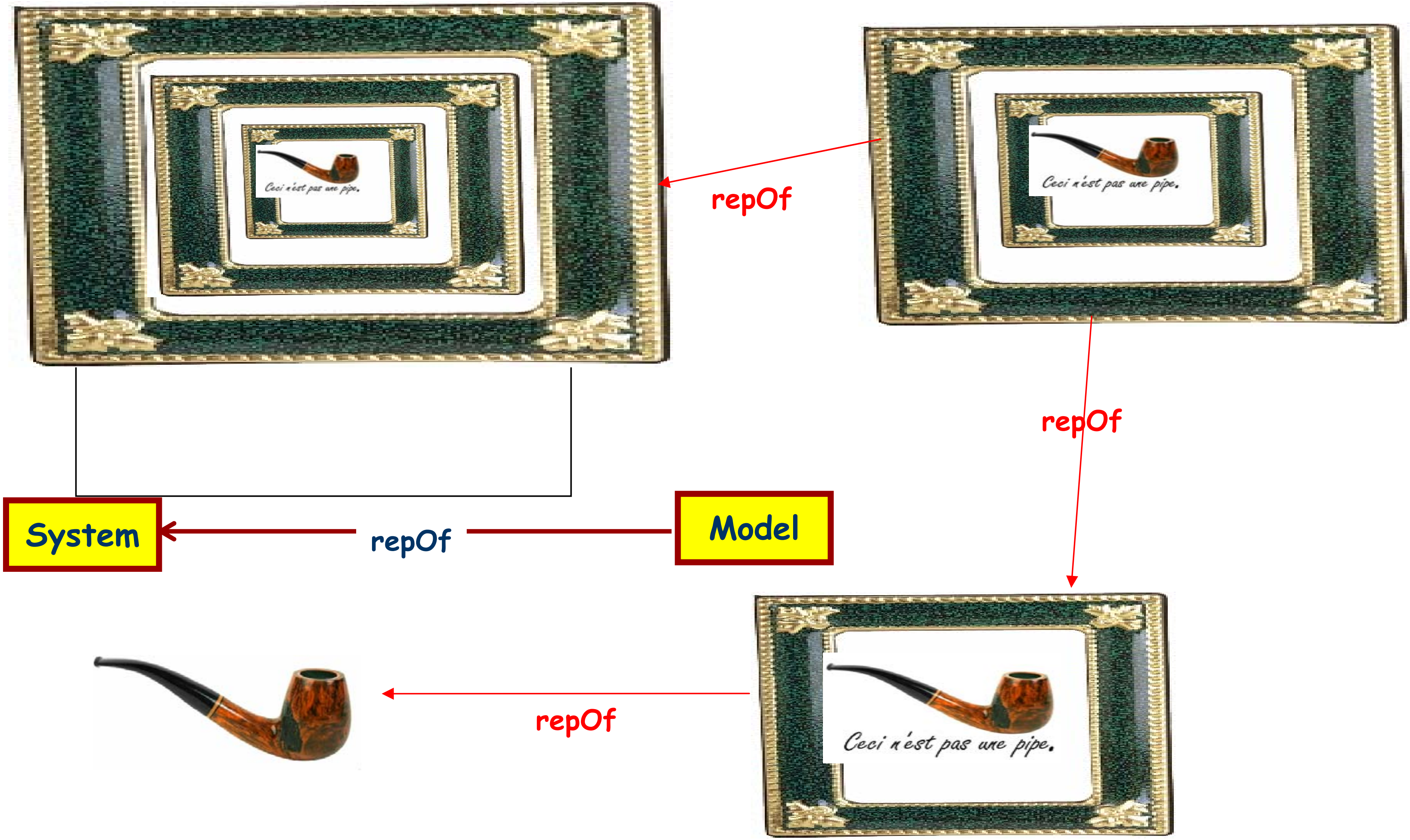
Respiratory model

Other Models muscular, nervous, circulatory, digestive, endocrinous, etc.

Don't confuse the model and the system

This is not a pipe by Magritte

Don't confuse the model and the system



repOf

repOf

System

repOf

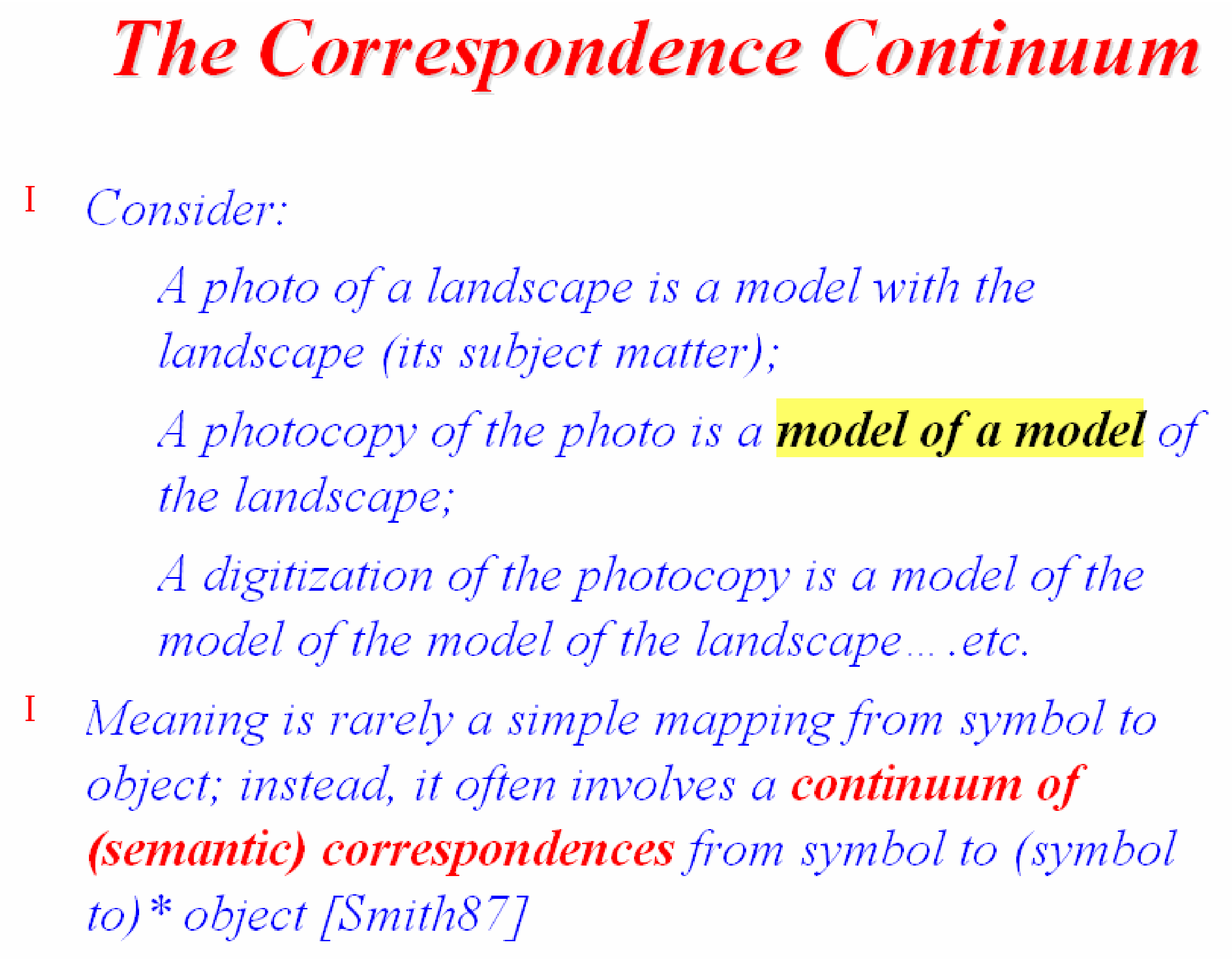
Model

repOf

The system

A model

Model of a model



The Correspondence Continuum

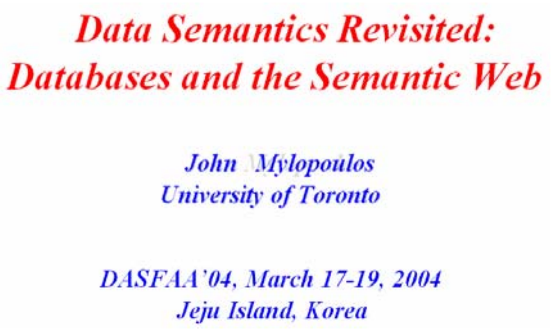
I *Consider:*

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.

I *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]*

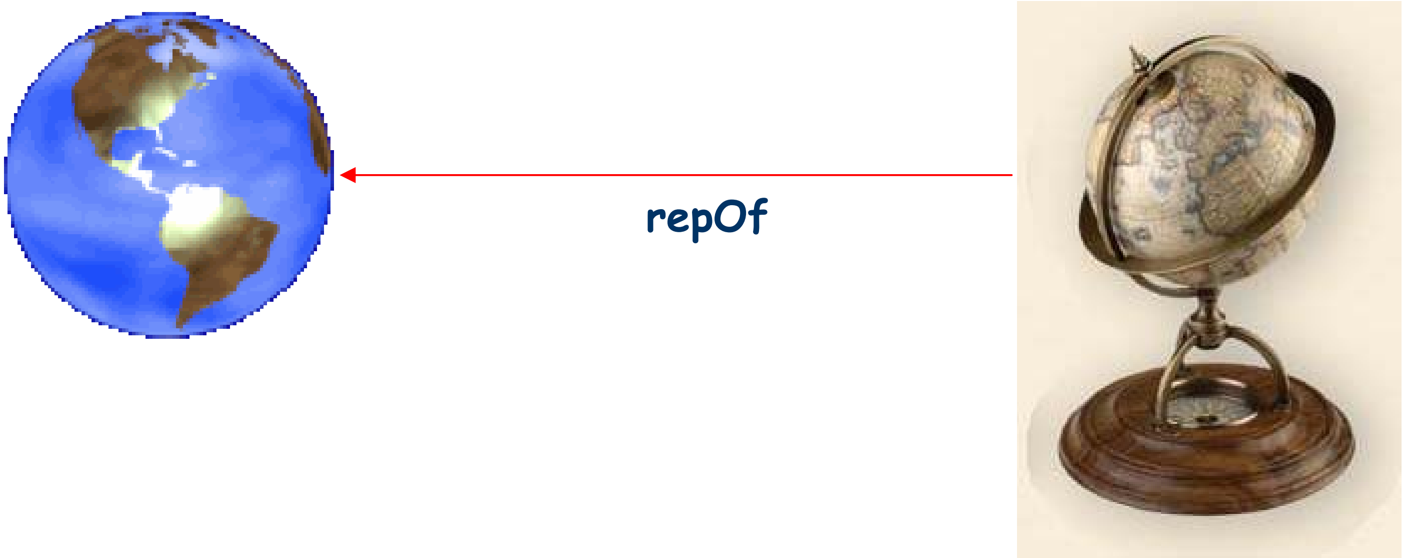


Data Semantics Revisited: Databases and the Semantic Web

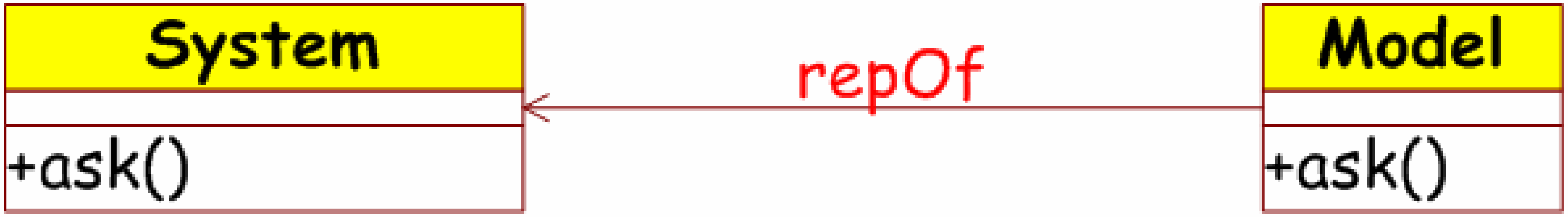
John Mylopoulos
University of Toronto

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

The globe is a model of the earth



repOf



System

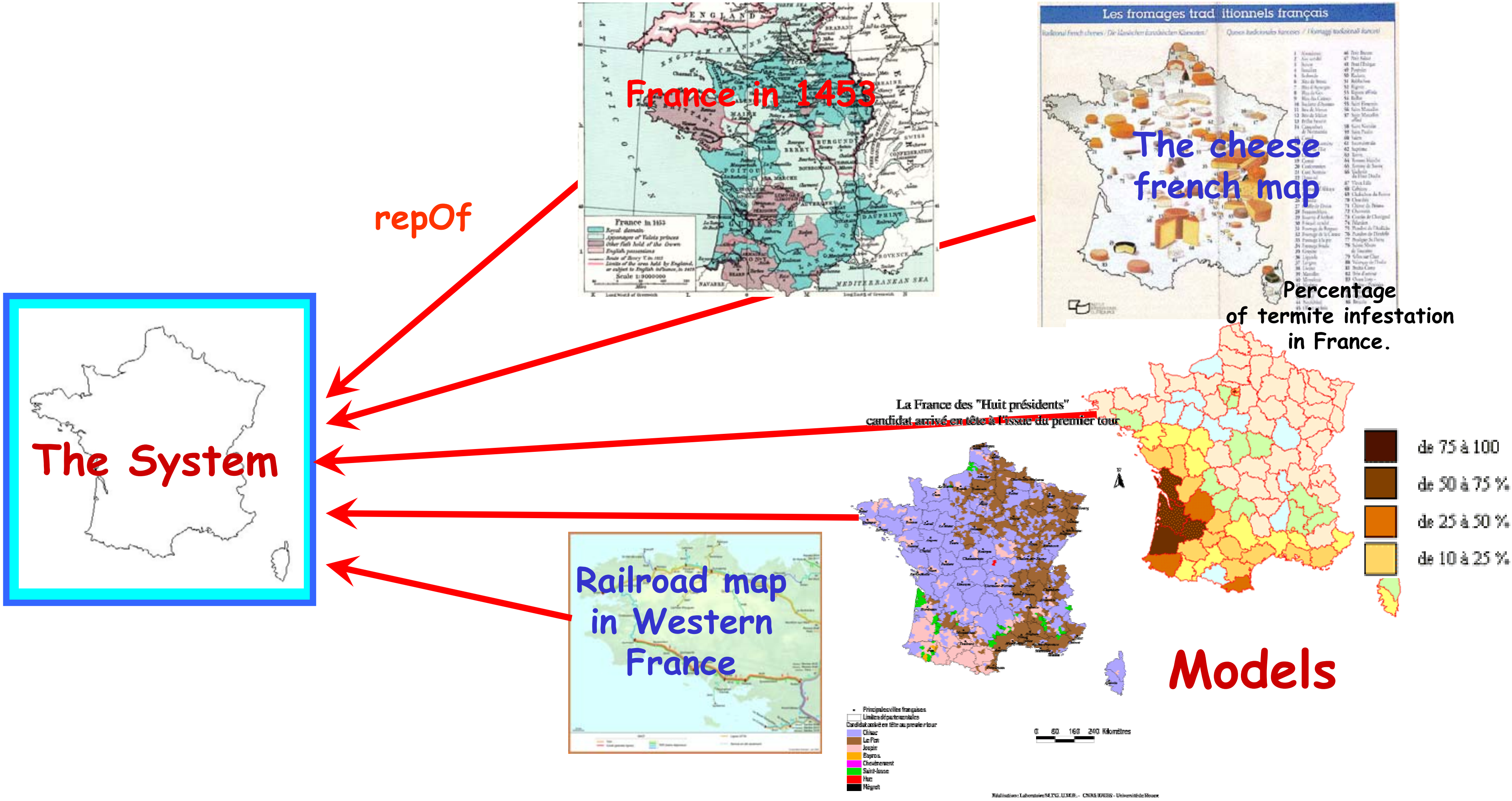
+ask()

repOf

Model

+ask()

A very popular model: geographical maps



repOf

The System

France in 1453

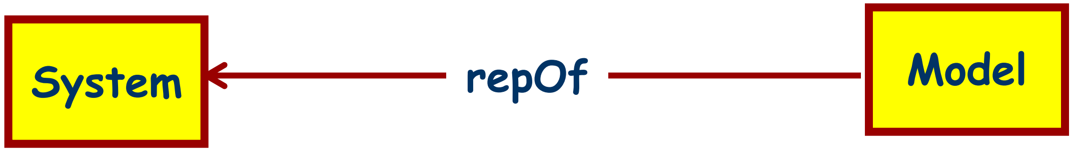
The cheese french map

Percentage of termite infestation in France.

Railroad map in Western France

Models

Les fromages traditionnels français
France in 1453
MEDITERRANEAN SEA
La France des "Huit présidents"
candidat arrivé en tête à l'issue du premier tour
Principales villes françaises
Limites départementales
Candidat arrivé en tête au premier tour
Chirac
Le Pen
Jospin
Bayrou
Chevènement
Saint-Josse
Hue
Mégret
0 80 160 240 Kilomètres
de 75 à 100
de 50 à 75 %
de 25 à 50 %
de 10 à 25 %



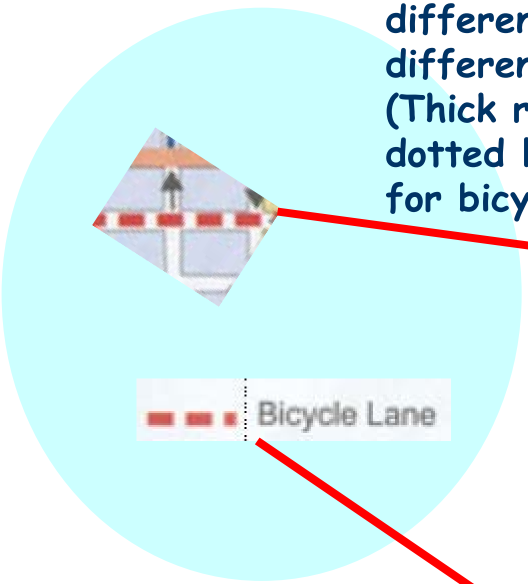
System

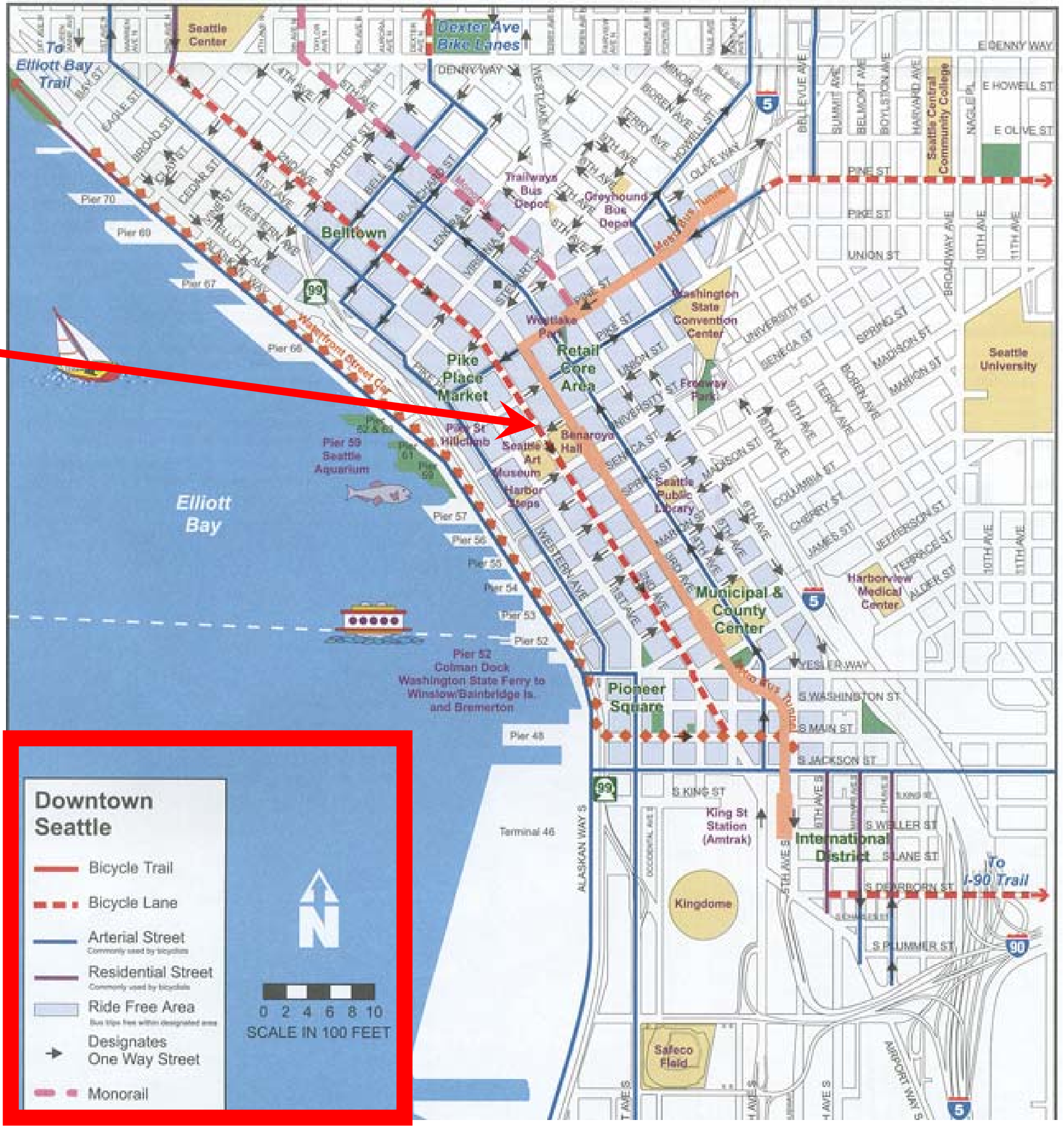
repOf

Model

Every map has a legend (implicit or explicit)

Same visual notation, different context, different meaning (Thick red dotted lines for bicycle lanes)



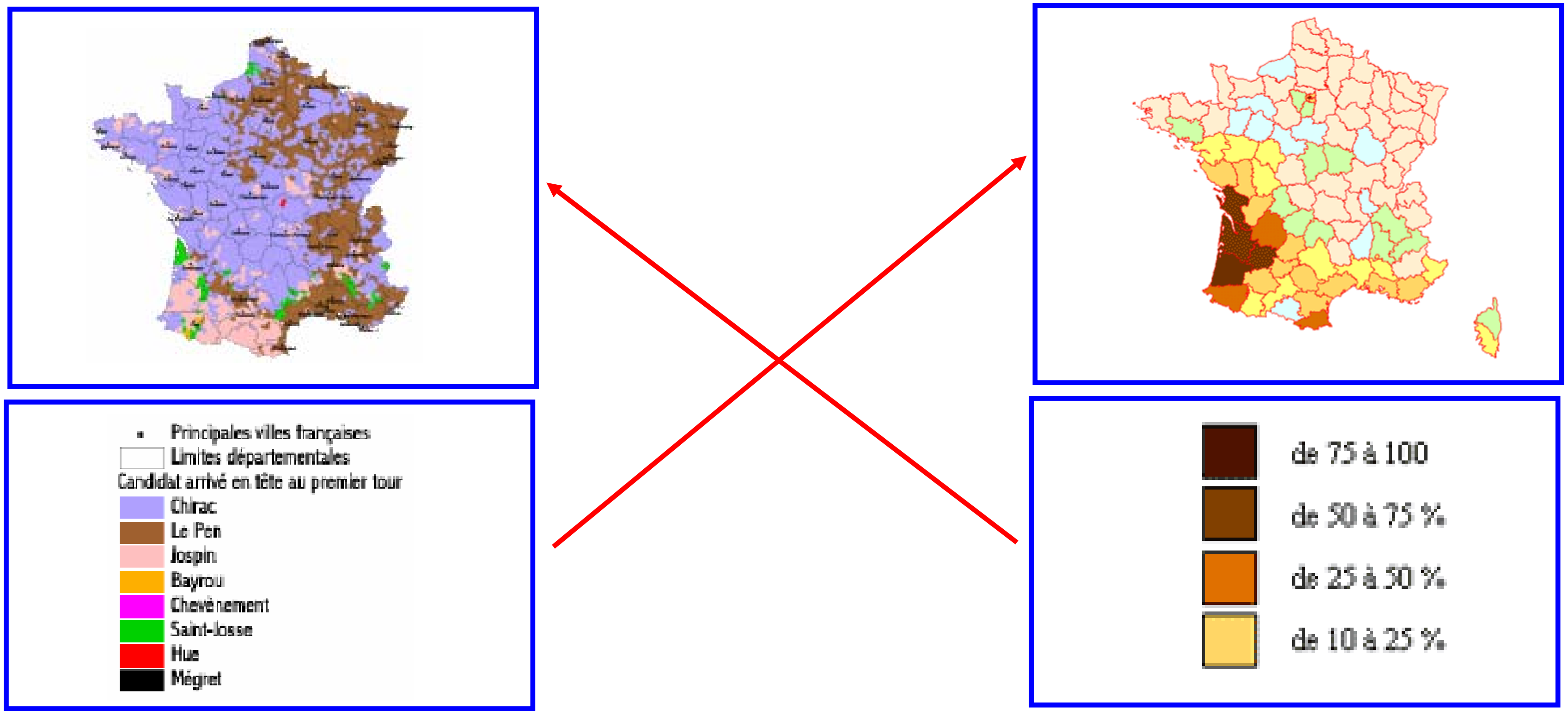


The legend is the metamodel

a Model has no meaning when separated from its metamodel

First round of political election in France in 2002.

Percentage of places infested by termites in France.



Lewis Carroll and the 1:1 map

"That's another thing we've learned from *your* Nation," said Mein Herr, "map-making. But we've carried it much further than you. What do you consider the *largest* map that would be really useful?"

"About six inches to the mile."

"Only *six inches!"* exclaimed Mein Herr. "We very soon got to six *yards* to the mile. Then we tried a *hundred* yards to the mile. And then came the grandest idea of all! We actually made a map of the country, on the scale of *a mile to the mile!"*

"Have you used it much?" I enquired.

"It has never been spread out, yet," said Mein Herr: "the farmers objected: they said it would cover the whole country, and shut out the sunlight! So we now use the country itself, as its own map, and I assure you it does nearly as well."

Lewis Carroll, *Sylvie and Bruno concluded* (London, 1893)

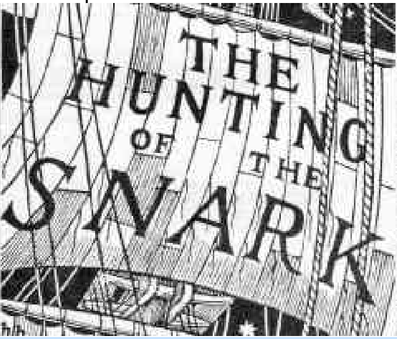
See also J.-L. Borges, J. François, and more recently Umberto Eco.

Lewis Carroll and the blank map

He had bought a large map representing the sea,
Without the least vestige of land:
And the crew were much pleased when they found it to be
A map they could all understand.

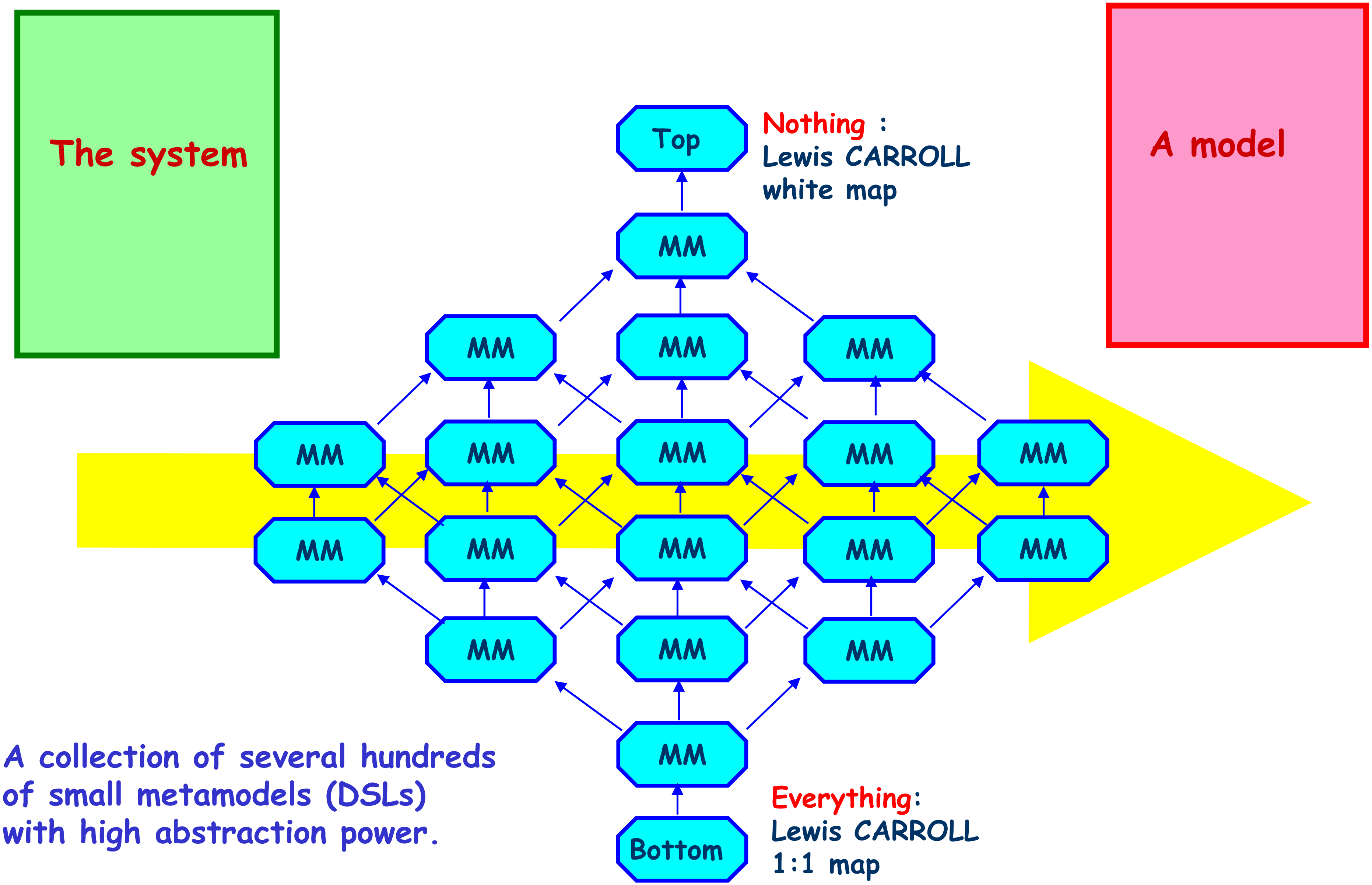
"What's the good of Mercator's North Poles and Equators,
Tropics, Zones, and Meridian Lines?"
So the Bellman would cry: and the crew would reply
"They are merely conventional signs!

"Other maps are such shapes, with their islands and capes!
But we've got our brave Captain to thank:
(So the crew would protest) "that he's bought us the best--
A perfect and absolute blank!"



THE HUNTING OF THE SNARK
an Agony in Eight Fits by Lewis Carroll

A "lattice" of metamodels



Nothing :
Lewis CARROLL
white map

Everything:
Lewis CARROLL
1:1 map

A collection of several hundreds
of small metamodels (DSLs)
with high abstraction power.

Small is beautiful

"Inside every large metamodel is a small metamodel struggling to get out"

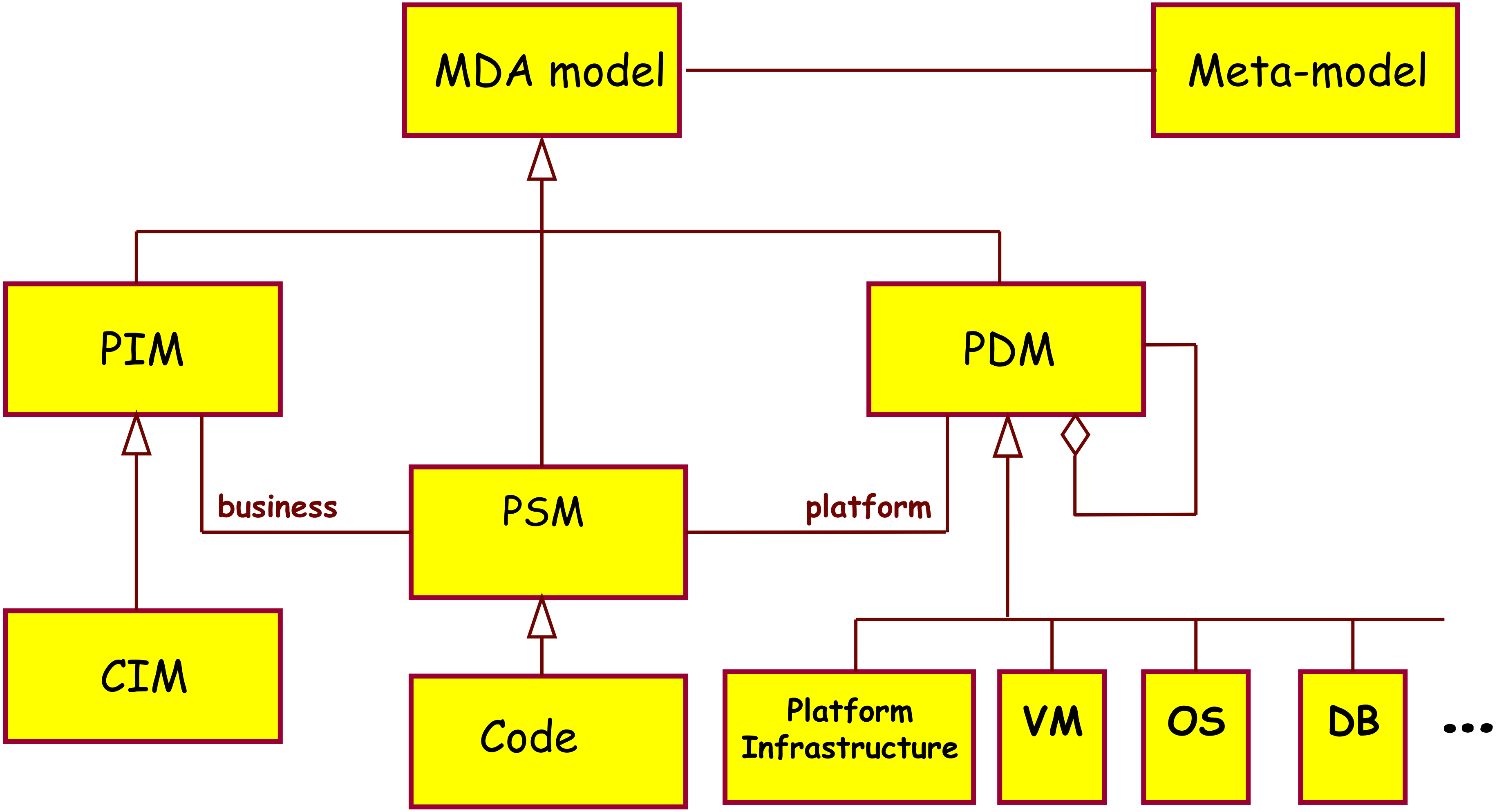
Paraphrasing Tony Hoare

[Compare to UML 2.0 approach]

Research agenda: Everything is a model

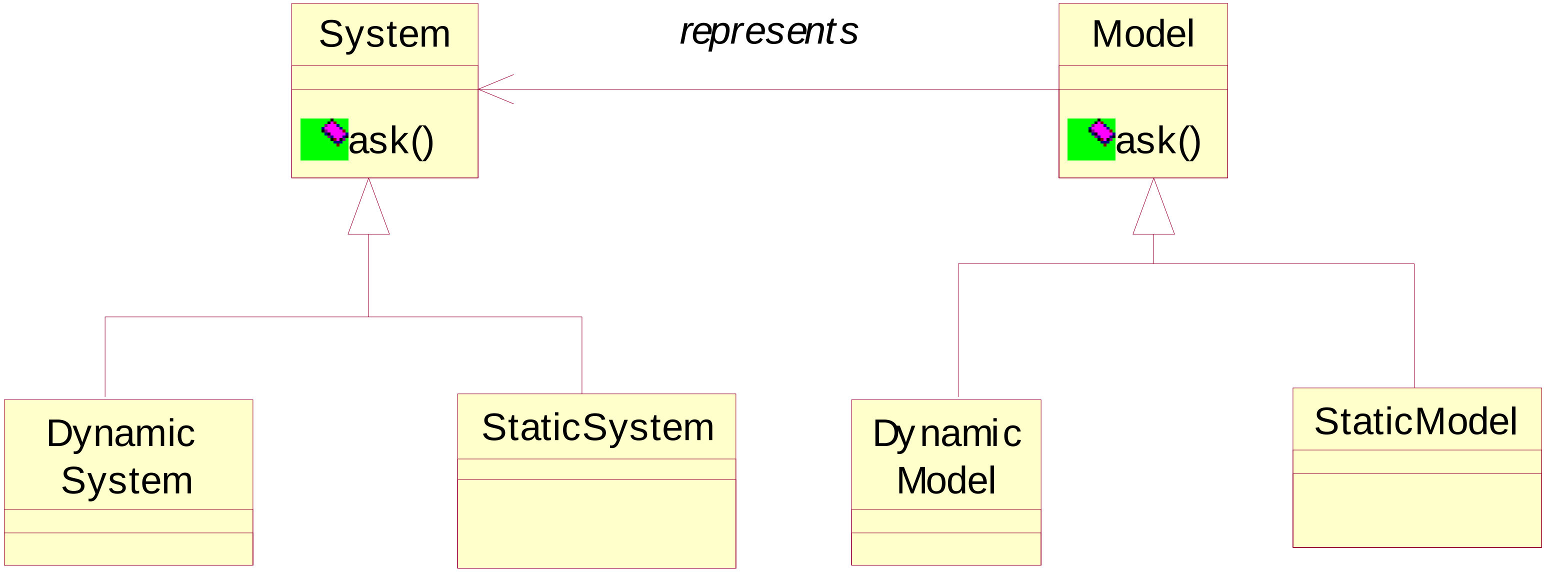
- What is a model?
  - A model is a representation of a system
  - A model is written in the language of its unique metamodel
  - A metamodel is written in the language of its unique metametamodel
    - The unique MMM of the MDA is the MOF
  - A model is a constrained directed labeled graph
  - A model may have a visual representation
- Where do models come from?
- What are the various kinds of models?

A classification of explicit models (incomplete)



Various kinds of models

- Products and processes
- Legacy and components
- Static and dynamic
- Functional and non-functional aspects
- Executable and non executable
- etc.

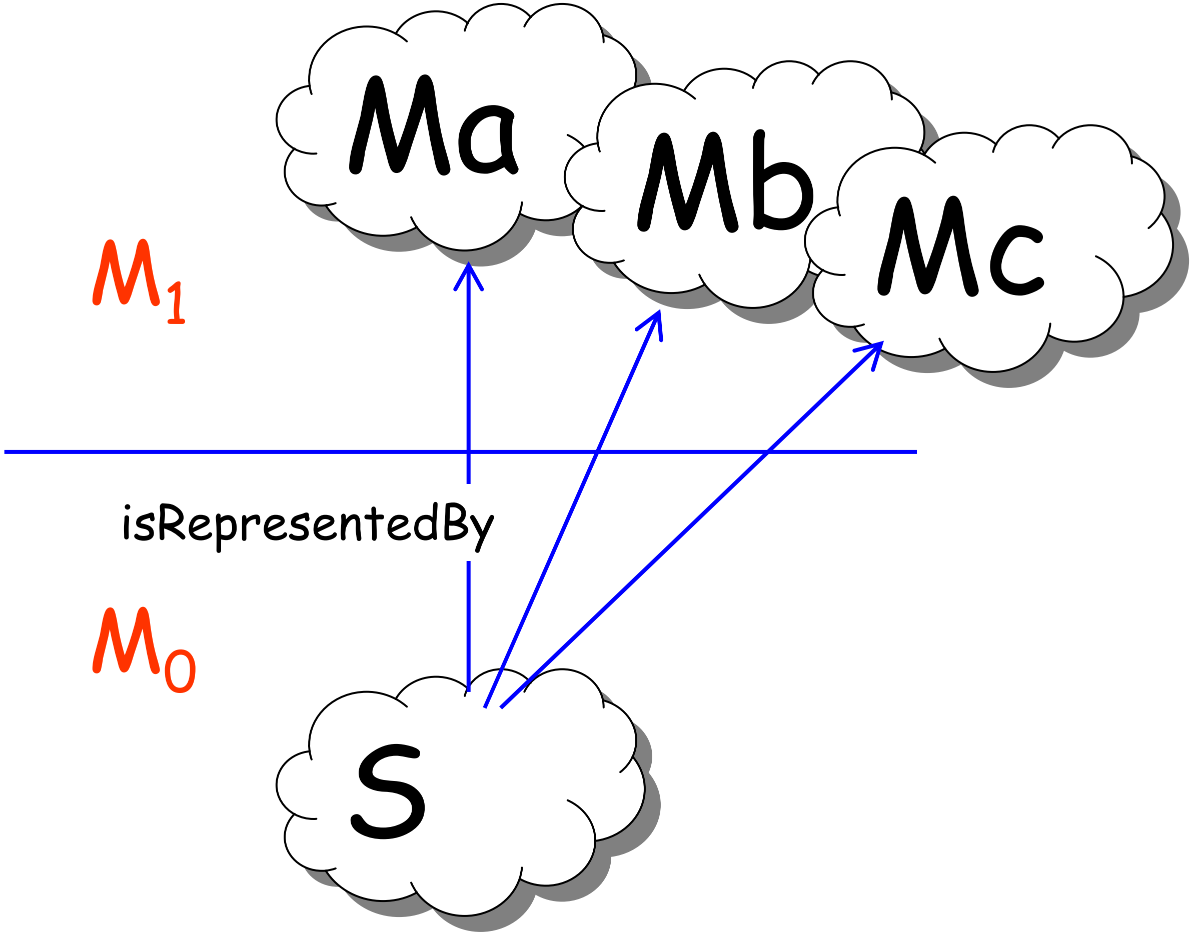


MDA proposed R&D Agenda : "Everything is a model …"

… (or may be converted into a model, or represents a model, or refers to a model, etc.), not only PIMs and PSMs

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

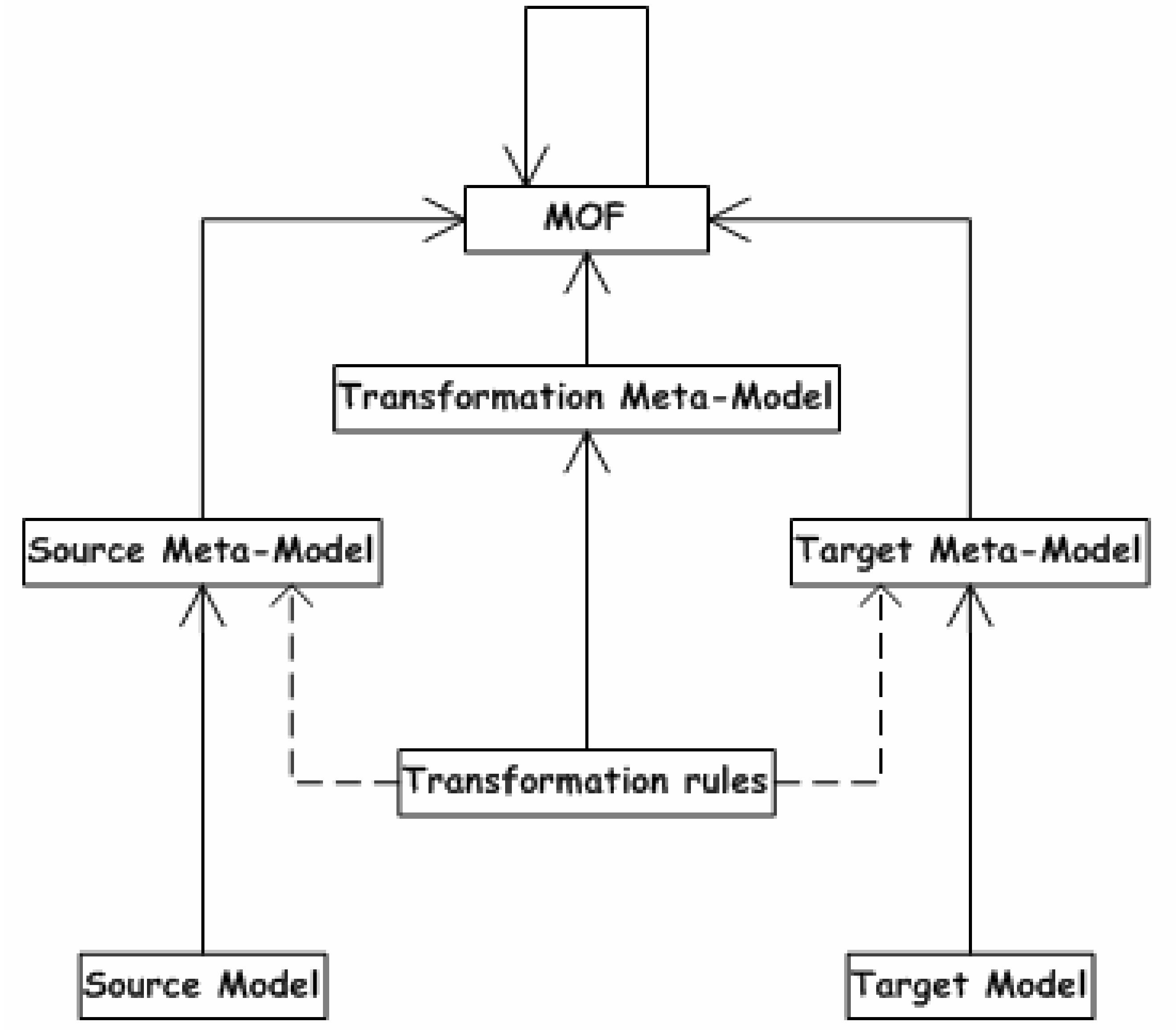
Aspects as models



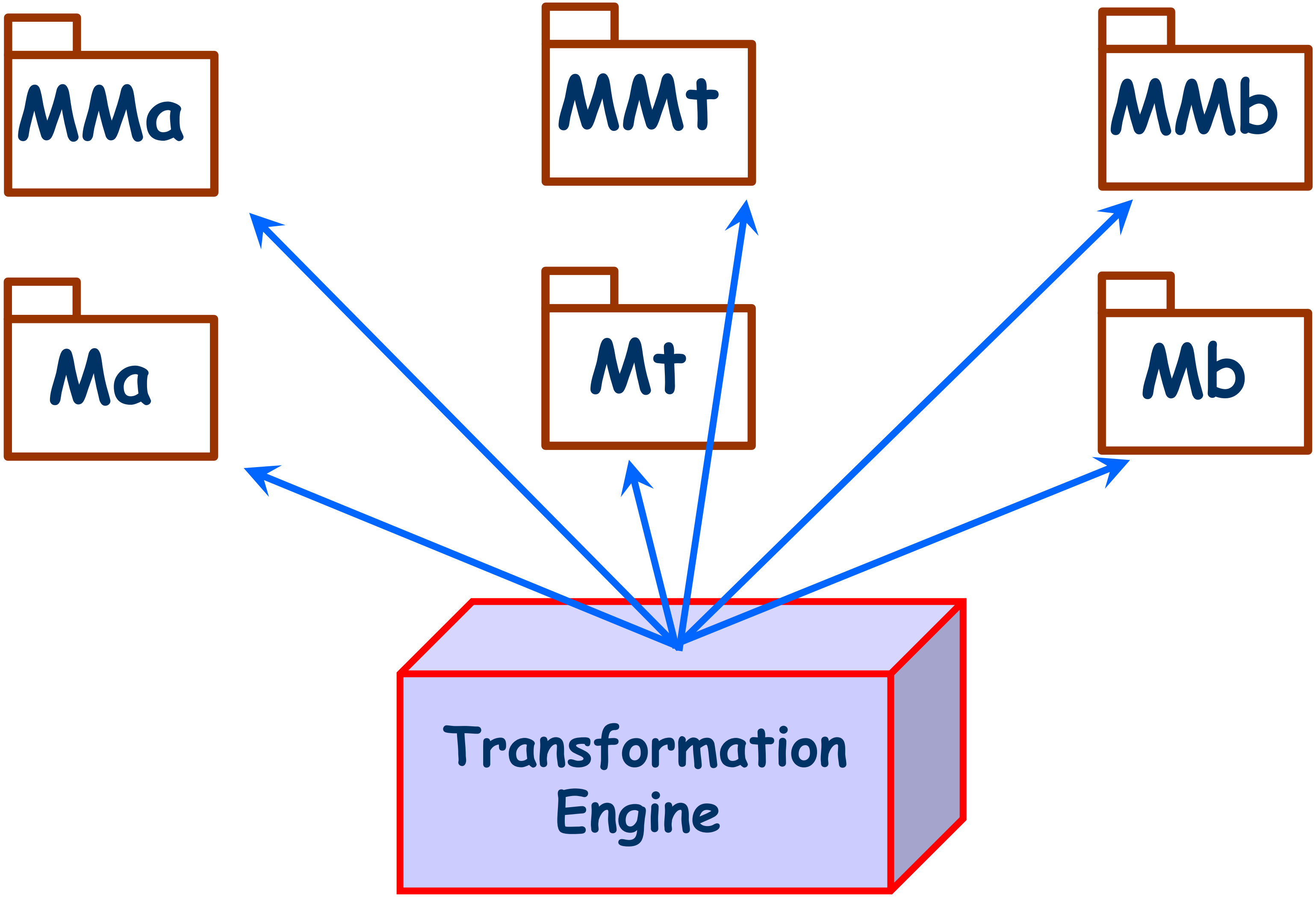
A given system may have plenty of different models.

Each model represents a given aspect of the system.

Transformations as models



Uniform access to models and metamodels



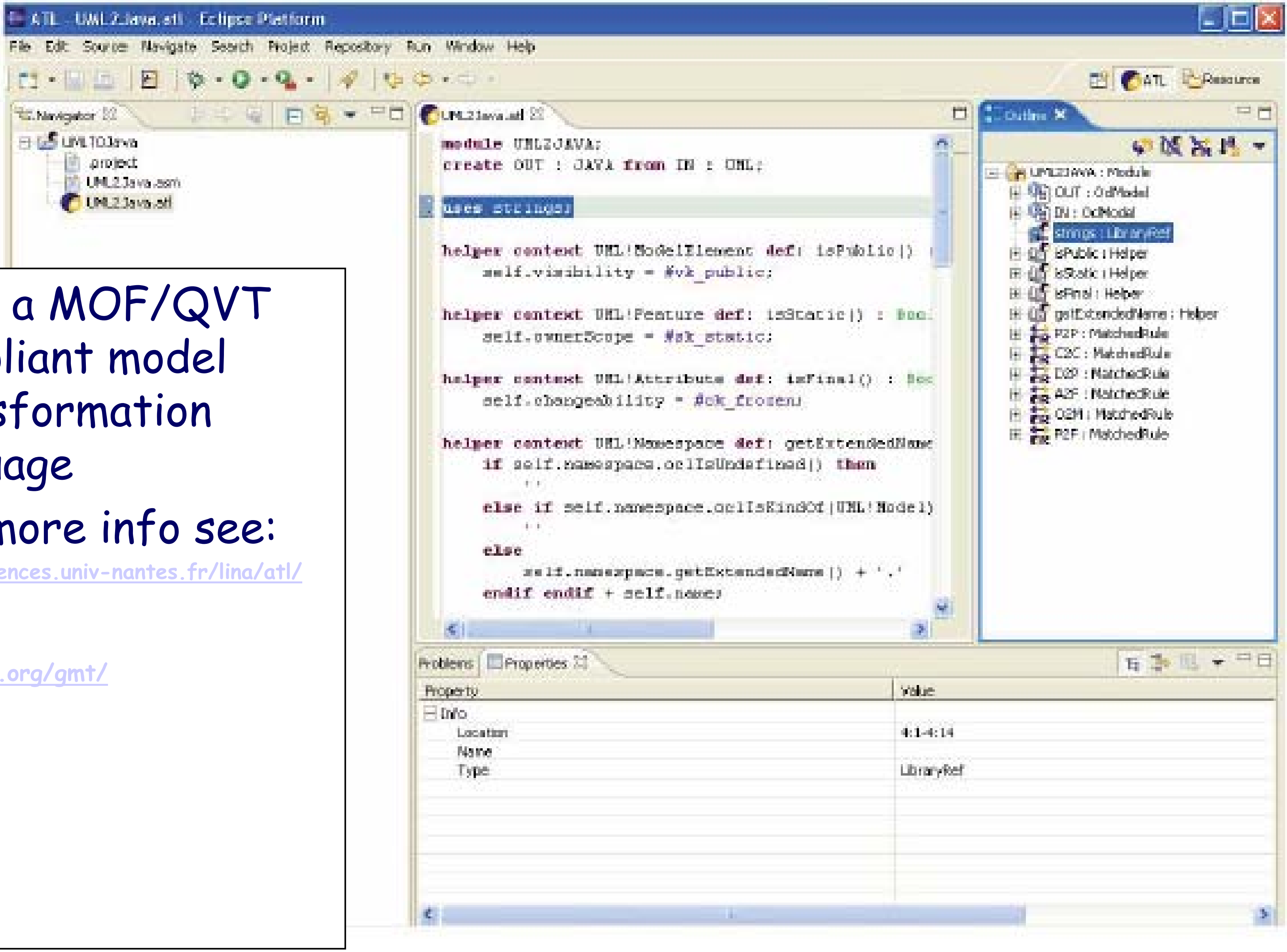
ATL: a model transformation language, engine and IDE

- ATL: a MOF/QVT compliant model transformation language
- For more info see:

http://www.sciences.univ-nantes.fr/lina/atl/

- **or**

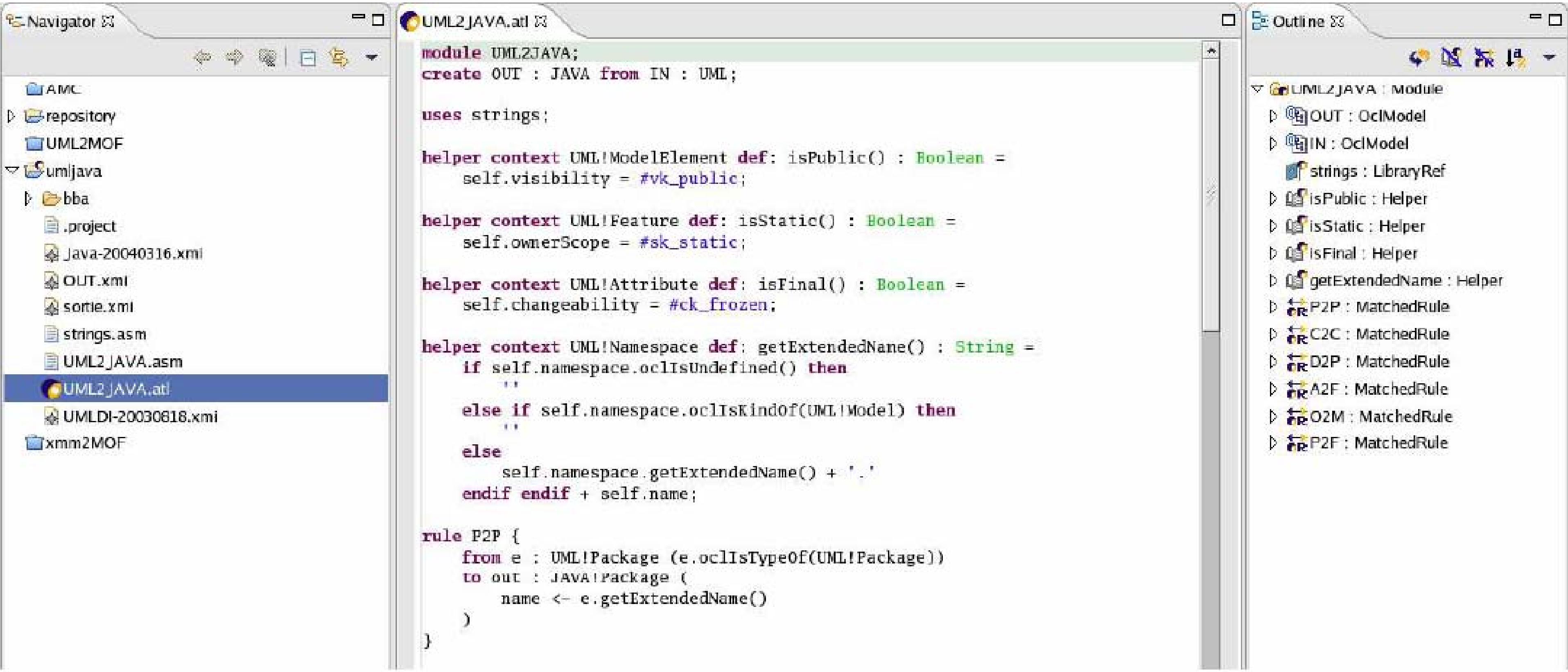
http://eclipse.org/gmt/



```
module UML2JAVA;
create OUT : JAVA from IN : UML;

uses strings;

helper context UML!ModelElement def: isPublic() :
    self.visibility = #vk_public;

helper context UML!Feature def: isStatic() : Boo
    self.ownerScope = #sk_static;

helper context UML!Attribute def: isFinal() : Boo
    self.changeability = #ck_frozen;

helper context UML!Namespace def: getExtendedName
    if self.namespace.oclIsUndefined() then
        ''
    else if self.namespace.oclIsKindOf(UML!Model)
        ''
    else
        self.namespace.getExtendedName() + '.'
    endif endif + self.name;
```

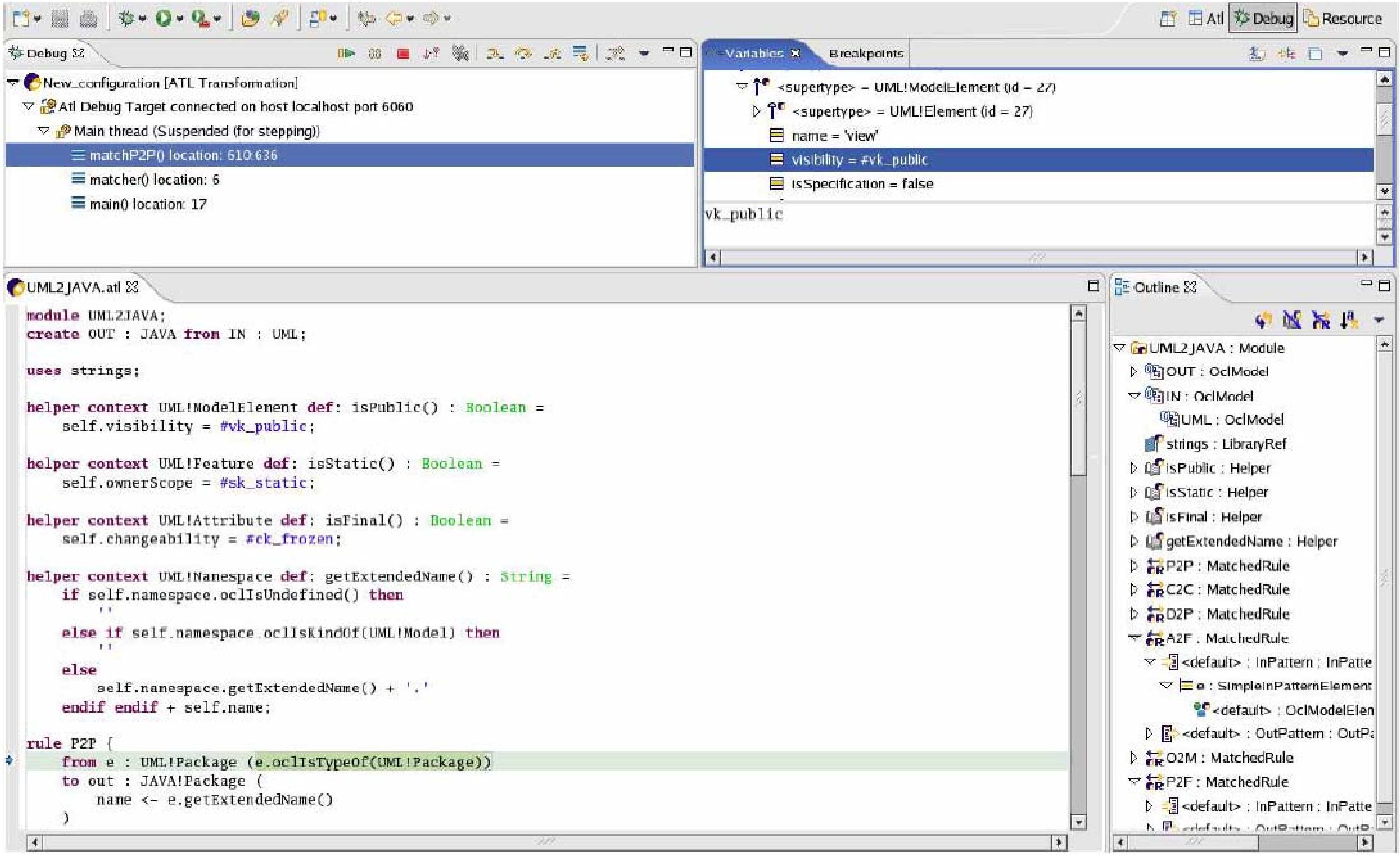
Property	Value
Info	
Location	4:1-4:14
Name	
Type	LibraryRef

ATL editor (part of ATL Integrated Development Environment)



```
module UML2JAVA;
create OUT : JAVA from IN : UML;

uses strings;

helper context UML!ModelElement def: isPublic() : Boolean =
    self.visibility = #vk_public;

helper context UML!Feature def: isStatic() : Boolean =
    self.ownerScope = #sk_static;

helper context UML!Attribute def: isFinal() : Boolean =
    self.changeability = #ck_frozen;

helper context UML!Namespace def: getExtendedName() : String =
    if self.namespace.oclIsUndefined() then
        ''
    else if self.namespace.oclIsKindOf(UML!Model) then
        ''
    else
        self.namespace.getExtendedName() + '.'
    endif endif + self.name;

rule P2P {
    from e : UML!Package (e.oclIsTypeOf(UML!Package))
    to out : JAVA!Package (
        name <- e.getExtendedName()
    )
}
```

ATL Editor with its content outline



New_configuration [ATL Transformation]
Atl Debug Target connected on host localhost port 6060
Main thread (Suspended (for stepping))
matchP2P() location: 610:636
matcher() location: 6
main() location: 17

<supertype> = UML!ModelElement (id = 27)
<supertype> = UML!Element (id = 27)
name = 'view'
visibility = #vk_public
isSpecification = false

vk_public

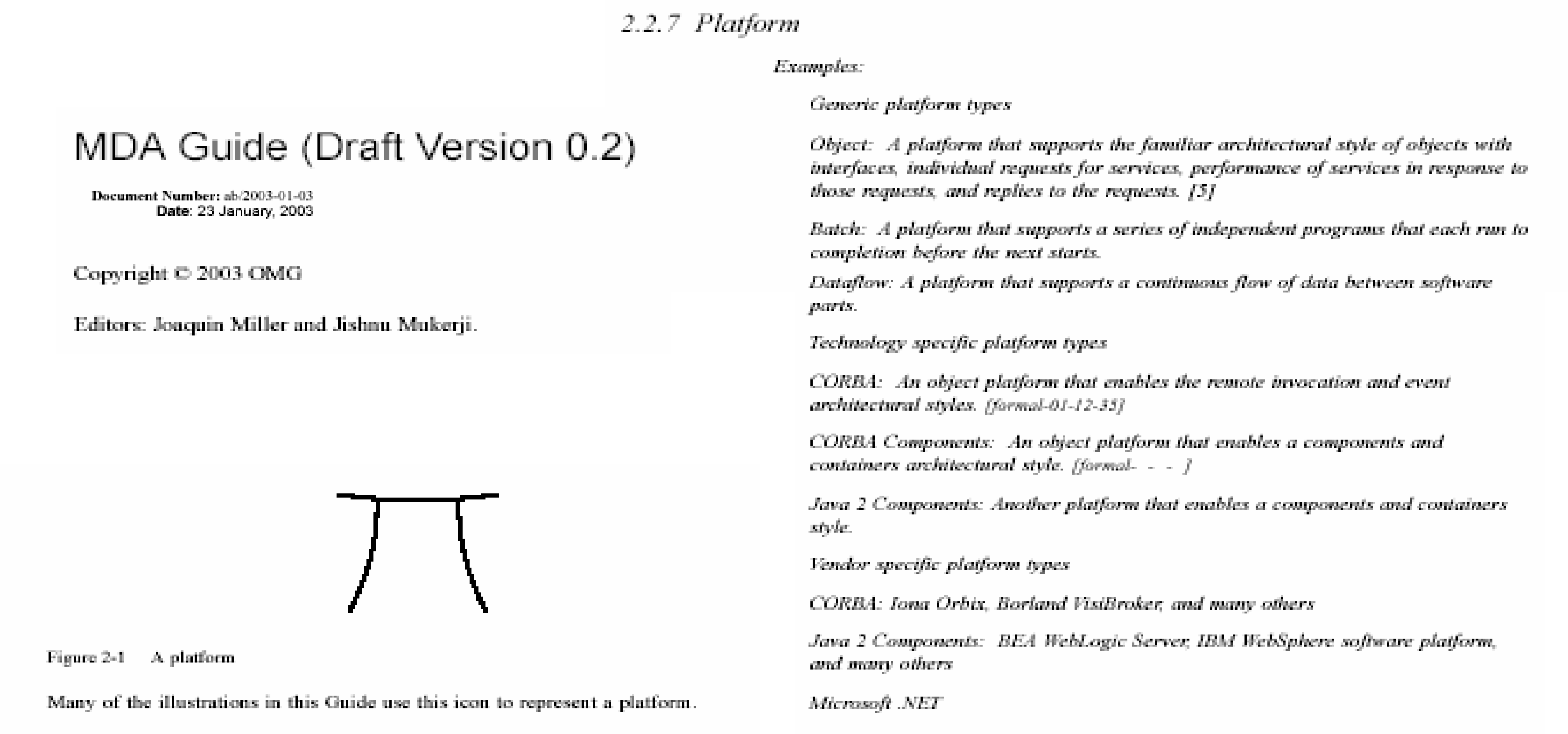
```
module UML2JAVA;
create OUT : JAVA from IN : UML;

uses strings;

helper context UML!ModelElement def: isPublic() : Boolean =
    self.visibility = #vk_public;

helper context UML!Feature def: isStatic() : Boolean =
    self.ownerScope = #sk_static;

helper context UML!Attribute def: isFinal() : Boolean =
    self.changeability = #ck_frozen;

helper context UML!Namespace def: getExtendedName() : String =
    if self.namespace.oclIsUndefined() then
        ''
    else if self.namespace.oclIsKindOf(UML!Model) then
        ''
    else
        self.namespace.getExtendedName() + '.'
    endif endif + self.name;

rule P2P {
    from e : UML!Package (e.oclIsTypeOf(UML!Package))
    to out : JAVA!Package (
        name <- e.getExtendedName()
    )
```

UML2JAVA : Module
OUT : OclModel
IN : OclModel
UML : OclModel
strings : LibraryRef
isPublic : Helper
isStatic : Helper
isFinal : Helper
getExtendedName : Helper
P2P : MatchedRule
C2C : MatchedRule
D2P : MatchedRule
A2F : MatchedRule
<default> : InPattern : InPatte
e : SimpleInPatternElement
<default> : OclModelElen
<default> : OutPattern : OutPa
O2M : MatchedRule
P2F : MatchedRule
<default> : InPattern : InPatte

ATL Debug perspective with the ATL Editor, Debug, Variable and Outline view

A platform is a model

- It is not possible to define platform specific models without having defined precisely what a platform is.



MDA Guide (Draft Version 0.2)

Document Number: ab/2003-01-03
Date: 23 January, 2003

Copyright © 2003 OMG

Editors: Joaquin Miller and Jishnu Mukerji.

Figure 2-1   A platform

Many of the illustrations in this Guide use this icon to represent a platform.

2.2.7 Platform

Examples:

Generic platform types

Object: A platform that supports the familiar architectural style of objects with interfaces, individual requests for services, performance of services in response to those requests, and replies to the requests. [5]

Batch: A platform that supports a series of independent programs that each run to completion before the next starts.

Dataflow: A platform that supports a continuous flow of data between software parts.

Technology specific platform types

CORBA: An object platform that enables the remote invocation and event architectural styles. [formal-01-12-35]

CORBA Components: An object platform that enables a components and containers architectural style. [formal- - -]

Java 2 Components: Another platform that enables a components and containers style.

Vendor specific platform types

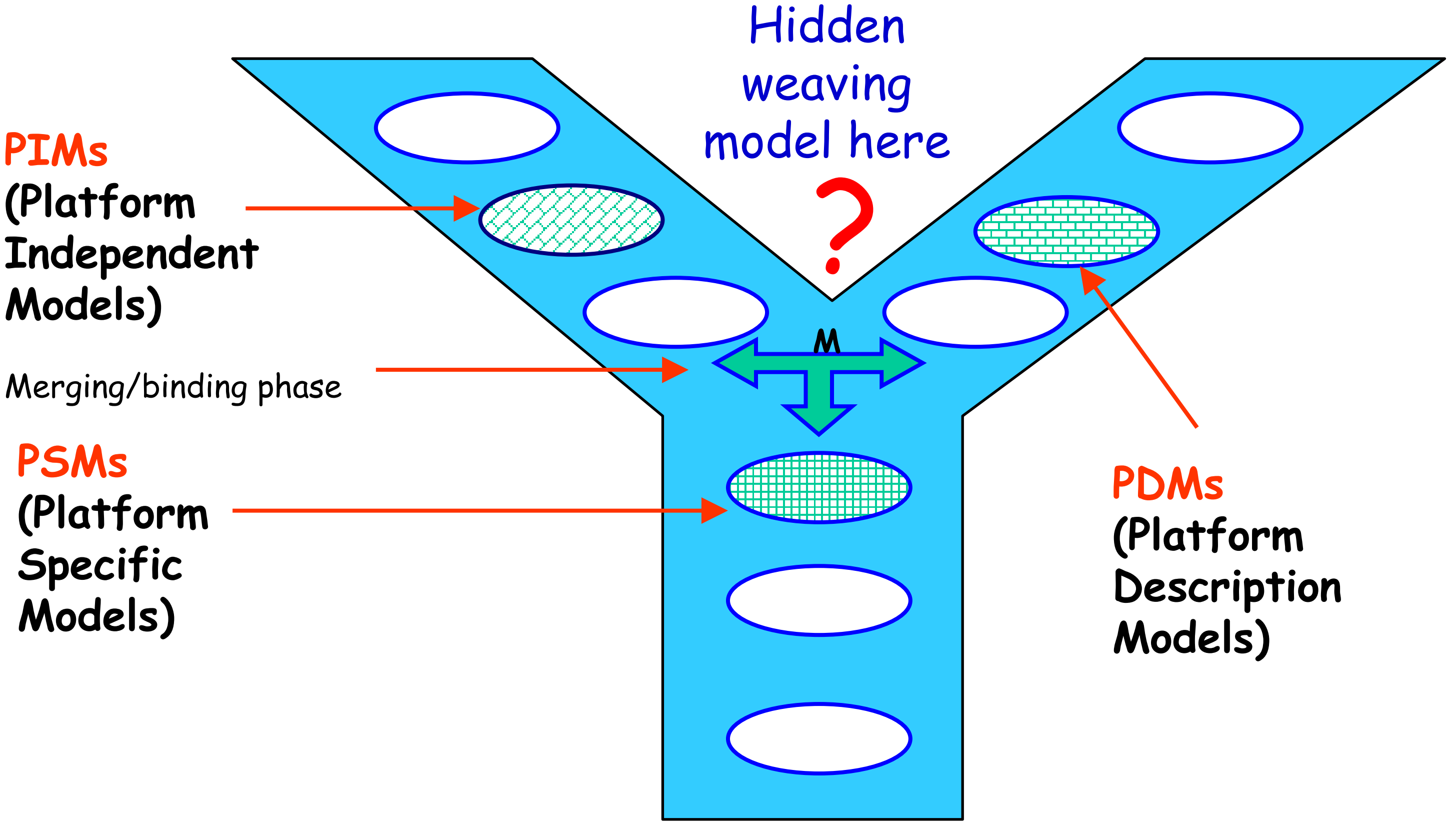
CORBA: Iona Orbix, Borland VisiBroker, and many others

Java 2 Components: BEA WebLogic Server, IBM WebSphere software platform, and many others

Microsoft .NET

A correspondence is a model

- It is not possible to weave two models without having exactly defined the weaving model.


Hidden weaving model here
?
PIMs
(Platform Independent Models)
Merging/binding phase
M
PSMs
(Platform Specific Models)
PDMs
(Platform Description Models)

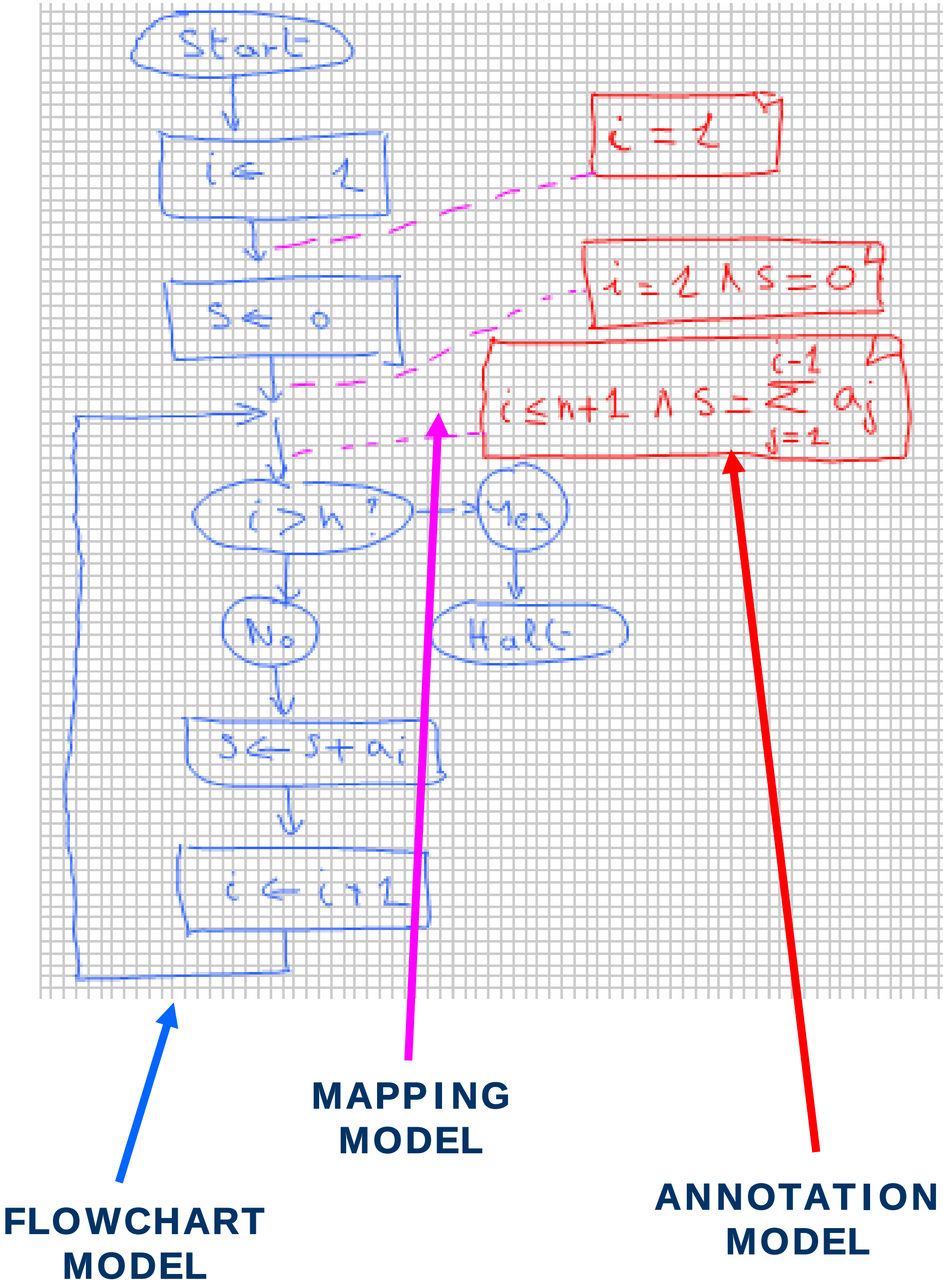
Assigning meanings to models

- 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

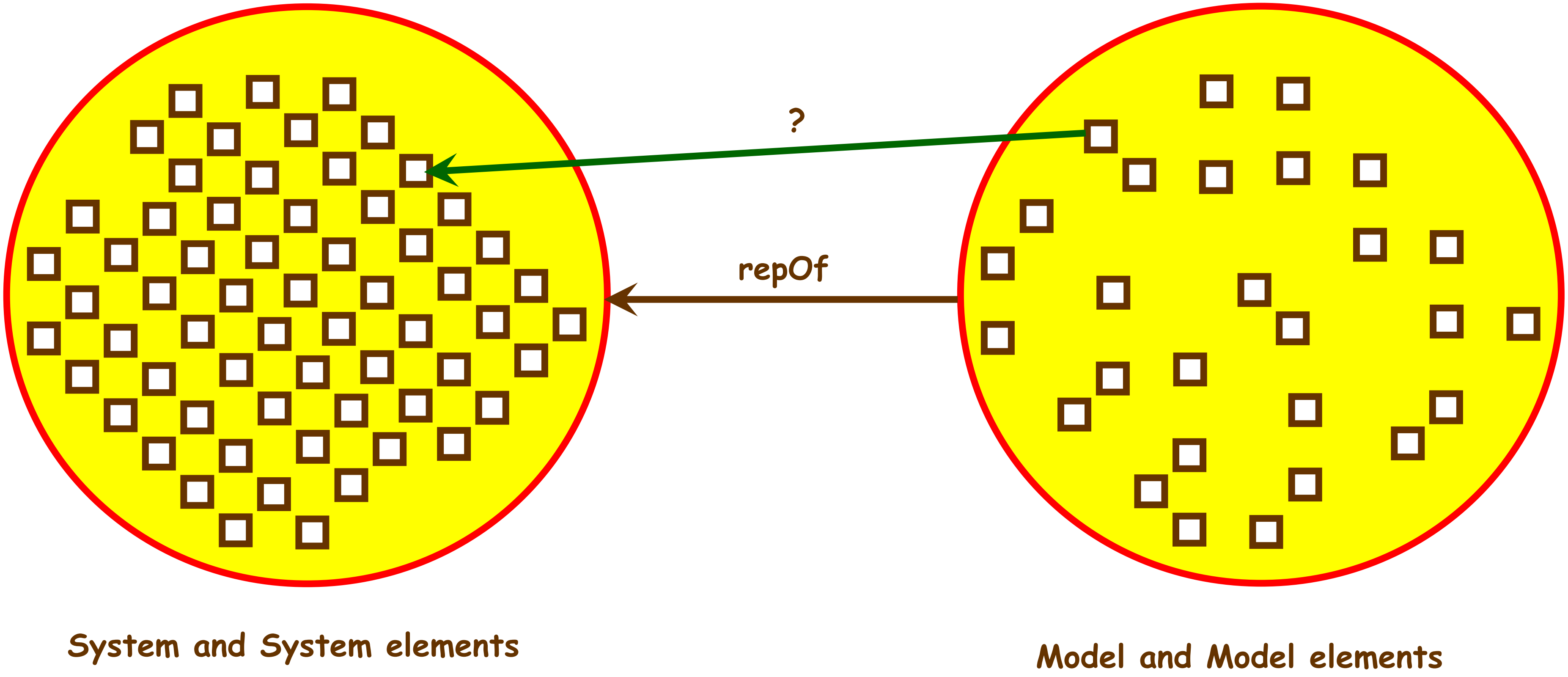


FLOWCHART MODEL

MAPPING MODEL

ANNOTATION MODEL

The "representation" relation



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

On the *repOf* relation

"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."

Cantwell Smith, B. Limits of Correctness in Computers, Report CSLI-85-36,
Center for the Study of Language and Information, Stanford University,
California, October 1985.

Agenda

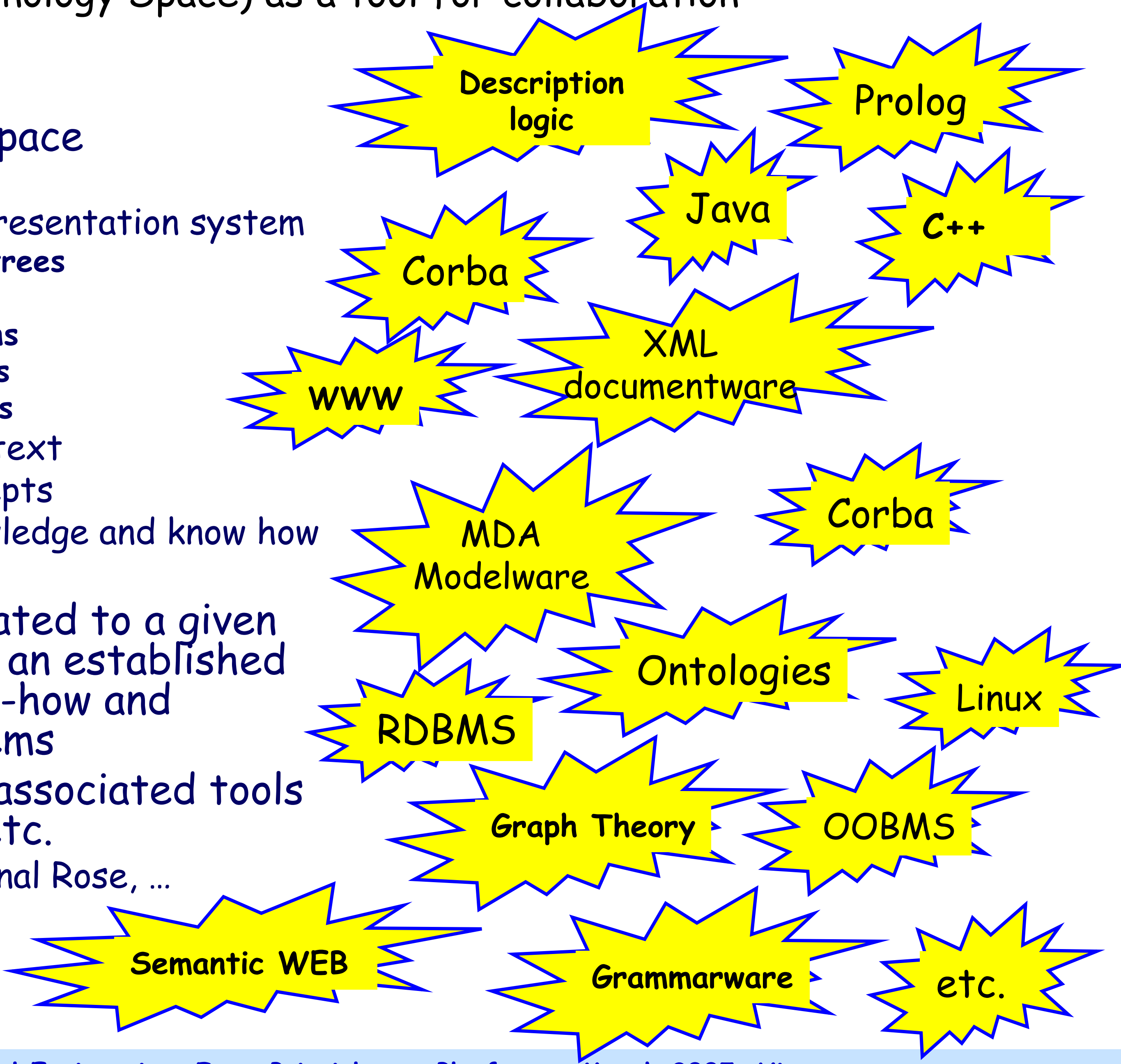
Technical spaces

Technical Spaces and Working Contexts

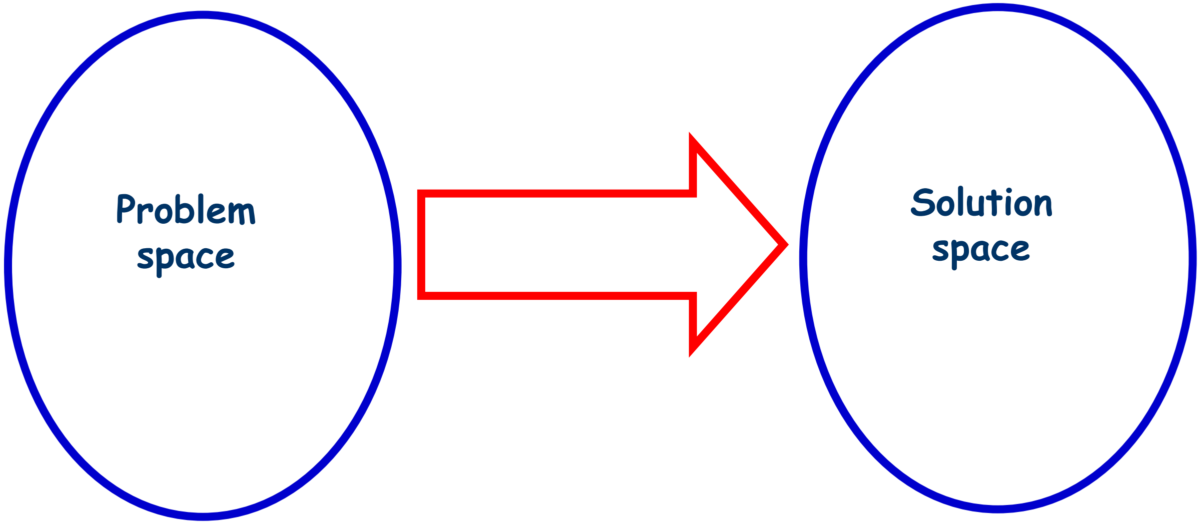
- Technical Spaces
  - Examples: MDE, EBNF, XML, DBMS, ontologies, etc.
  - Conjecture:
    - Each TS is represented by a metametamodel
    - Each TS is organized in a 3 metalevel architecture
- Working contexts
  - Local
    - MM specific, e.g. UML
  - Global
    - TS specific, MM independent, e.g. MOF
  - Universal
    - Across several TSs

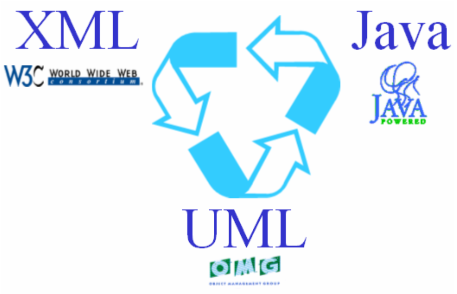
The notion of TS (Technology Space) as a tool for collaboration

- A Technology Space corresponds to:
  - A uniform representation system
    - **Syntactic trees**
    - **XML trees**
    - **Sowa graphs**
    - **UML graphs**
    - **MOF graphs**
  - A working context
  - A set of concepts
  - A shared knowledge and know how
  - etc.
- It is usually related to a given community with an established expertise, know-how and research problems
- It has a set of associated tools and practices, etc.
  - Protégé, Rational Rose, …



The solution space is multiple and complex





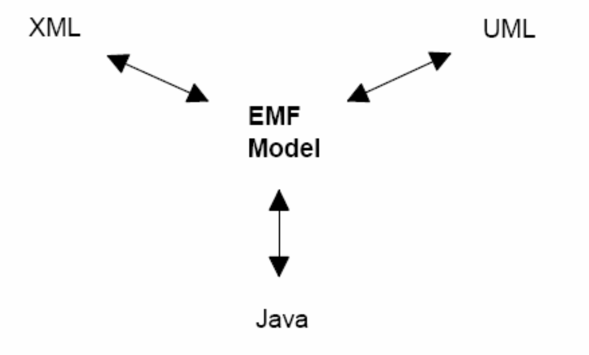
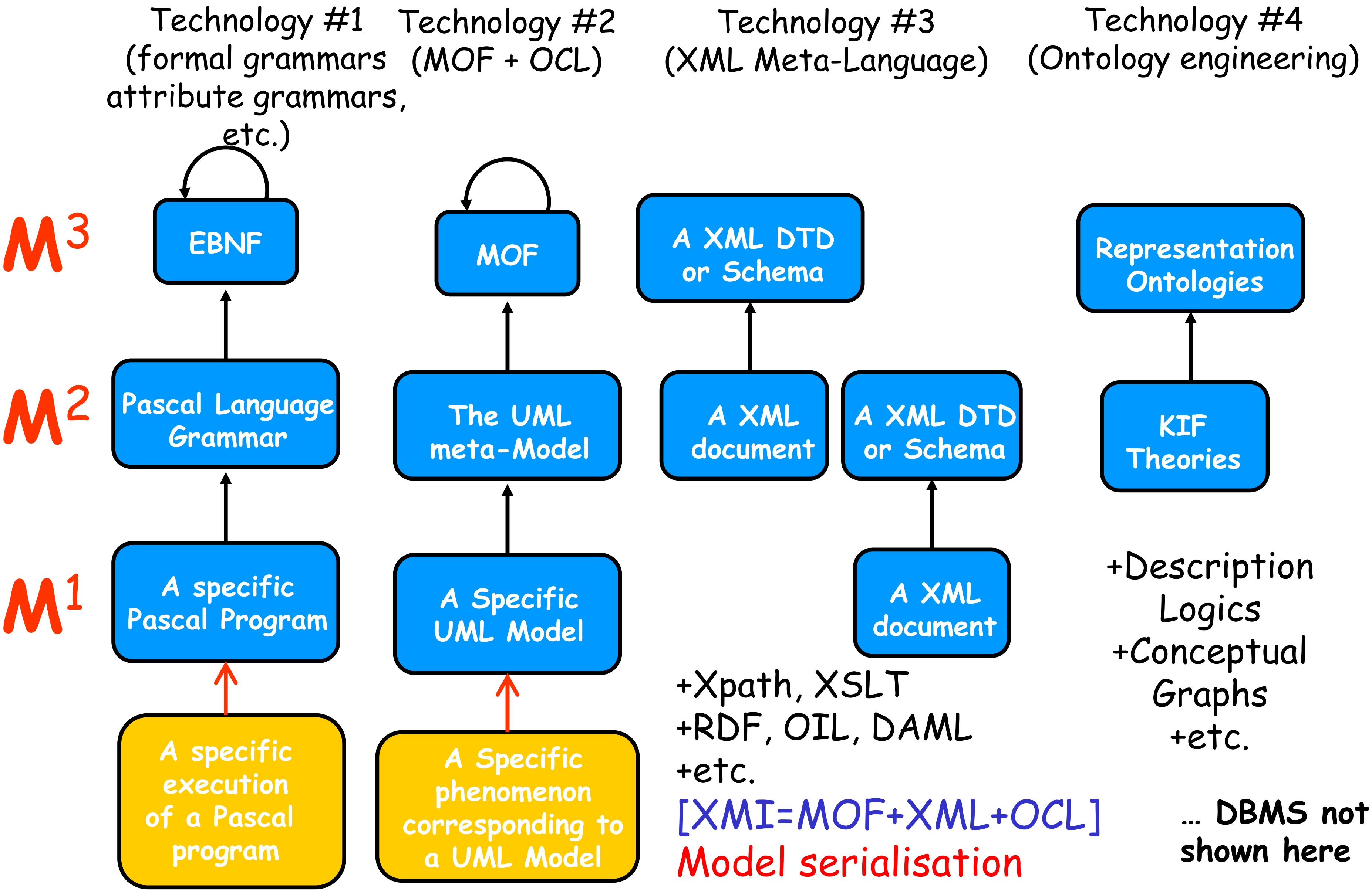


Figure 2.2 EMF unifies Java, XML, and UML.

How to map the problem space onto the composite solution space?

Abstract Syntax Systems Compared



Technology #1 (formal grammars attribute grammars, etc.)

Technology #2 (MOF + OCL)

Technology #3 (XML Meta-Language)

Technology #4 (Ontology engineering)

M^3

M^2

M^1

EBNF

Pascal Language Grammar

A specific Pascal Program

A specific execution of a Pascal program

MOF

The UML meta-Model

A Specific UML Model

A Specific phenomenon corresponding to a UML Model

A XML DTD or Schema

A XML document

A XML DTD or Schema

A XML document

Representation Ontologies

KIF Theories

+Xpath, XSLT
+RDF, OIL, DAML
+etc.

[XMI=MOF+XML+OCL]

Model serialisation

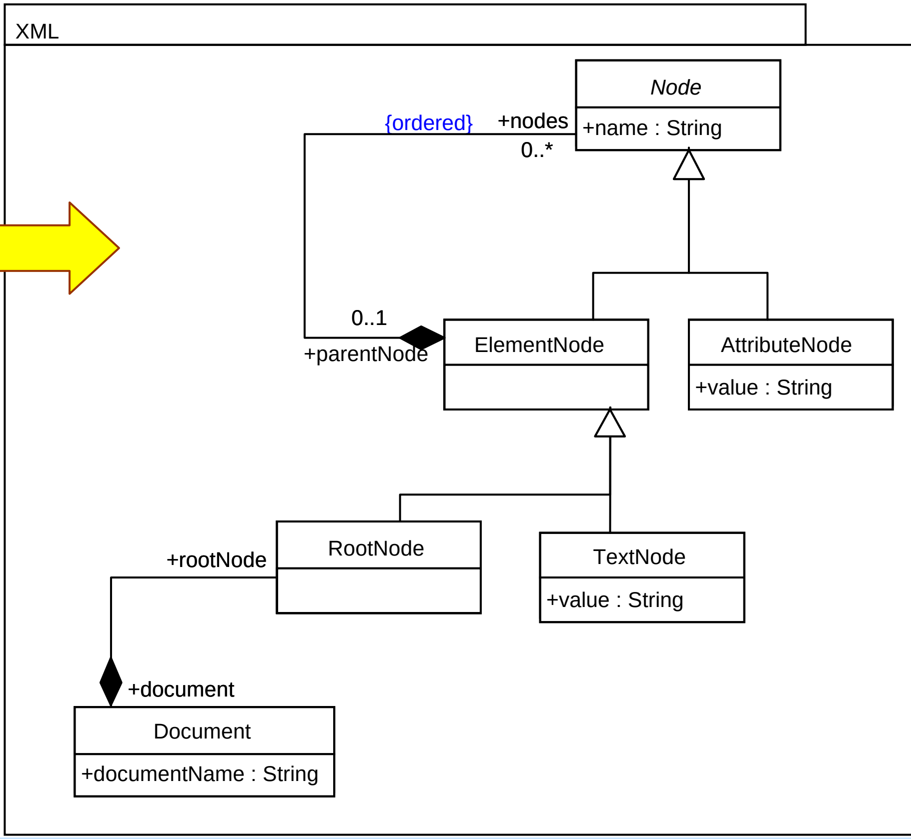
+Description Logics
+Conceptual Graphs
+etc.

… DBMS not shown here

Representation issues

- Each TSpace is rooted on a specific representation ontology (RO)
- This representation ontology is sometimes called a **metametamodel**
- For MDA people this is called the MOF
- Protegé for example is based on a specific RO
- XML is based on a specific RO describing a special kind of trees (well formed XML documents)
- Abstract syntax trees have a different RO
- MOF is a RO for a special kind of graphs
- sNets or Vanilla are other alternatives
- Some ROs use hypergraphs

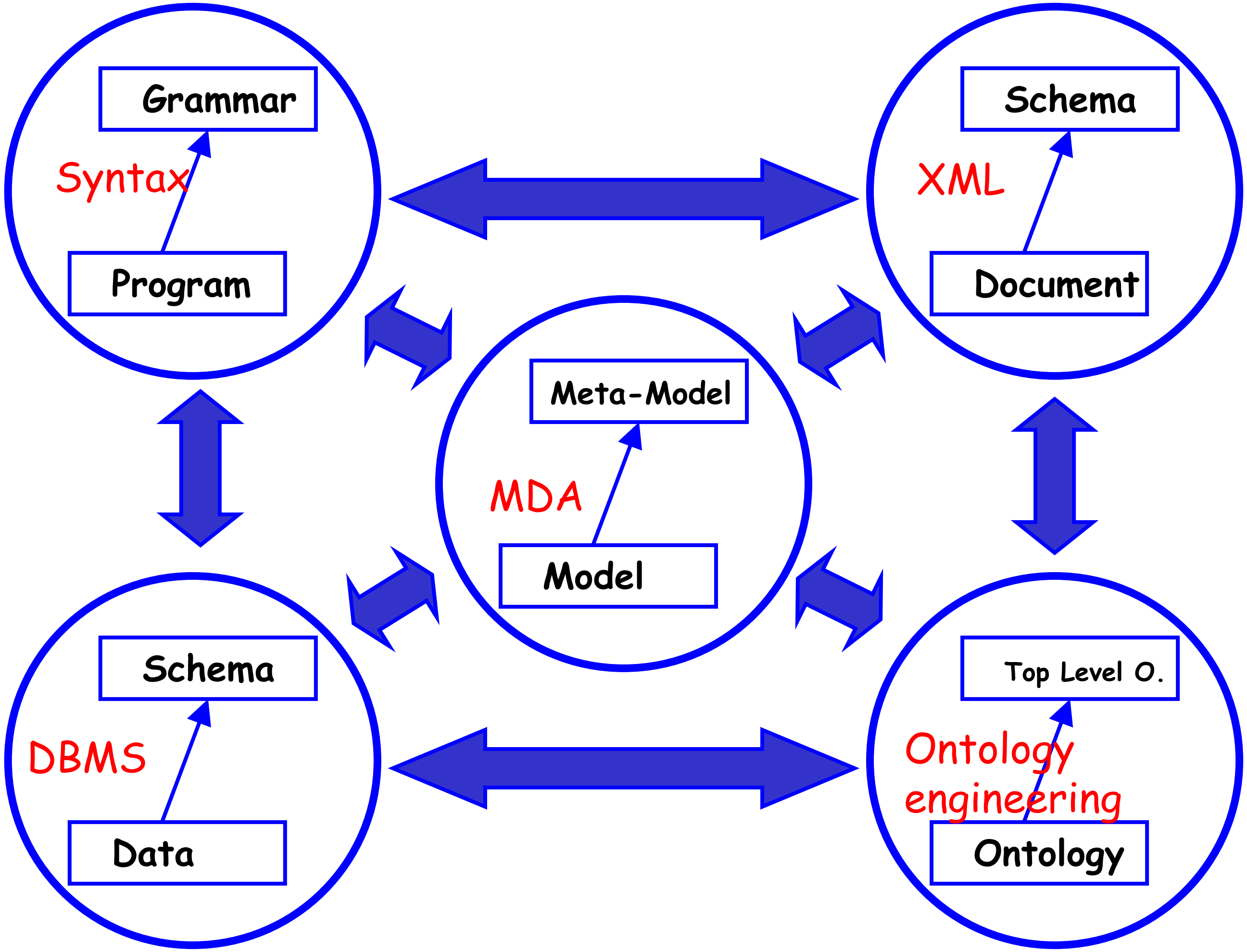
Fragment of the XML representation ontology



A technology space is organized around a set of concepts

Spaces may be connected via bridges

Spaces are often similarly organized



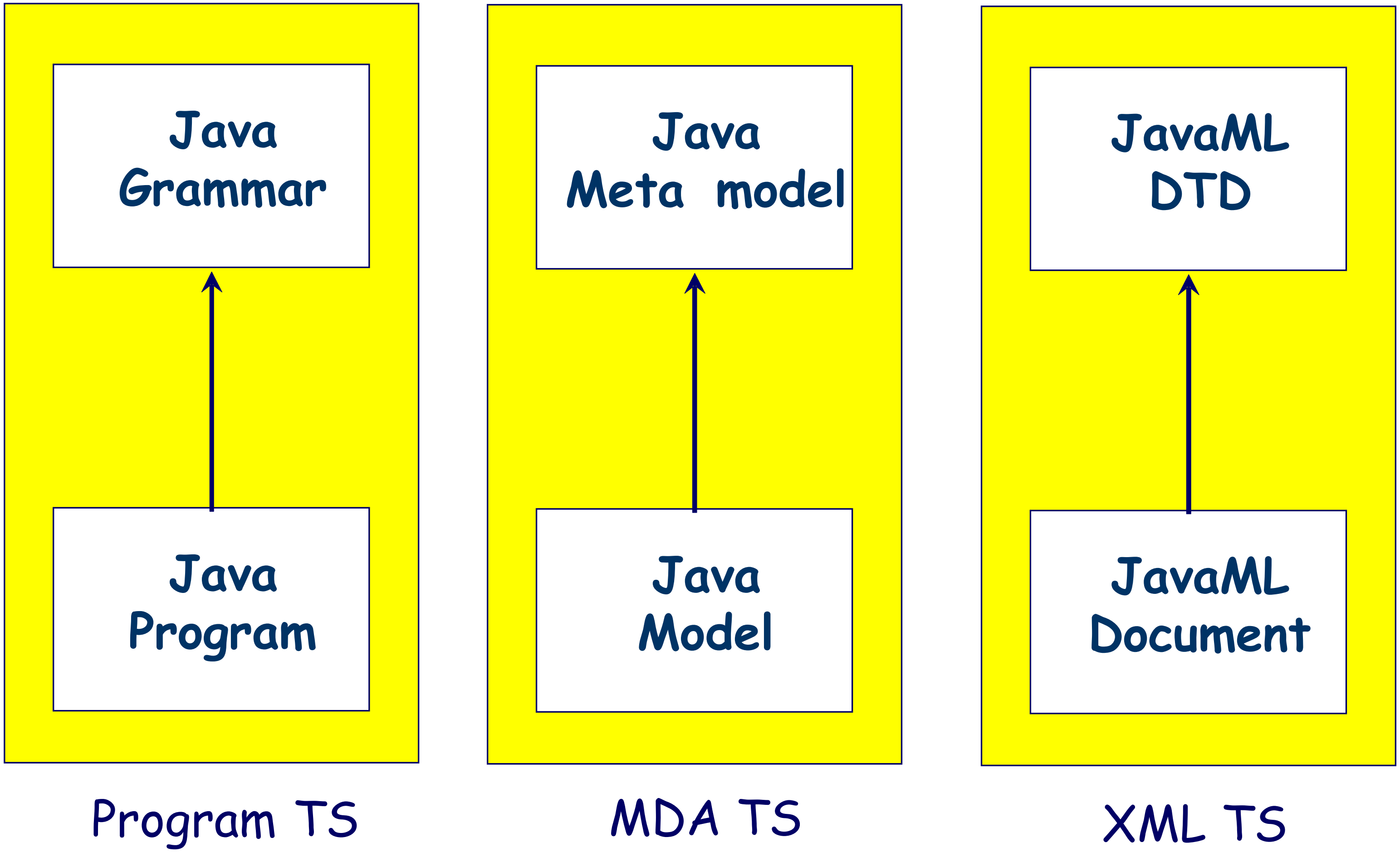
Comparing spaces

+ stability in time

	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	Good	Good	Poor	Poor
Traceability	Good	Fair	Poor	Excellent
Transformability	Excellent	Fair	Fair	Fair

(NB: marks are indicative)

Three representations for the same program


Java Grammar

Java Program

Program TS

Java Meta model

Java Model

MDA TS

JavaML DTD

JavaML Document

XML TS

Three representations for the same program

```java
import java.applet.*;
import java.awt.*;

public class FirstApplet extends Applet {
  public void paint(Graphics g) {
    g.drawString("FirstApplet", 25, 50);
  }
}
```

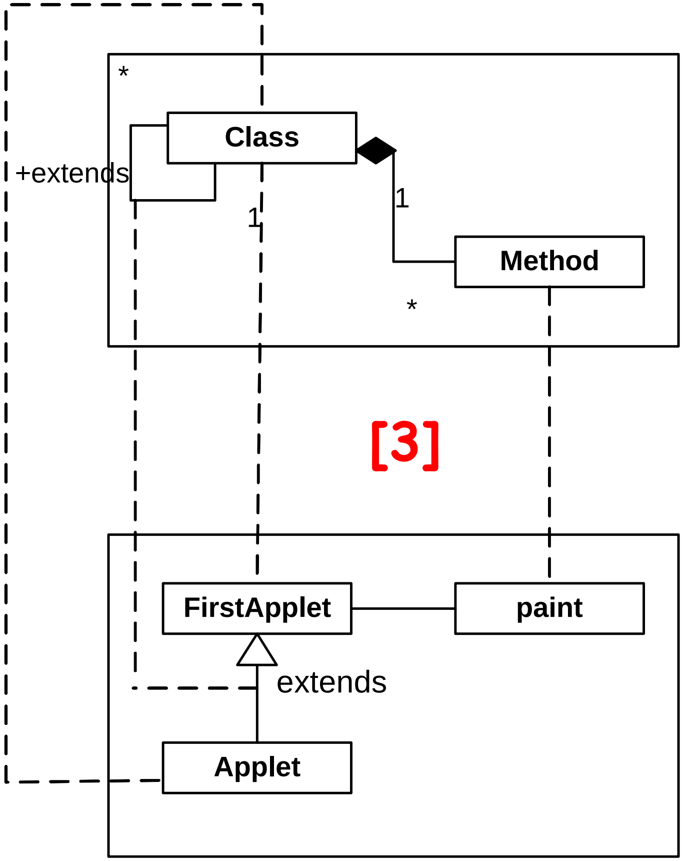
[1]

Java source code

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE java-source-program SYSTEM "java-ml.dtd">
<java-source-program name="FirstApplet.java">
  <import module="java.applet.*"/>
  <import module="java.awt.*"/>
  <class name="FirstApplet" visibility="public">
        <superclass class="Applet"/>
        <method name="paint" visibility="public" id="meth-15">
                  <type name="void" primitive="true"/>
                  <formal-arguments>
                            <formal-argument name="g" id="frmarg-13">
                            <type name="Graphics"/></formal-argument>
                  </formal-arguments> …….
```

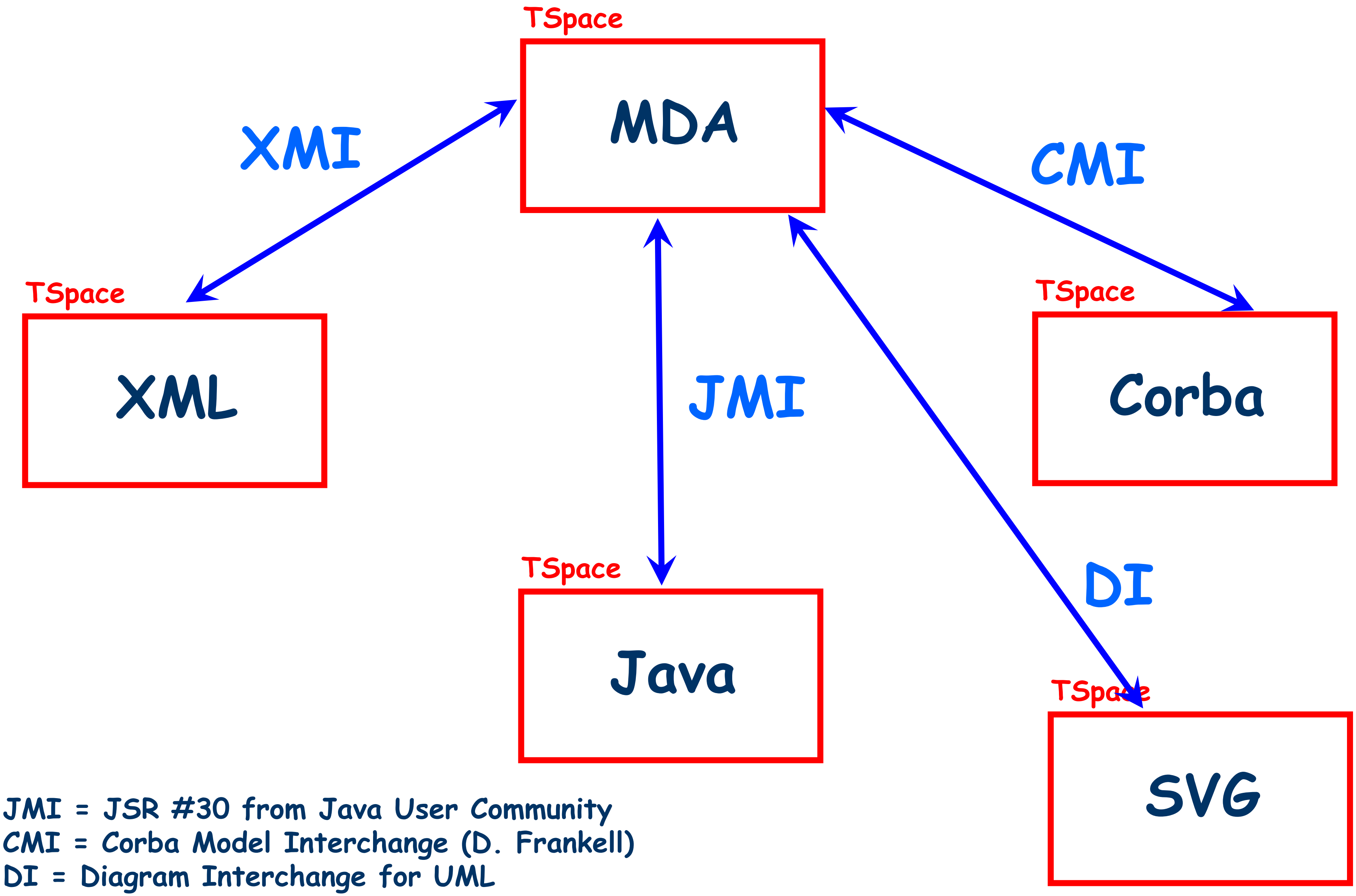
[2]

JavaML document



Each of these representations may be more convenient to perform some operation on the program.

Collaborations between TSpaces: Projectors



JMI = JSR #30 from Java User Community
CMI = Corba Model Interchange (D. Frankell)
DI = Diagram Interchange for UML

Models revisited

- Everything is a model
  - A λ-model
  - λ meaning the specific TS
  - An XML document is an XML-model
  - A Java source program is a Java-model
  - An UML model is a MDA-model
  - etc.
- Each TS is rooted in a metametamodel defining its representation scheme
- Distinguish between intra-space and inter-space operations

Example: Databases TS

"By model we mean a complex structure that represents a design artifact, such as a relational schema, object-oriented interface, UML model, XML DTD, web-site schema, semantic network, complex document, or software configuration. Many uses of models involve managing changes in models and transformations of data from one model into another. These uses require an explicit representation of mappings between models. We propose to make database systems easier to use for these applications by making model and model mapping first-class objects with special operations that simplify their use. We call this capacity model management."

P.A. Bernstein, A.L. Levy & R.A. Pottinger MSR-TR-2000-53

Agenda

What we learn from OT and why it is not the solution

Software factories

"…

The software industry remains reliant on the craftsmanship of skilled individuals engaged in labor intensive manual tasks. However, growing pressure to reduce cost and time to market and to improve software quality may catalyze a transition to more automated methods. We look at how the software industry may be industrialized, and we describe technologies that might be used to support this vision.

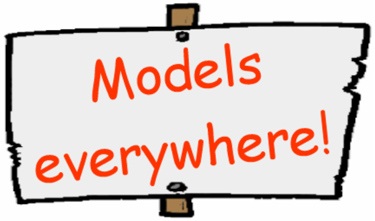
…

We suggest that the current software development paradigm, based on object orientation, may have reached the point of exhaustion, and we propose a model for its successor.

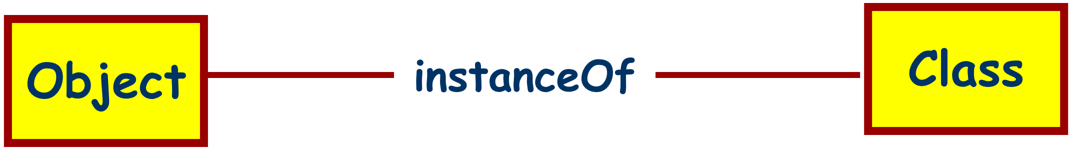
…"

J. Greenfield and K. Short

From objects to models

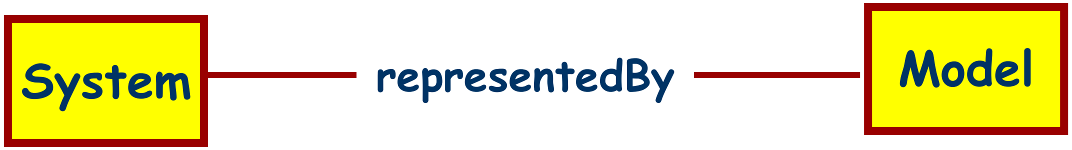


- "Everything is an object" was one of the strongest technology improvement driving principles in the last twenty years
  - As long as this principle was followed, steady progresses were achieved



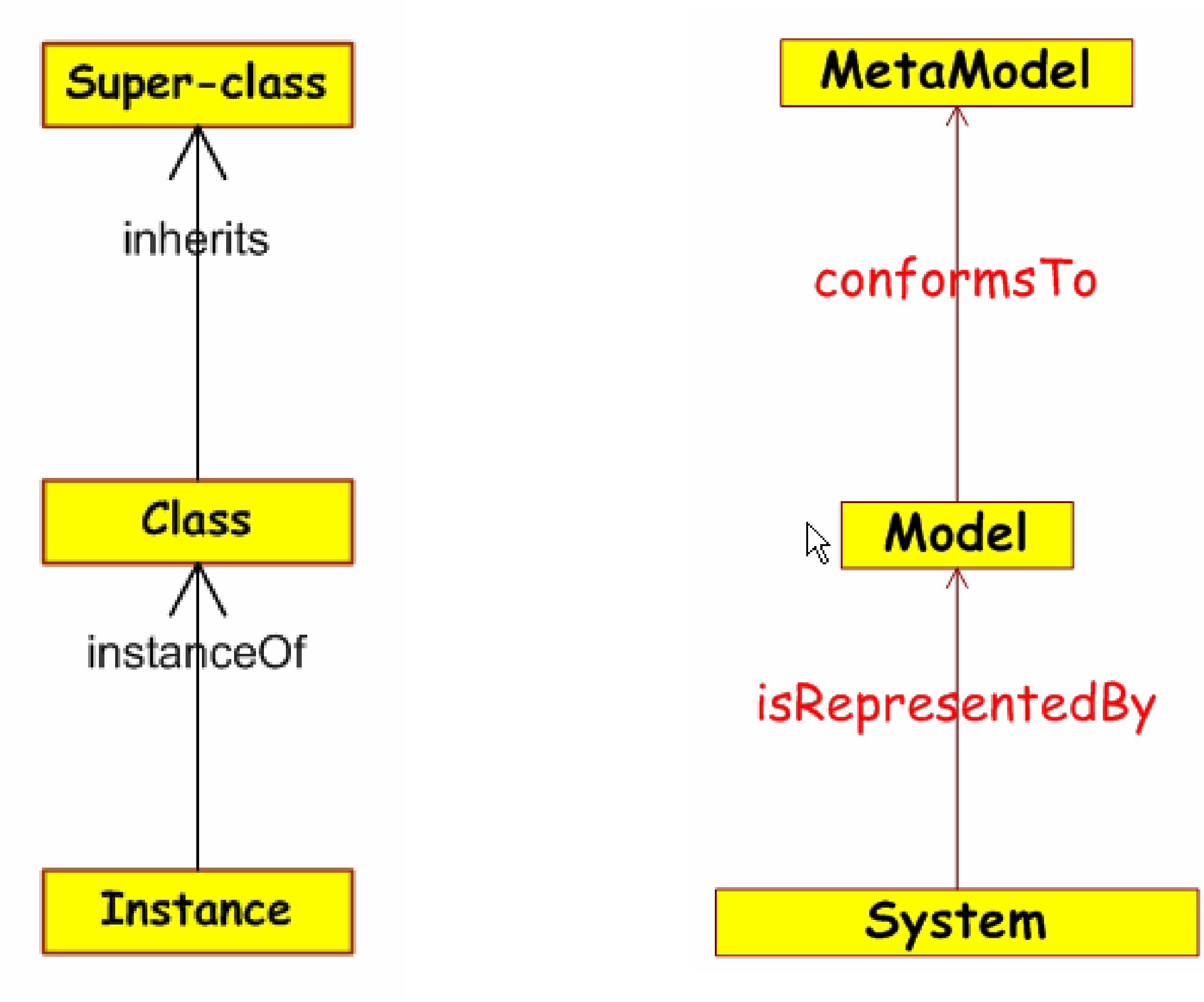
Object — instanceOf — Class

- "Everything is a model" is a current driving principle for the MDE/MDA
  - As long as we follow this principle, steady progresses may be achieved



System — representedBy — Model

Paradigm Change



Unification principle

- In the 1980's: Everything is an object
- In the 2000's: Everything is a model

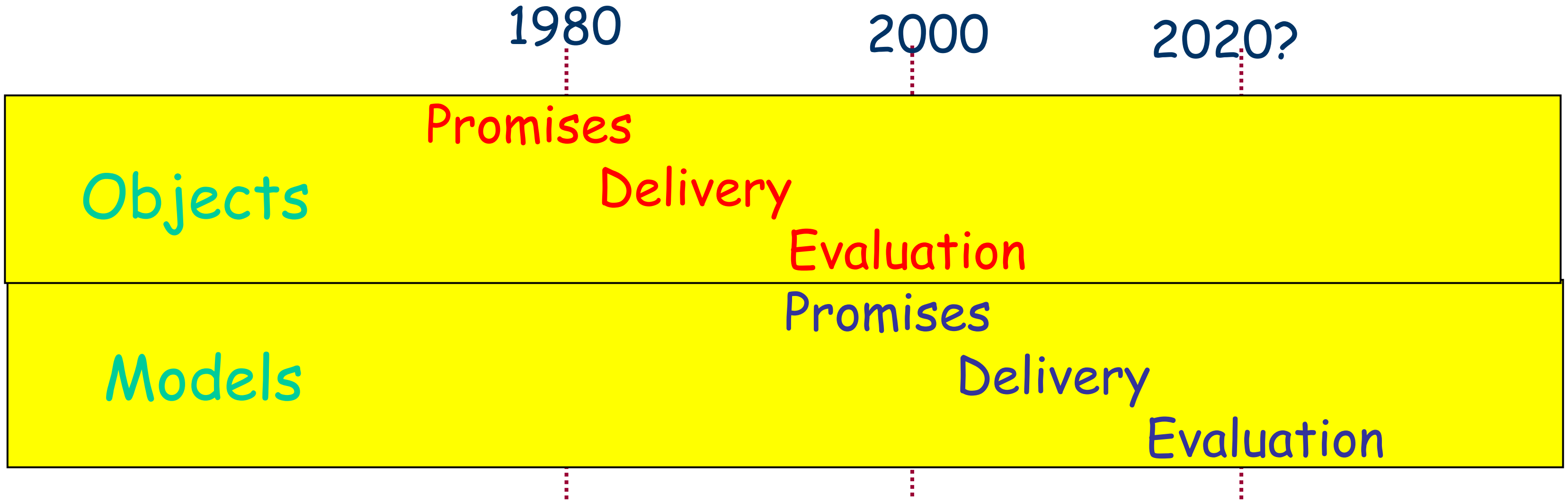
Both assertions are "basic engineering postulates".

The world is not constituted of objects,
but this helps considerably when building our systems,
that are partial images of the real or imaginary world,
if we understand clearly what is an object
from an engineering point of view.

The same applies now to models.

Summary

- Object technology realized some promises but failed to achieve others
  - Stopping the search for generality by unification may be one of the causes for this
- Model engineering is making many promises today
  - Will it be able to deliver correspondingly?
  - Sticking with the principle that "everything is a model" seems a good way to make progresses

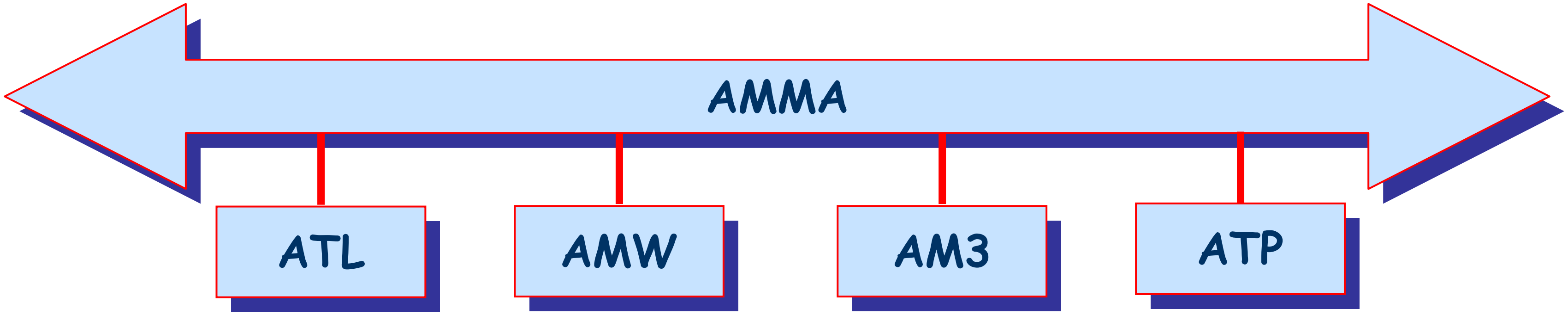


Agenda

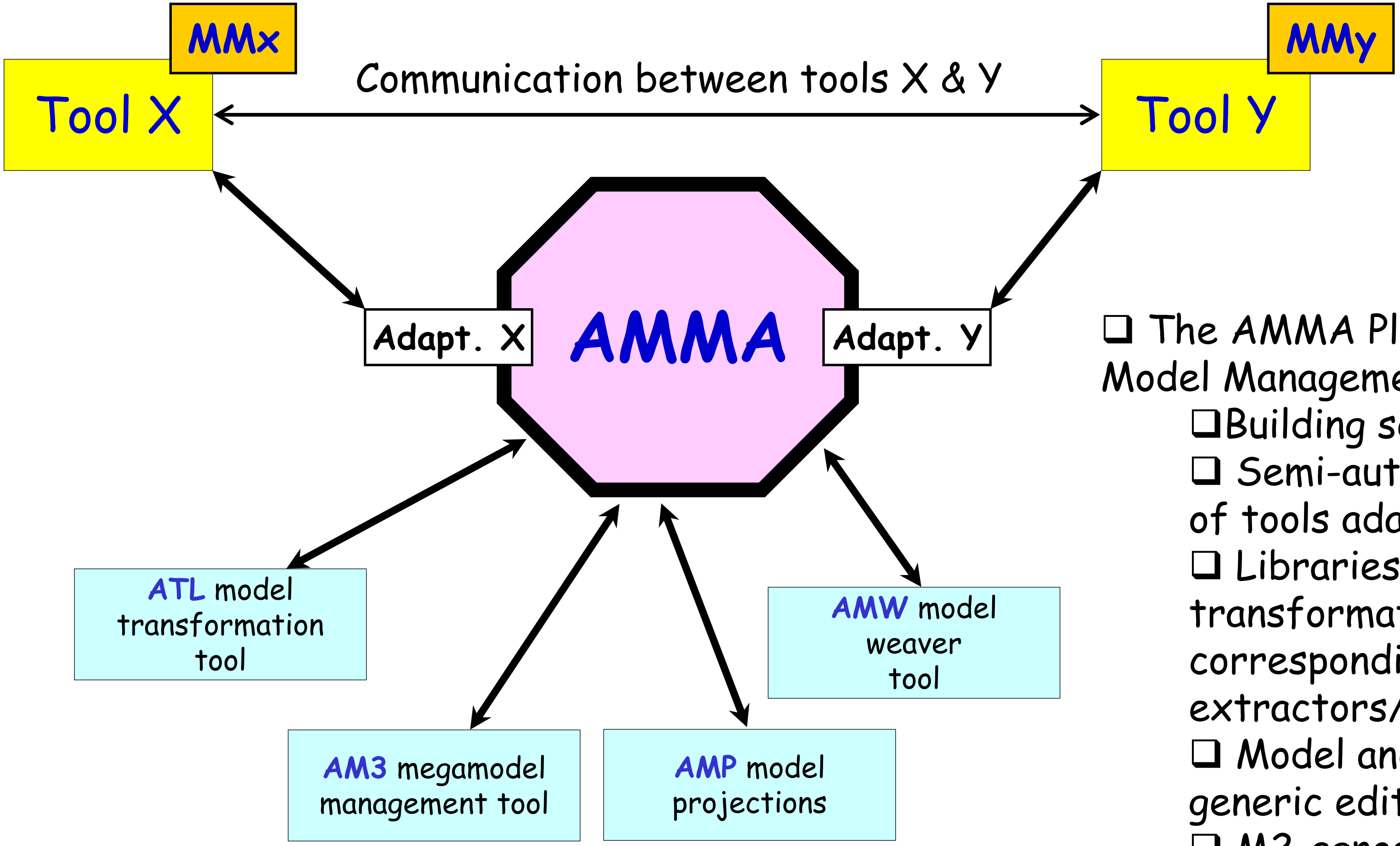
Towards an open MDE platform?

AMMA: A Lightweight Architectural Style for for Generic Model Management Platforms

- ATLAS Model Management Architecture
- Build around a minimal set of sound principles
- Defines the conventions for the various connected tools to interoperate
- Lightweight : Not reinventing CORBA
- Model-based interoperability and not Middelware-based interoperability
- Four basic blocks:



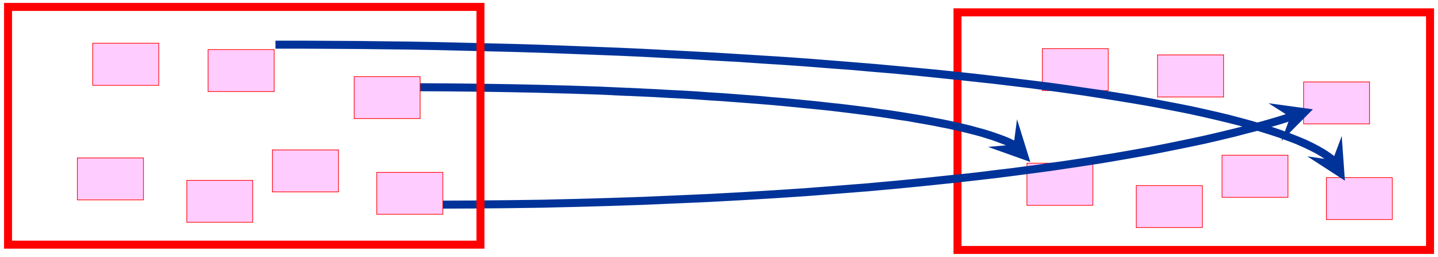
Many groups building Open MDE platforms


MMx
Tool X
Communication between tools X & Y
MMy
Tool Y
Adapt. X
AMMA
Adapt. Y
ATL model transformation tool
AMW model weaver tool
AM3 megamodel management tool
AMP model projections

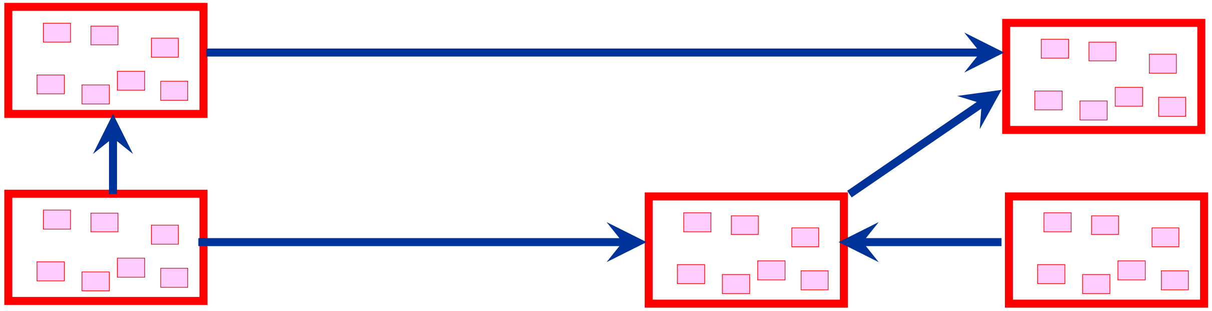
- ❑ The AMMA Platform (ATLAS Model Management Architecture)
  - ❑Building semantic bridges
  - ❑ Semi-automatic generation of tools adapters
  - ❑ Libraries of transformation and corresponding extractors/injectors
  - ❑ Model and metamodel generic editors
  - ❑ M2-agnostic and M3-agnostic tools
  - ❑ Application to data-intensive systems and software modernization

Atlas Model Management Architecture

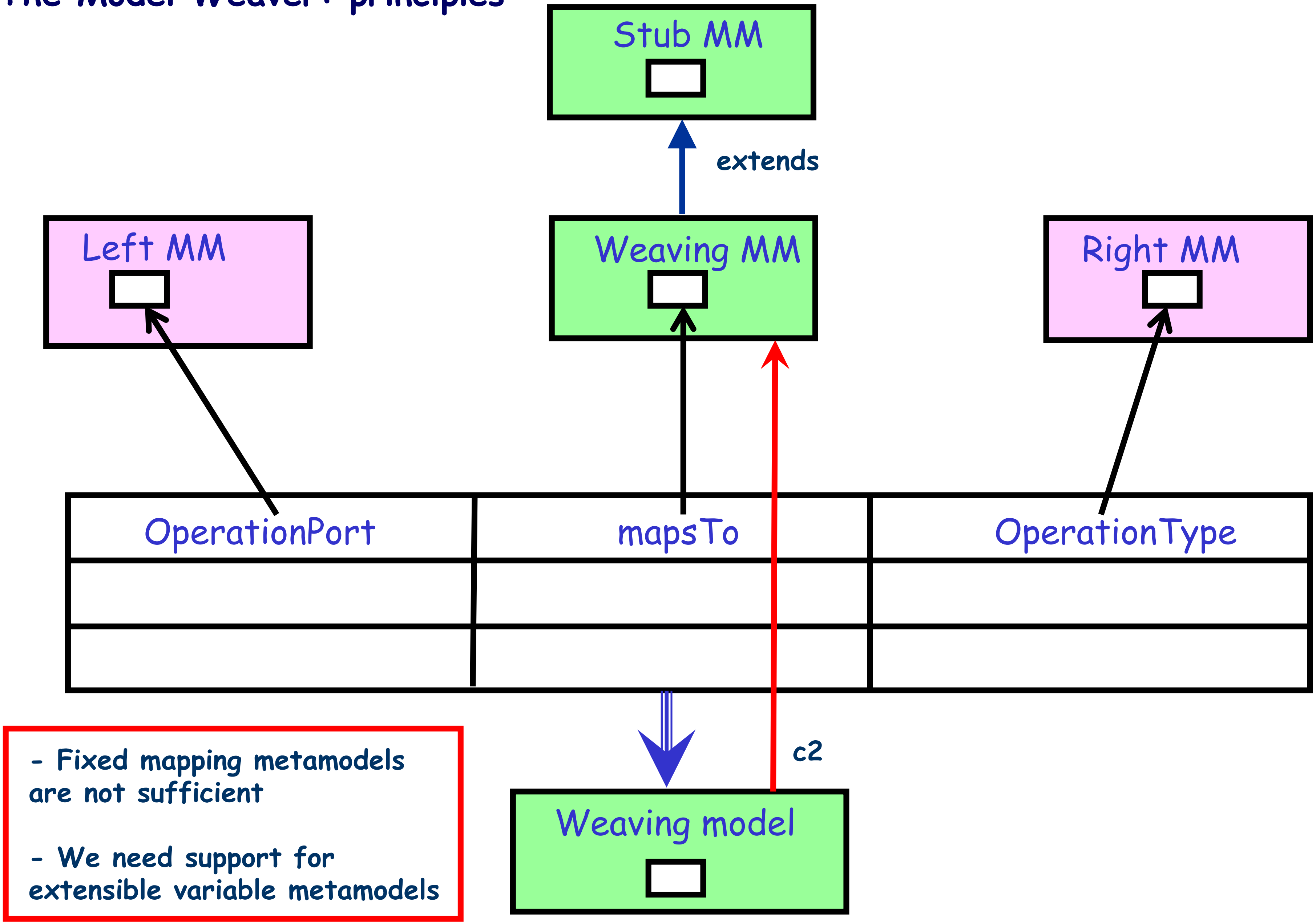
- Modeling in the small
  - Working at the level of model and metamodel elements



- Modeling in the large
  - Working with models and metamodels as global entities, for what they represent, and their mutual relations, independently of their content
  - A megamodel is a model which elements represents models, metamodels and other global entities (ako model registry with metadata on models and metamodels). A megamodel has a metamodel.



The Model Weaver: principles



Stub MM

extends

Left MM

Weaving MM

Right MM

OperationPort	mapsTo	OperationType

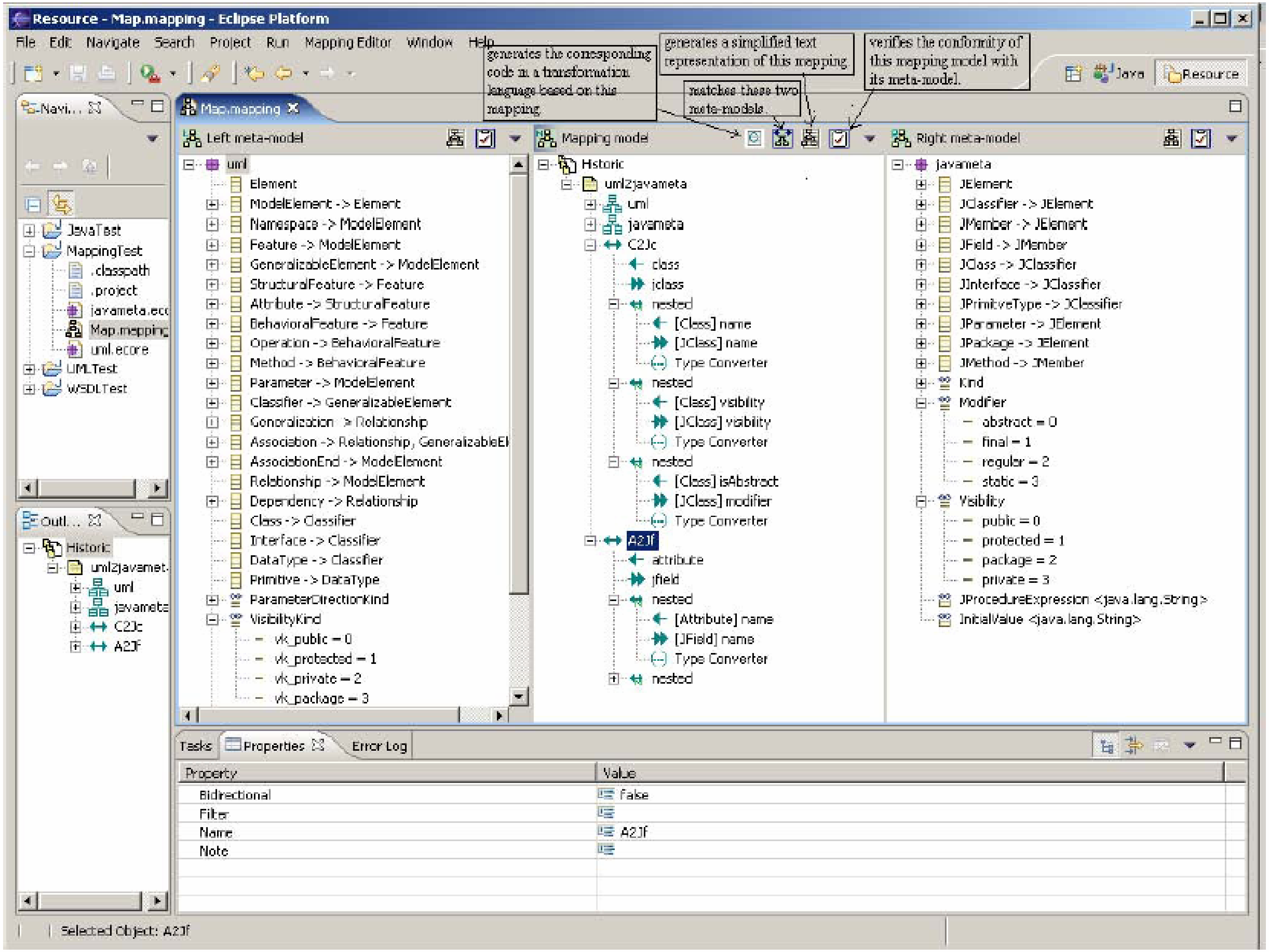
c2

Weaving model

- Fixed mapping metamodels are not sufficient

- We need support for extensible variable metamodels

The Model Weaver: first prototype


Resource - Map.mapping - Eclipse Platform
File Edit Navigate Search Project Run Mapping Editor Window Help
Java Resource
Navi...
Outl...
Map.mapping
Left meta-model
Mapping model
Right meta-model
Tasks Properties Error Log

generates the corresponding code in a transformation language based on this mapping

generates a simplified text representation of this mapping

matches these two meta-models

verifies the conformity of this mapping model with its meta-model.

```
JavaTest
MappingTest
  .classpath
  .project
  javameta.ecc
  Map.mapping
  uml.ecore
UMLTest
WSDLTest
```

```
Historic
  uml2javamet
    uml
    javameta
    C2Jc
    A2Jf
```

```
uml
  Element
  ModelElement -> Element
  Namespace -> ModelElement
  Feature -> ModelElement
  GeneralizableElement -> ModelElement
  StructuralFeature -> Feature
  Attribute -> StructuralFeature
  BehavioralFeature -> Feature
  Operation -> BehavioralFeature
  Method -> BehavioralFeature
  Parameter -> ModelElement
  Classifier -> GeneralizableElement
  Generalization -> Relationship
  Association -> Relationship, GeneralizableEl
  AssociationEnd -> ModelElement
  Relationship -> ModelElement
  Dependency -> Relationship
  Class -> Classifier
  Interface -> Classifier
  DataType -> Classifier
  Primitive -> DataType
  ParameterDirectionKind
  VisibilityKind
    vk_public = 0
    vk_protected = 1
    vk_private = 2
    vk_package = 3
```

```
Historic
  uml2javameta
    uml
    javameta
    C2Jc
      class
      jclass
      nested
        [Class] name
        [JClass] name
        Type Converter
      nested
        [Class] visibility
        [JClass] visibility
        Type Converter
      nested
        [Class] isAbstract
        [JClass] modifier
        Type Converter
    A2Jf
      attribute
      jfield
      nested
        [Attribute] name
        [JField] name
        Type Converter
      nested
```

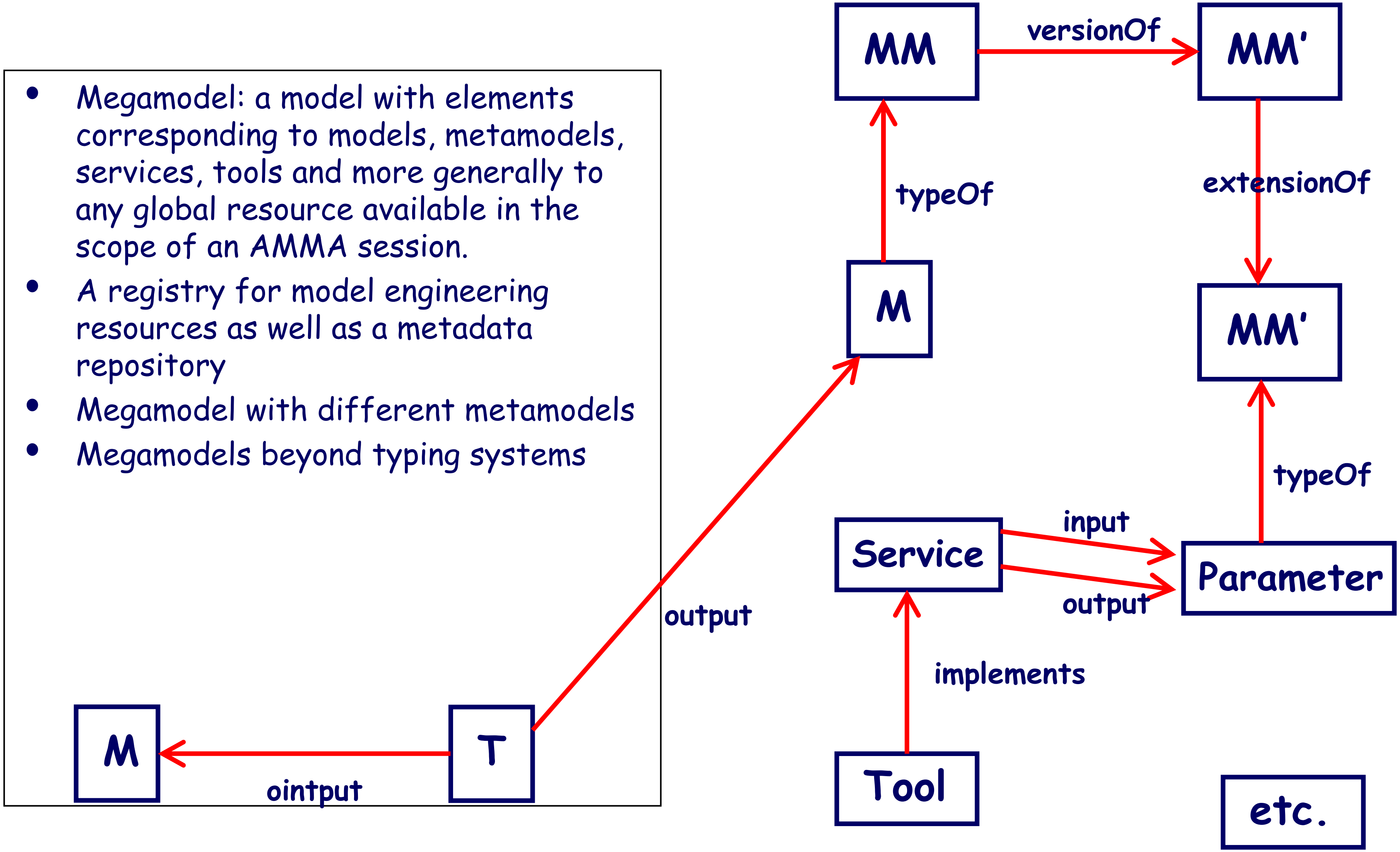
```
javameta
  JElement
  JClassifier -> JElement
  JMember -> JElement
  JField -> JMember
  JClass -> JClassifier
  JInterface -> JClassifier
  JPrimitveType -> JClassifier
  JParameter -> JElement
  JPackage -> JElement
  JMethod -> JMember
  Kind
  Modifier
    abstract = 0
    final = 1
    regular = 2
    static = 3
  Visibility
    public = 0
    protected = 1
    package = 2
    private = 3
  JProcedureExpression <java.lang.String>
  InitialValue <java.lang.String>
```

Property	Value
Bidirectional	False
Filter	
Name	A2Jf
Note	

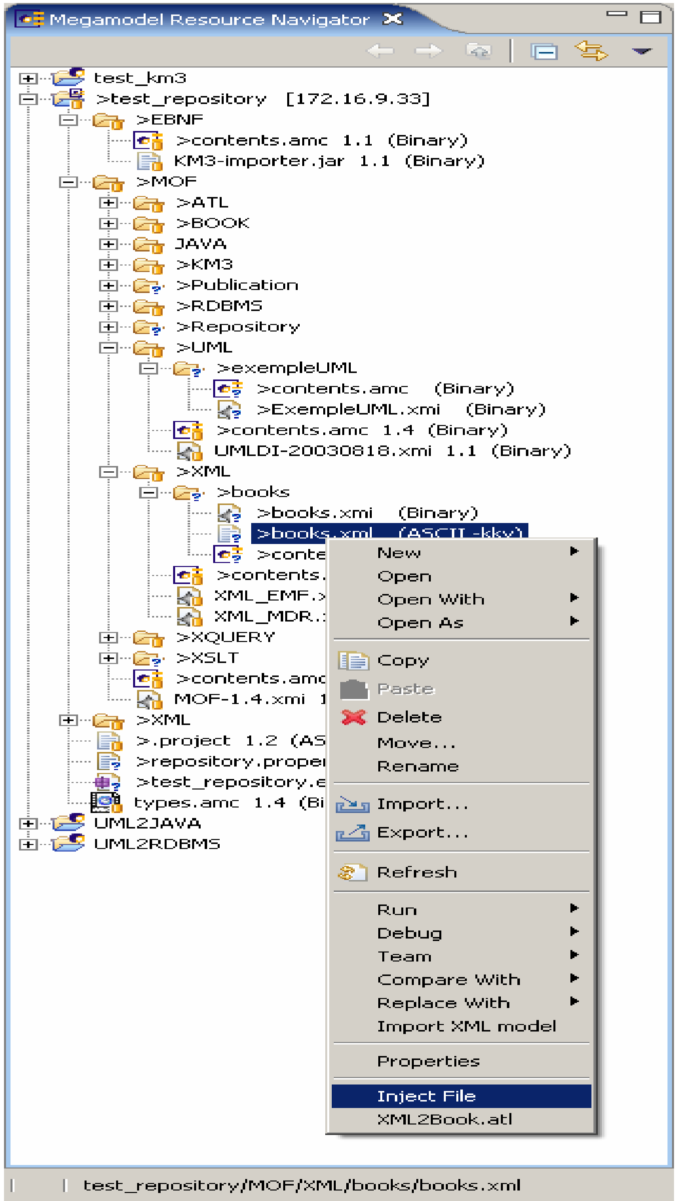
Selected Object: A2Jf

AM3: ATLAS MegaModel Management Tool

- Megamodel: a model with elements corresponding to models, metamodels, services, tools and more generally to any global resource available in the scope of an AMMA session.
- A registry for model engineering resources as well as a metadata repository
- Megamodel with different metamodels
- Megamodels beyond typing systems



Megamodel Resource Navigator

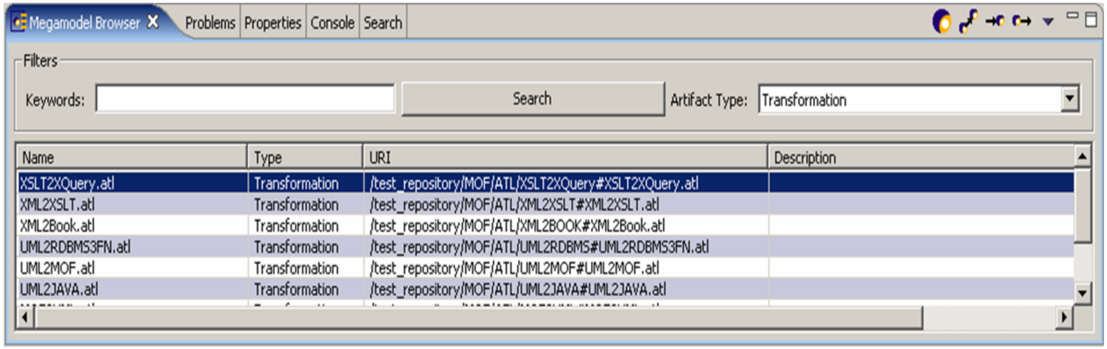


```
test_km3
>test_repository  [172.16.9.33]
  >EBNF
    >contents.amc  1.1  (Binary)
    KM3-importer.jar  1.1  (Binary)
  >MOF
    >ATL
    >BOOK
    JAVA
    >KM3
    >Publication
    >RDBMS
    >Repository
    >UML
      >exempleUML
        >contents.amc   (Binary)
        >ExempleUML.xmi   (Binary)
      >contents.amc  1.4  (Binary)
      UMLDI-20030818.xmi  1.1  (Binary)
    >XML
      >books
        >books.xmi   (Binary)
        >books.xml   (ASCII -kkv)
        >conte
      >contents.
      XML_EMF.>
      XML_MDR.:
    >XQUERY
    >XSLT
    >contents.amc
    MOF-1.4.xmi  1
  >XML
  >.project  1.2  (AS
  >repository.proper
  >test_repository.e
  types.amc  1.4  (Bi
UML2JAVA
UML2RDBMS
```

test_repository/MOF/XML/books/books.xml

- Extension of Resource View
- According to metadata, tools may be available for an element

Megamodel Browser



Keywords:

Artifact Type: Transformation

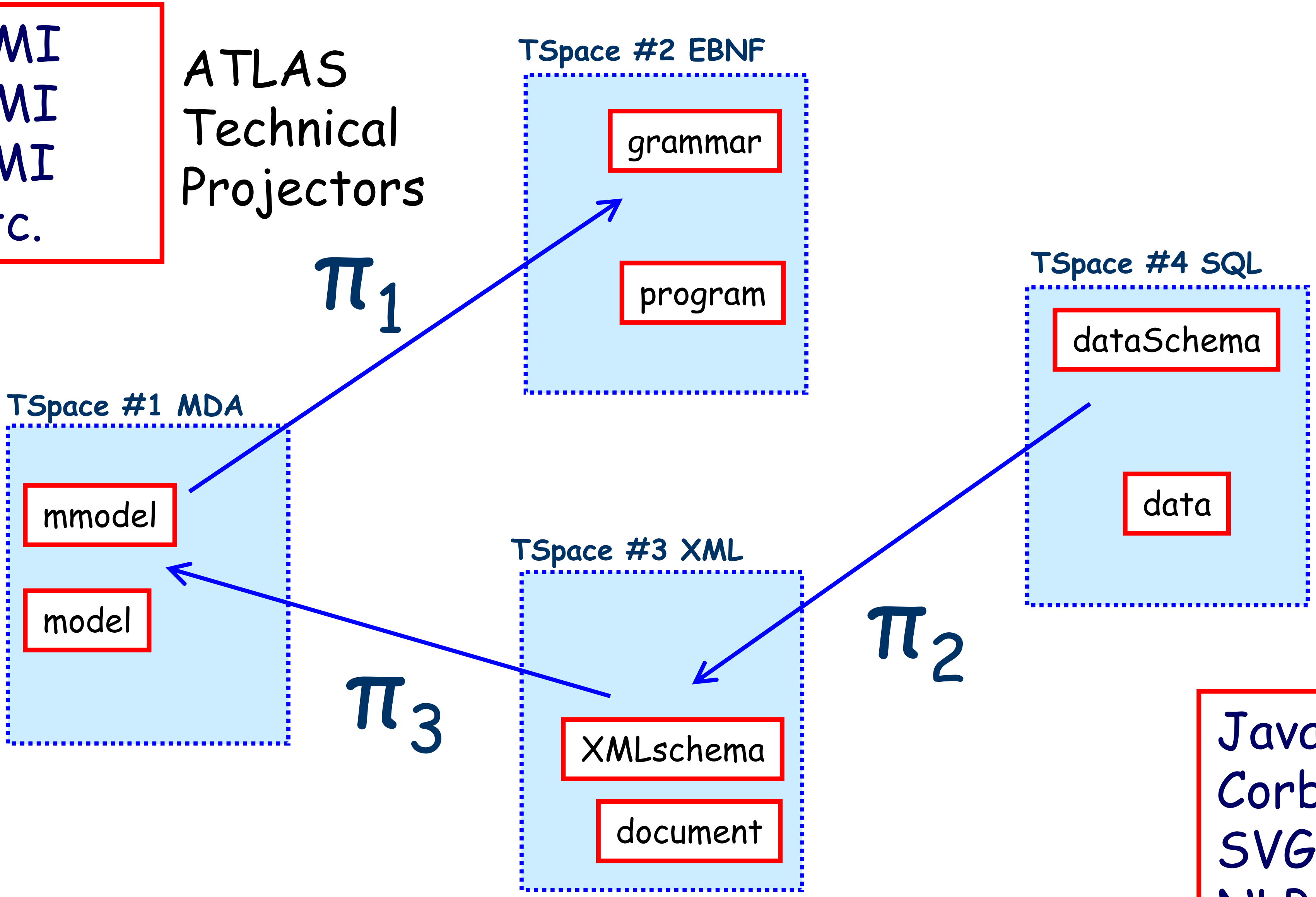
Name	Type	URI	Description
XSLT2XQuery.atl	Transformation	/test_repository/MOF/ATL/XSLT2XQuery#XSLT2XQuery.atl	
XML2XSLT.atl	Transformation	/test_repository/MOF/ATL/XML2XSLT#XML2XSLT.atl	
XML2Book.atl	Transformation	/test_repository/MOF/ATL/XML2BOOK#XML2Book.atl	
UML2RDBMS3FN.atl	Transformation	/test_repository/MOF/ATL/UML2RDBMS#UML2RDBMS3FN.atl	
UML2MOF.atl	Transformation	/test_repository/MOF/ATL/UML2MOF#UML2MOF.atl	
UML2JAVA.atl	Transformation	/test_repository/MOF/ATL/UML2JAVA#UML2JAVA.atl	

- Quick access to elements
- Four views:
  - List of artifacts
  - List of transformations (Metamodel In - Transformation - Metamodel Out)
  - List of injectors (From - Injector - Metamodel Out)
  - List of extractors (Metamodel In - Extractor - To)

ATP: TS and projectors

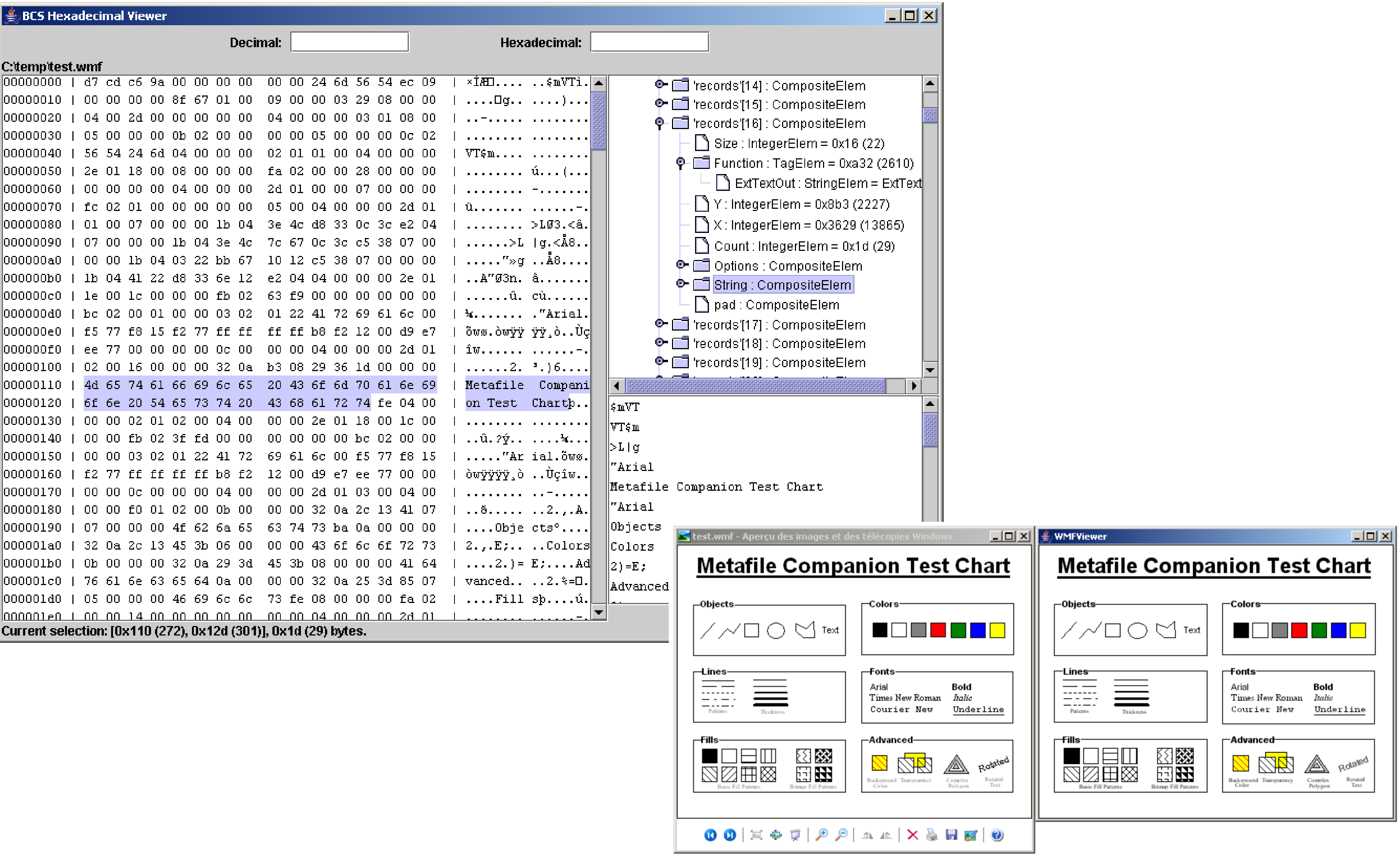
XMI
JMI
CMI
etc.

ATLAS
Technical
Projectors



Java
Corba
SVG
NLP,
etc.

Illustration : Hexadecimal and structured view of test.wmf



```
00000000 | d7 cd c6 9a 00 00 00 00  00 00 24 6d 56 54 ec 09 | ×ÍÆ□.... ..$mVTì.
00000010 | 00 00 00 00 8f 67 01 00  09 00 00 03 29 08 00 00 | ....□g.. ....)...
00000020 | 04 00 2d 00 00 00 00 00  04 00 00 00 03 01 08 00 | ..-..... ........
00000030 | 05 00 00 00 0b 02 00 00  00 00 05 00 00 00 0c 02 | ........ ........
00000040 | 56 54 24 6d 04 00 00 00  02 01 01 00 04 00 00 00 | VT$m.... ........
00000050 | 2e 01 18 00 08 00 00 00  fa 02 00 00 28 00 00 00 | ........ ú...(...
00000060 | 00 00 00 00 04 00 00 00  2d 01 00 00 07 00 00 00 | ........ -.......
00000070 | fc 02 01 00 00 00 00 00  05 00 04 00 00 00 2d 01 | ü....... ......-.
00000080 | 01 00 07 00 00 00 1b 04  3e 4c d8 33 0c 3c e2 04 | ........ >LØ3.<â.
00000090 | 07 00 00 00 1b 04 3e 4c  7c 67 0c 3c c5 38 07 00 | ......>L |g.<Å8..
000000a0 | 00 00 1b 04 03 22 bb 67  10 12 c5 38 07 00 00 00 | ....."»g ..Å8....
000000b0 | 1b 04 41 22 d8 33 6e 12  e2 04 04 00 00 00 2e 01 | ..A"Ø3n. â.......
000000c0 | 1e 00 1c 00 00 00 fb 02  63 f9 00 00 00 00 00 00 | ......û. cù......
000000d0 | bc 02 00 01 00 00 03 02  01 22 41 72 69 61 6c 00 | ¼....... ."Arial.
000000e0 | f5 77 f8 15 f2 77 ff ff  ff ff b8 f2 12 00 d9 e7 | õwø.òwÿÿ ÿÿ¸ò..Ùç
000000f0 | ee 77 00 00 00 00 0c 00  00 00 04 00 00 00 2d 01 | îw...... ......-.
00000100 | 02 00 16 00 00 00 32 0a  b3 08 29 36 1d 00 00 00 | ......2. ³.)6....
00000110 | 4d 65 74 61 66 69 6c 65  20 43 6f 6d 70 61 6e 69 | Metafile  Compani
00000120 | 6f 6e 20 54 65 73 74 20  43 68 61 72 74 fe 04 00 | on Test  Chartþ..
00000130 | 00 00 02 01 02 00 04 00  00 00 2e 01 18 00 1c 00 | ........ ........
00000140 | 00 00 fb 02 3f fd 00 00  00 00 00 00 bc 02 00 00 | ..û.?ý.. ....¼...
00000150 | 00 00 03 02 01 22 41 72  69 61 6c 00 f5 77 f8 15 | ....."Ar ial.õwø.
00000160 | f2 77 ff ff ff ff b8 f2  12 00 d9 e7 ee 77 00 00 | òwÿÿÿÿ¸ò ..Ùçîw..
00000170 | 00 00 0c 00 00 00 04 00  00 00 2d 01 03 00 04 00 | ........ ..-.....
00000180 | 00 00 f0 01 02 00 0b 00  00 00 32 0a 2c 13 41 07 | ..ð..... ..2.,.A.
00000190 | 07 00 00 00 4f 62 6a 65  63 74 73 ba 0a 00 00 00 | ....Obje ctsº....
000001a0 | 32 0a 2c 13 45 3b 06 00  00 00 43 6f 6c 6f 72 73 | 2.,.E;.. ..Colors
000001b0 | 0b 00 00 00 32 0a 29 3d  45 3b 08 00 00 00 41 64 | ....2.)= E;....Ad
000001c0 | 76 61 6e 63 65 64 0a 00  00 00 32 0a 25 3d 85 07 | vanced.. ..2.%=□.
000001d0 | 05 00 00 00 46 69 6c 6c  73 fe 08 00 00 00 fa 02 | ....Fill sþ....ú.
000001e0 | 00 00 14 00 00 00 00 00  00 00 04 00 00 00 2d 01 | ........ ......-.
```

Current selection: [0x110 (272), 0x12d (301)], 0x1d (29) bytes.

```
'records'[14] : CompositeElem
'records'[15] : CompositeElem
'records'[16] : CompositeElem
    Size : IntegerElem = 0x16 (22)
    Function : TagElem = 0xa32 (2610)
        ExtTextOut : StringElem = ExtText
    Y : IntegerElem = 0x8b3 (2227)
    X : IntegerElem = 0x3629 (13865)
    Count : IntegerElem = 0x1d (29)
    Options : CompositeElem
    String : CompositeElem
    pad : CompositeElem
'records'[17] : CompositeElem
'records'[18] : CompositeElem
'records'[19] : CompositeElem
```

```
$mVT
VT$m
>L|g
"Arial
Metafile Companion Test Chart
"Arial
Objects
Colors
2)=E;
Advanced
```

Agenda

Conclusions and perspectives

Conclusions

- **Model engineering is the future of object technology**
  - As object and classes were seen in the 80's as "first class entities", with libraries of several hundred of classes hierarchically organized, models and metamodels are beginning to be considered alike in the 2000's.
  - Libraries of hundreds of domain specific models and metamodels of high abstraction are beginning to appear. Each such metamodel contains a number of concepts and relations.
  - Tools will be needed to work with these vast libraries of models and metamodels. These tools will be much different of present CASE tools and class browsers. They will interoperate on open platforms (EMF or VisualStudio based). Many of them will be partially automatically generated from metamodels.

Conclusions

- The problems of understanding what is a good metamodel for a PIM, a PSM or a PDM may take longer to settle than initially planned. The concept of PSM itself may be questionable and is probably still ill-defined.
- The model transformation operation is beginning to be understood (QVT++). Other operations like model weaving still need research work. The global relations between models should be considered.

Conclusions

- What could kill the MDE? Only two things:
  - Lack of modesty
    - Overselling
    - Hiding difficult open problems
    - Claiming that everything is simple and under control
    - etc.
  - Lack of ambition
    - Ad-hoc solutions
    - No research or insufficient research.
    - "UML case tool vendor" restricted
    - If the MDE does not fly, then other technology spaces will harbor similar ideas

Thanks

- Questions?
- Comments?

http://www.sciences.univ-nantes.fr/lina/atl/

Jean Bézivin

Jean.Bezivin{noSpamAt}lina.univ-nantes.fr

Equipe ATLAS, INRIA & LINA, Nantes, France