

Uni Fribourg
Informationssystem-Entwicklung
An Advanced Lecture in MDA

Milan Ignjatovic

31.05.2005

Agenda

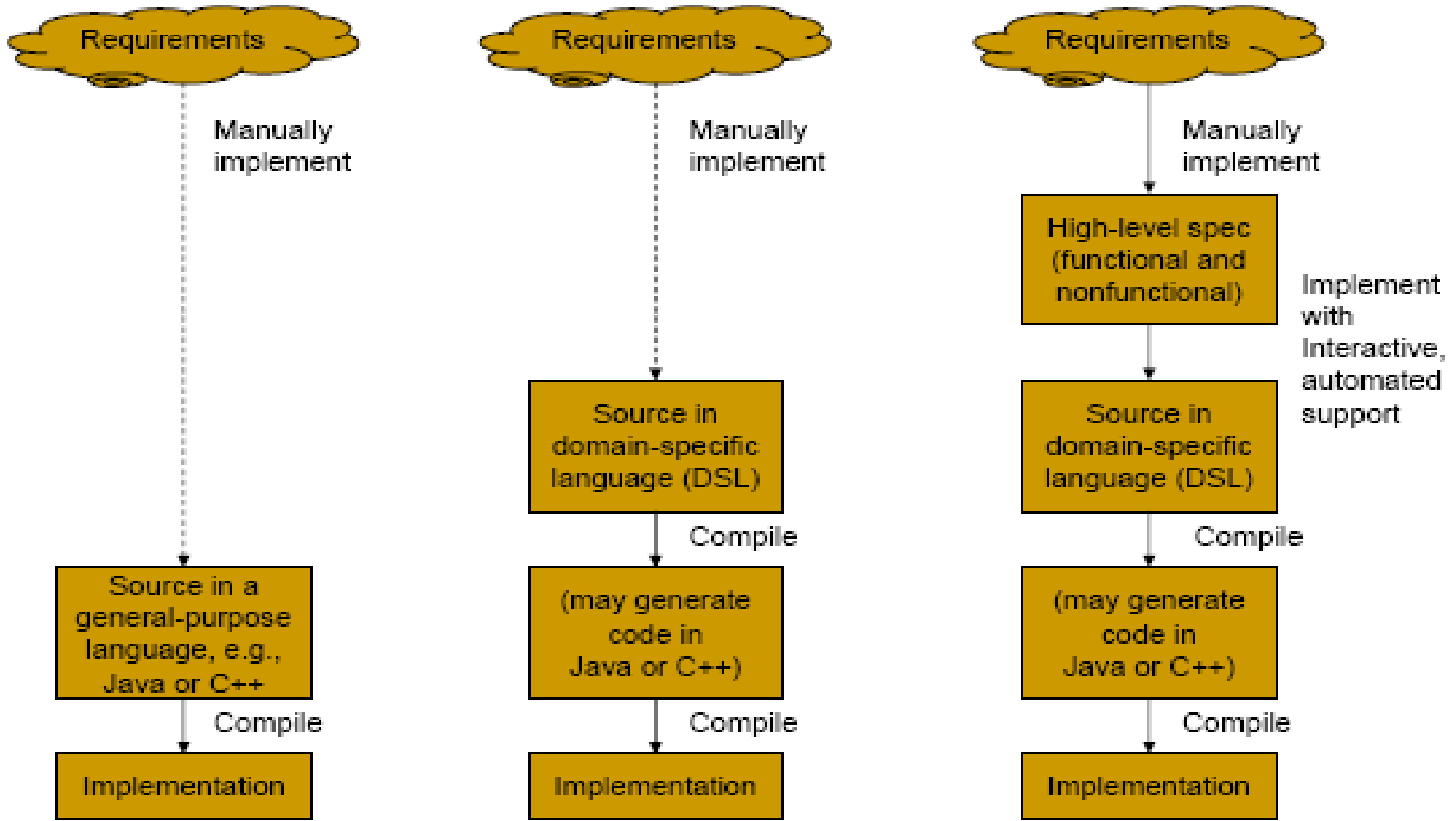
- **MDA Basics**
- Metamodelling
- Model Transformation
- Tools
- Demonstration
- MDE
- Discussion

The Saga

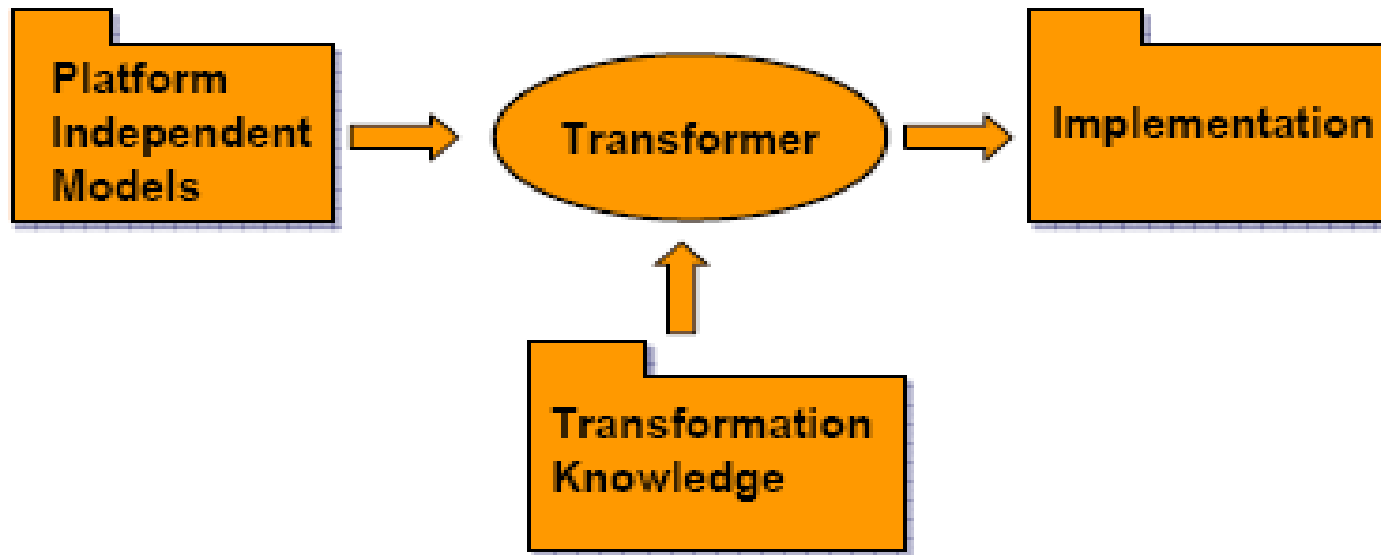
- Anaxagore de Clazomènes:
"Rien ne se crée, rien ne se perd tout se transforme"
(n'est pas de Lavoisier mais de Anaxagore de Clazomènes, philosophe de l'antiquité)
- Lavoisier:
"Car rien ne se crée, ni dans les opérations de l'Art, ni dans celles de la Nature, et l'on peut en principe poser que dans toute opération, il y a une égale quantité de matière avant et après l'opération, que la qualité et la quantité des principes est la même, et qu'il n'y a que des changements, des modifications"

Source: <http://histoirechimie.free.fr/chap05.htm>

Automation Overview

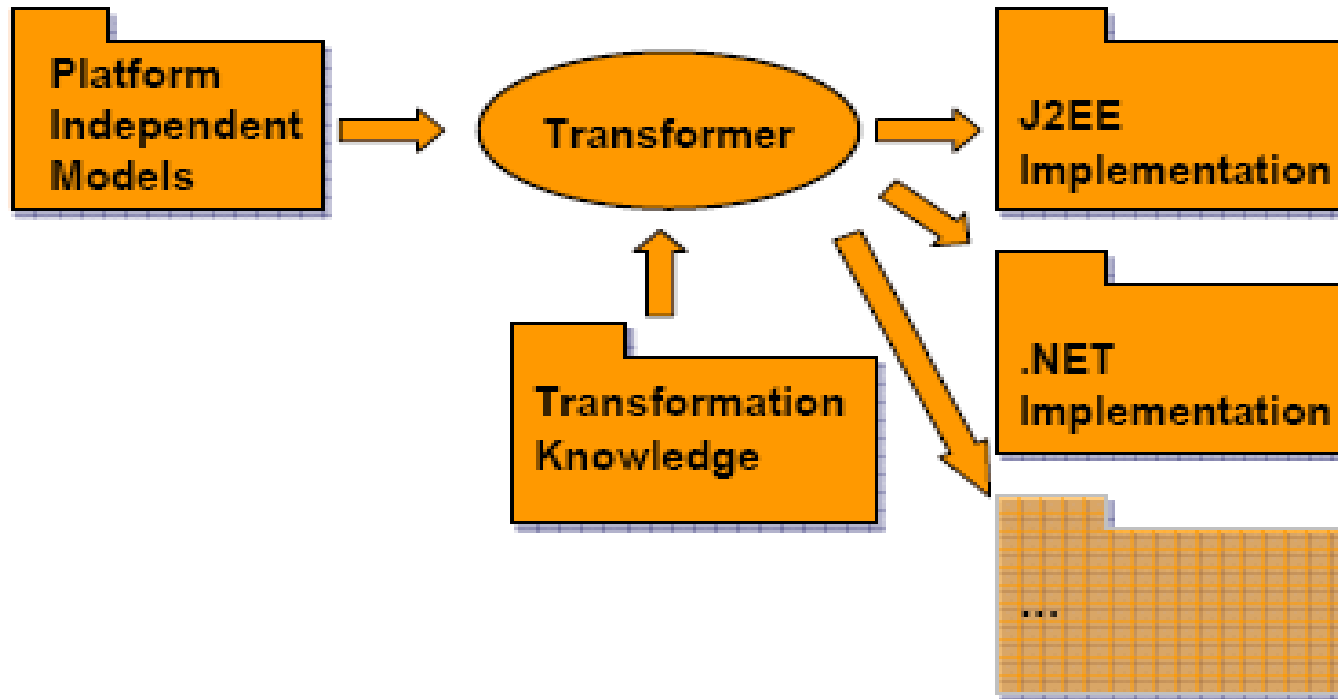


The Vision



- Platform Independent Models are used to specify structure, behaviour i.e. define functionality
- Transformations are iteratively used to transform PIMs into PSMs and finally deliver running systems

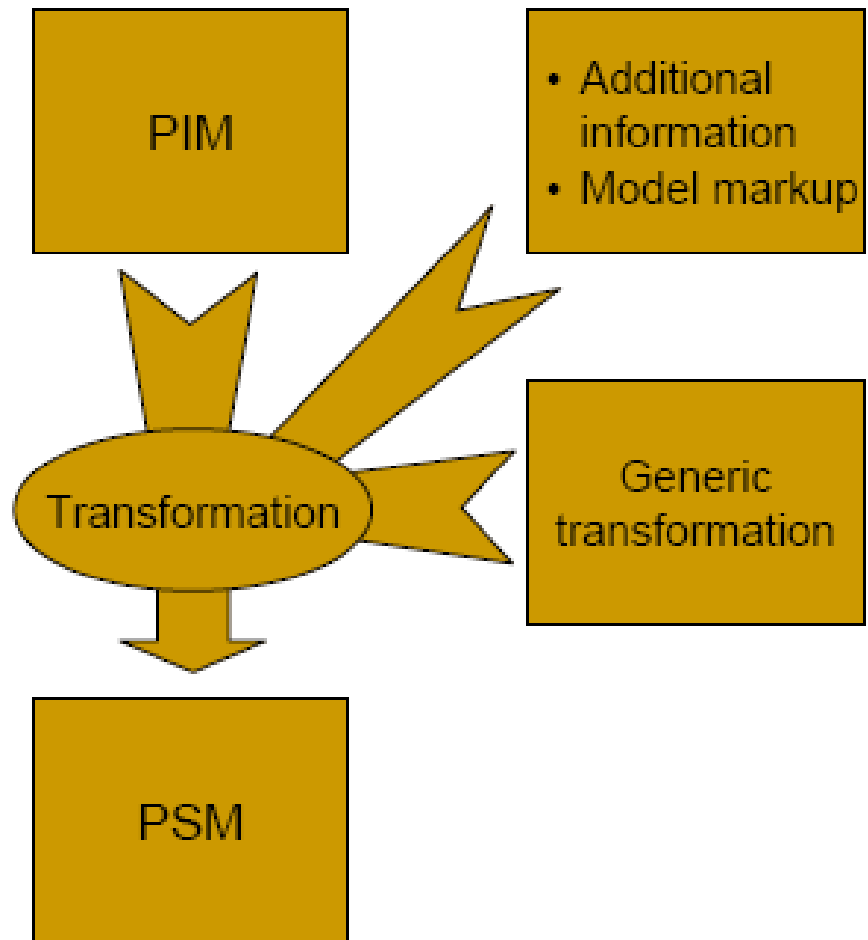
The User's Dream



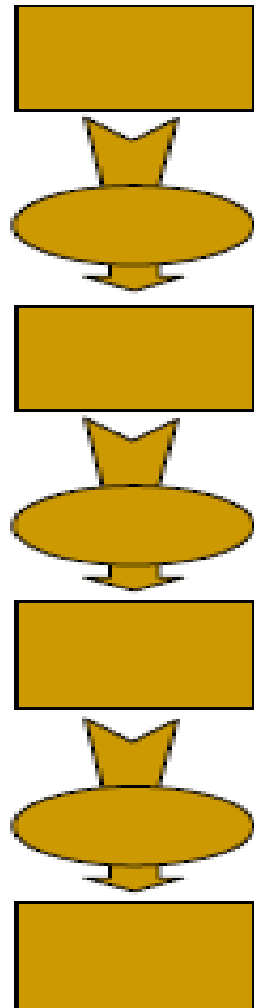
- A single PIM may be programmatically transformed for deployment on different platforms

MDA Pattern

- Generic transformation: best practices, design patterns, technology patterns (J2EE), customisation of patterns
- Additional information: model compiler options
- Model markup: define the transformation of modelling elements i.e. one markup for one platform => Markup not in PIM



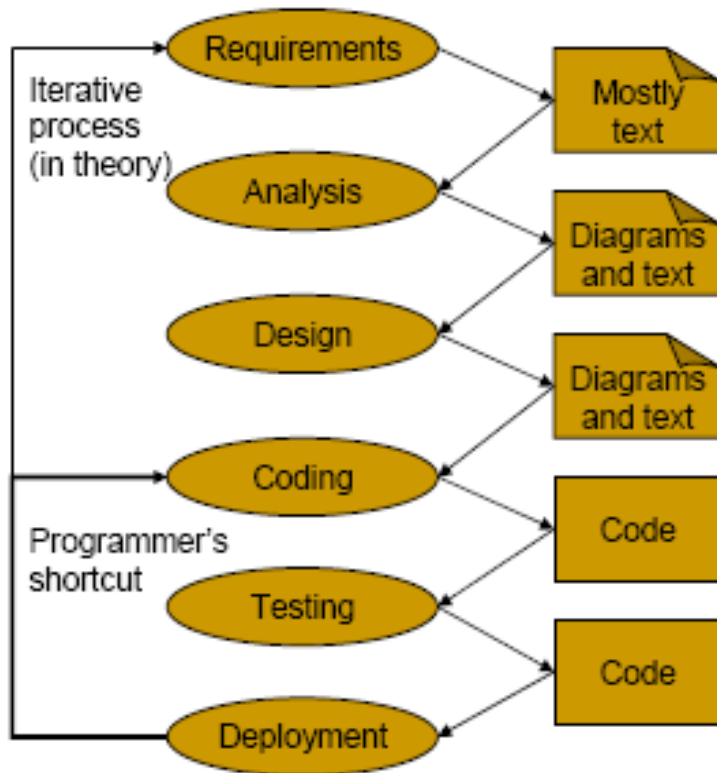
MDA Pattern - pipelined



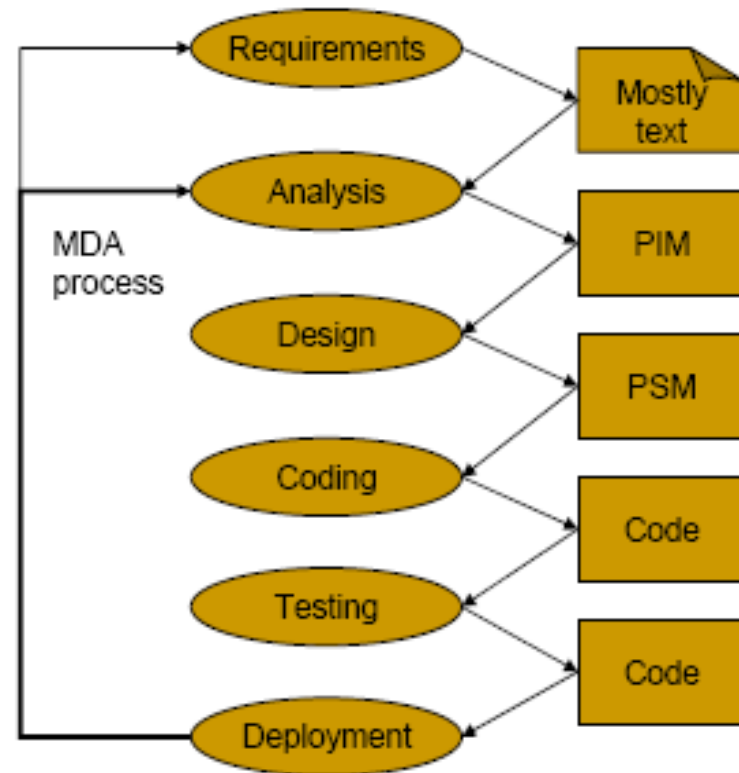
- The MDA Pattern can be applied iteratively, many times over
- PIMs and PSMs are relative concepts depending in which pipeline stage we are currently in

Development process issues

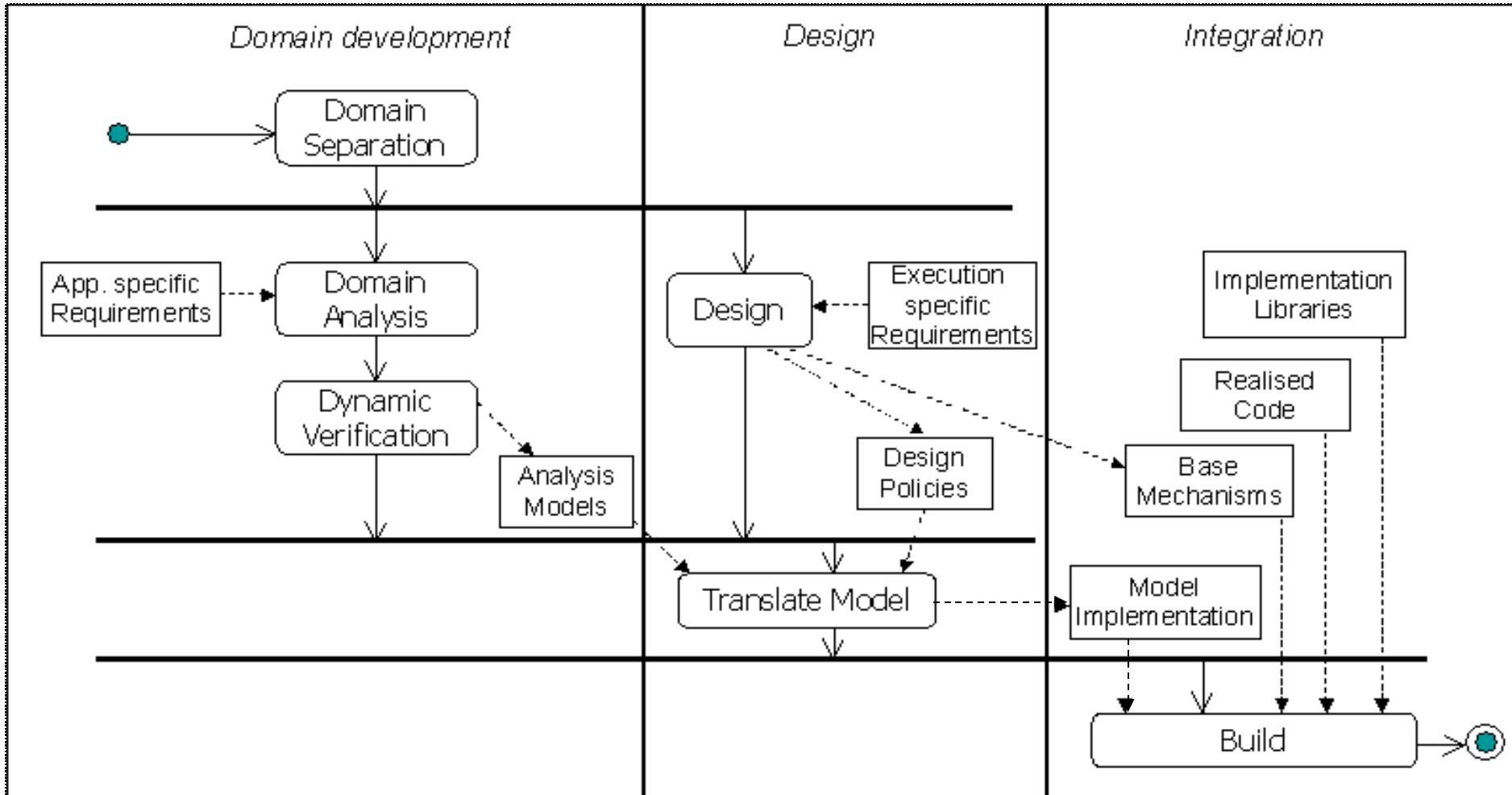
Traditional lifecycle



MDA lifecycle

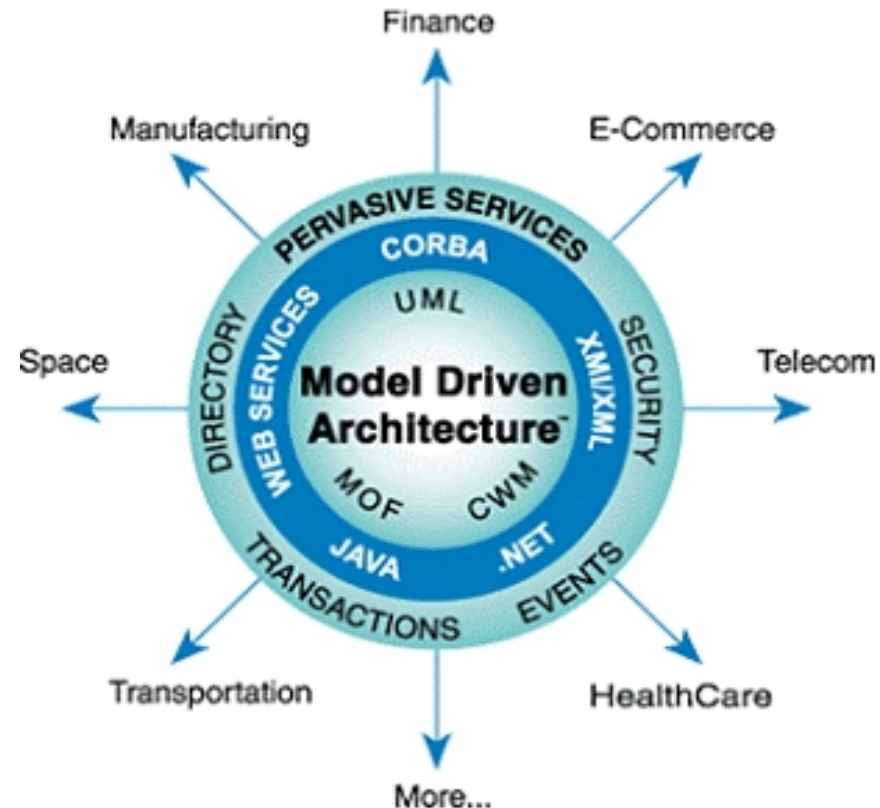


Development process: MBSE



MDA related standards

- Modelling is done in UML
- Meta-modelling is done using MOF
- Action semantics is used to define behaviour
- Action language – an instance of action semantics
- Standard model interchange format is XMI includes diagram information
- Common warehouse model, CWM



MDA Advantages

- Preserves the knowledge investment:
 - domain knowledge i.e. PIMs, Meta-models for a 10-20 year lifecycle
- Efficiency during development:
 - Transformers process models so that most of the implementation is generated
- Quality of implementation
 - Use of proven architectures, patterns...
- Extensibility, Maintenance, Documentation...
 - traceability between Specification and Implementation is completely available
 - traceability between Analysis and Design models is available

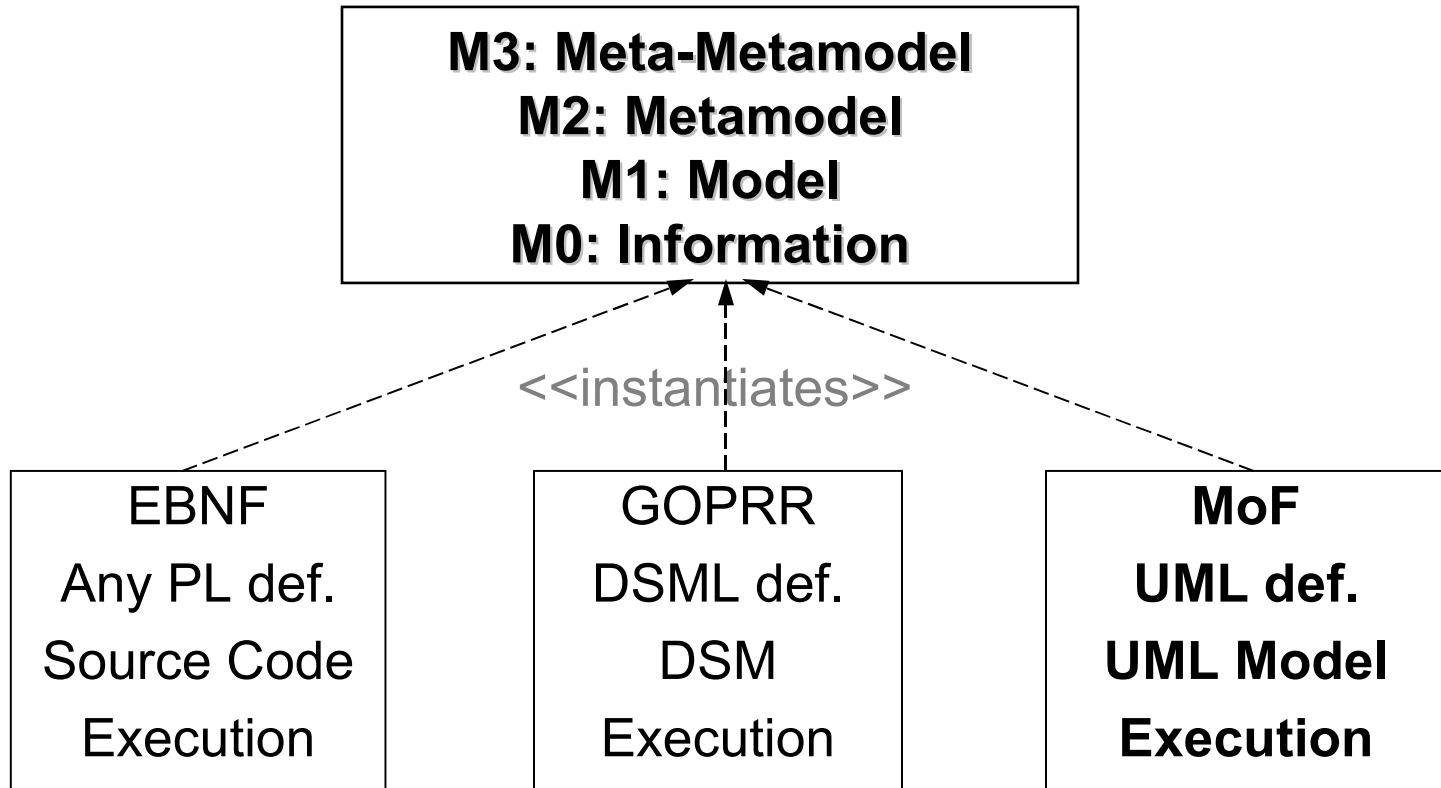
Agenda

- MDA Basics
- **Metamodelling**
- Model Transformation
- Tools
- Demonstration
- MDE
- Discussion

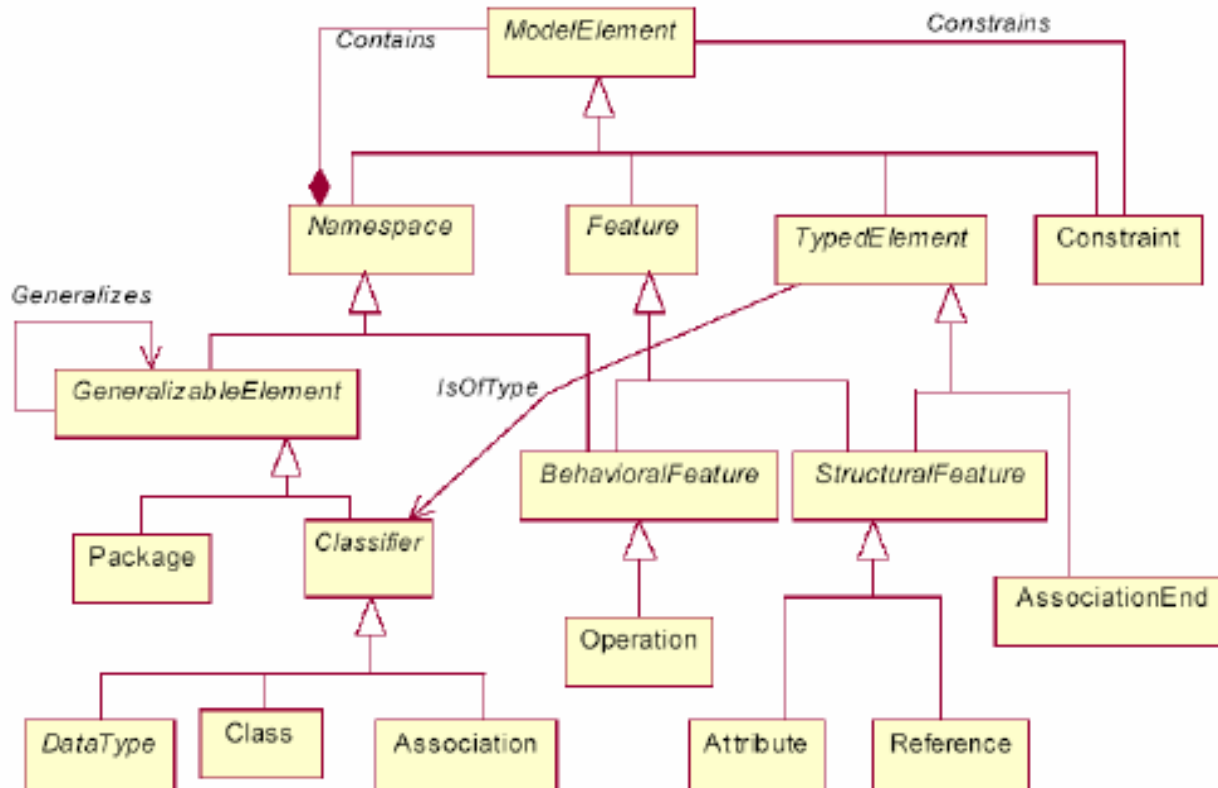
The basic premise

- There will be more than one modelling language
- Each specific modelling language will be used to model specific system aspects i.e. viewpoints
- Different modelling languages will have different definitions
- How do we communicate now?
- A modest degree of commonality is achievable by using a single (meta) language to define different modelling languages. It forms the basis for translation between different modelling languages
- Metamodel is a model of a modelling language

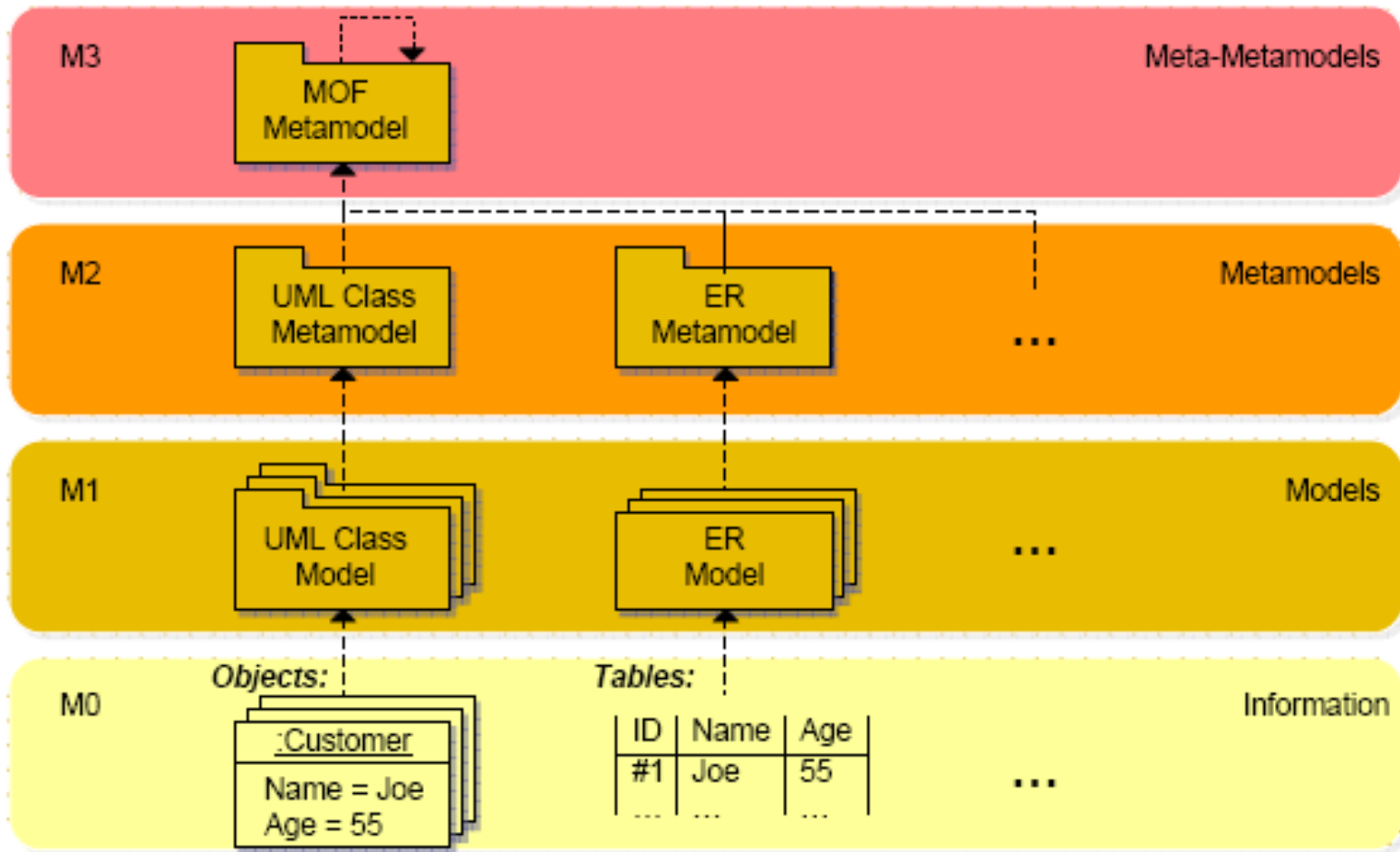
The 4 Metalevels



MOF is used to define UML



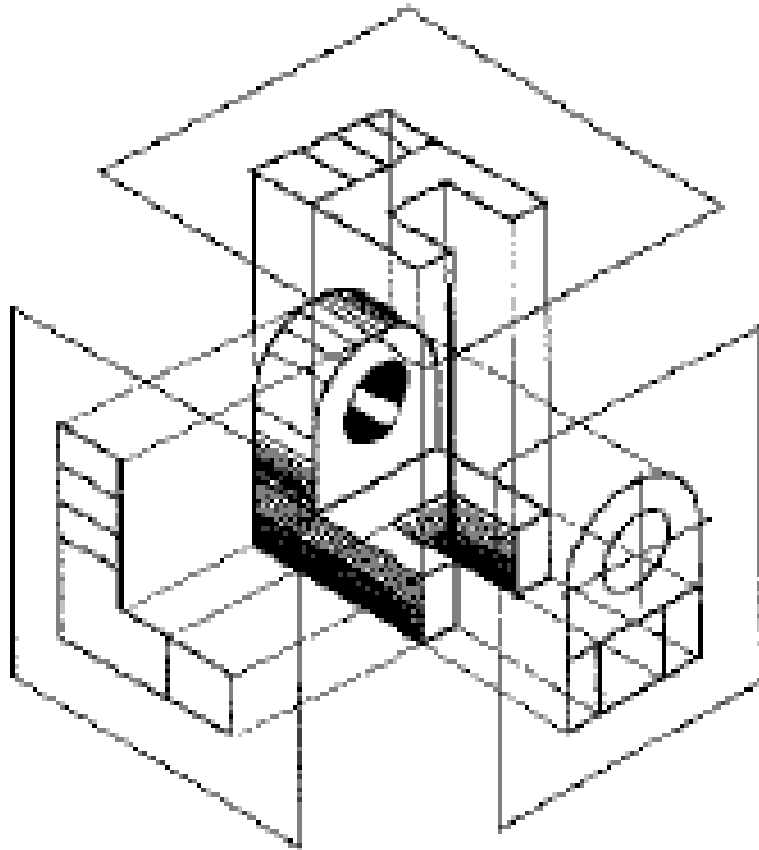
UML Meta-Stack



UML issues

- UML attempts to unify the best of all modelling practices found in the history of software engineering
- It is a general purpose modelling language
 - Advantage: allows almost everything to be modelled
 - Disadvantage: has grown to be extremely large and complex
- UML cannot provide inherent (predefined) support for every application domain
- UML2.0 has been re-engineered:
 - To allow better meta-modelling
 - Allows UML to be directly extended by use of MOF
 - This capability brings a shift from Profiles towards “MOF family of languages” paradigm within UML itself (Wim Bast, Compuware et al.)

Viewpoints

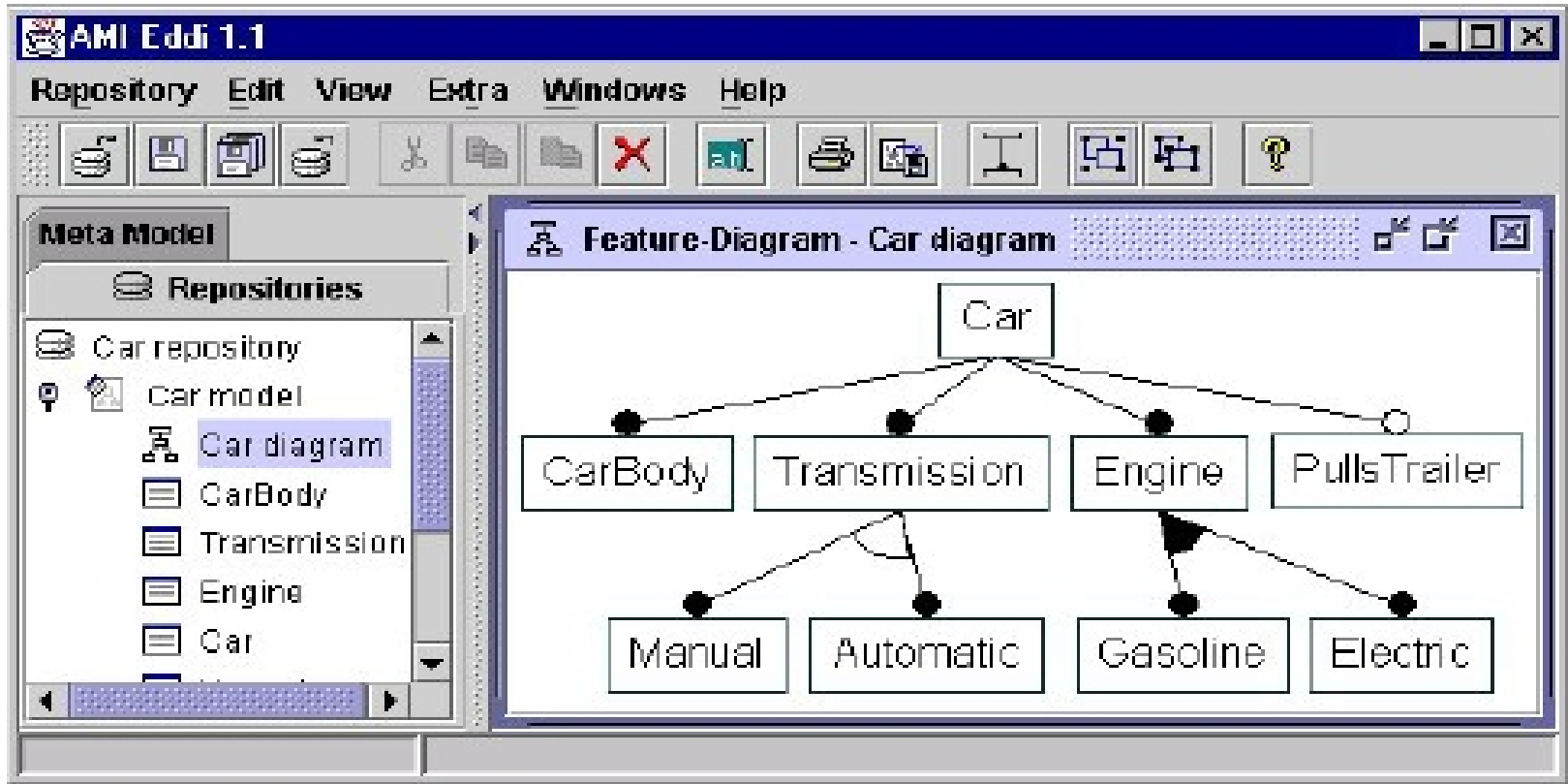


- System models are organised into multiple views i.e. different abstractions, aspects
- Each view conforms to some viewpoint that defines an appropriate modelling language
- Each viewpoint is relevant to some stakeholder

The ideal MDA application

- Use an MDA modelling tool to define a viewpoint (DSL):
 - draw a MOF meta-model and provide well-formedness rules in OCL
 - Define your concrete syntax and editing behaviour
 - Provide semantics of your MOF model by defining transformations PIM->PSM (e.g. to code)
- Package all your work as a DSL plug-in
- Load your DSL plug-in in an MDA Tool (e.g. Eclipse)
- Load other DSLs for all your required viewpoints as dictated by the application

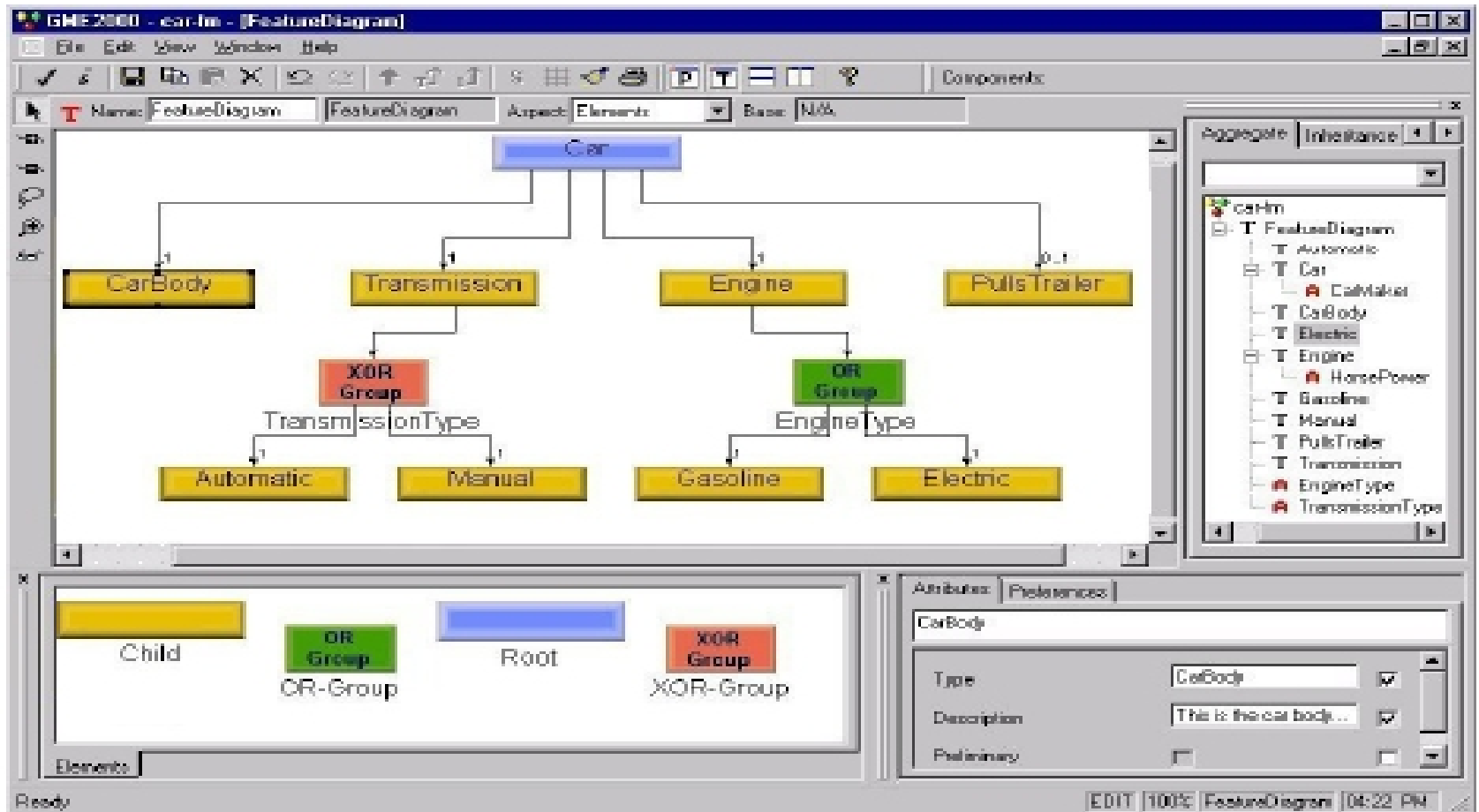
The result may look like this...



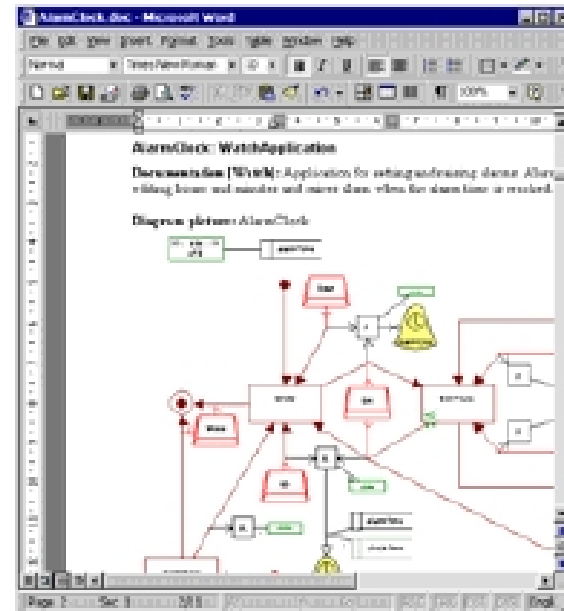
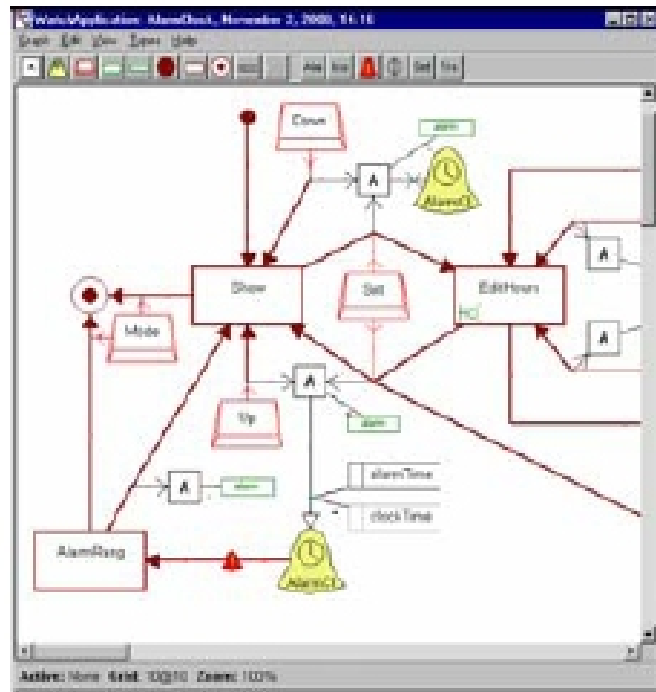
Legend:



Or...Feature Model in GME



Or... in MetaEdit+



```
import java.util.*;

public class AlarmClock extends AbstractWatchApplication {
    public int alarmTime = new int();

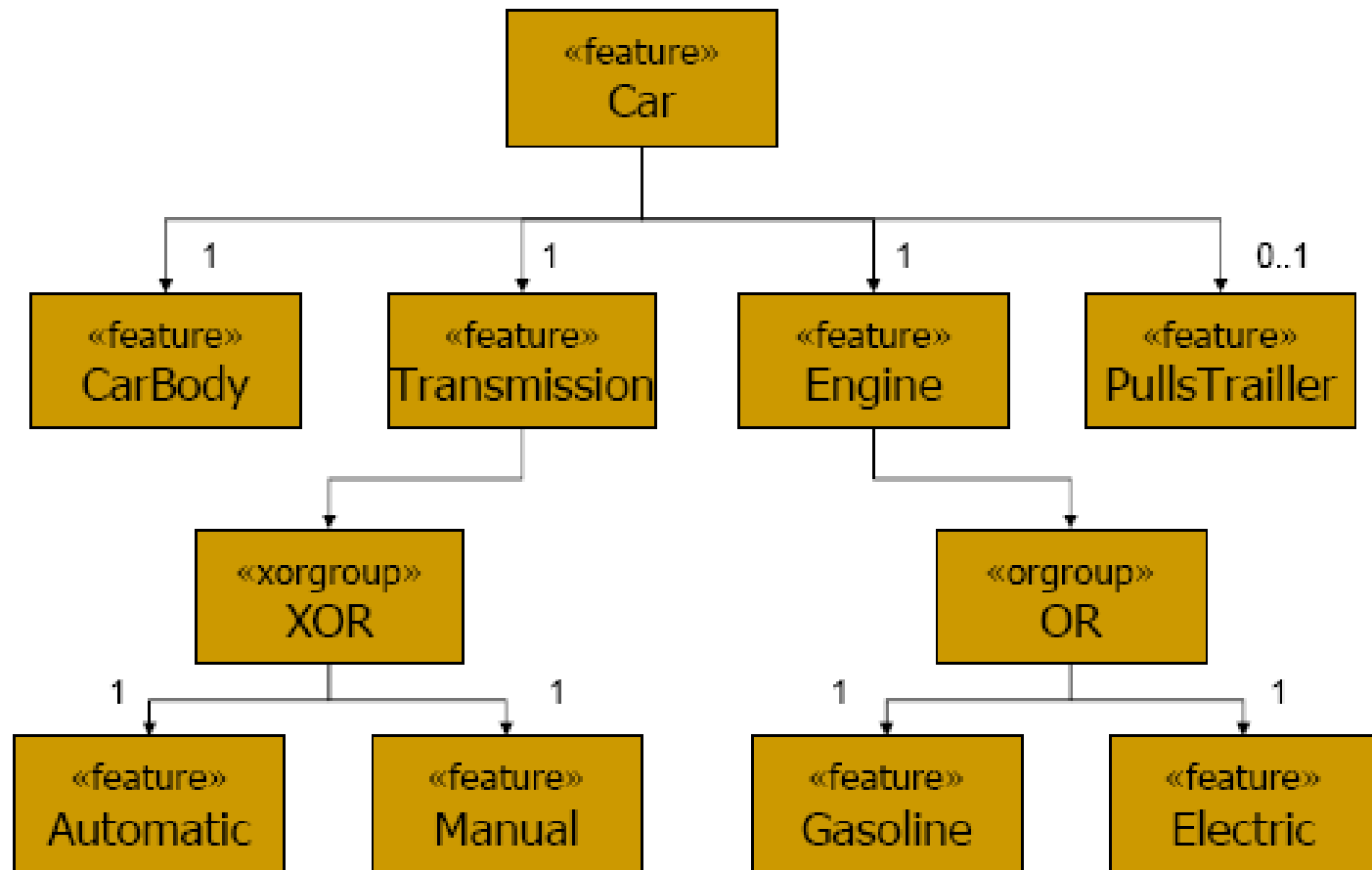
    public int getAlarmTime() {
        return alarmTime;
    }

    public void setAlarmTime(int k) {
        alarmTime = k;
    }

    public AlarmClock(ExecutionContext ec, WatchApplet) {
        super(ec, watchApplet);
        addTransition ("Show", "Set", "add_S01", "S
```



Or...a UML Profile (stereotypes)



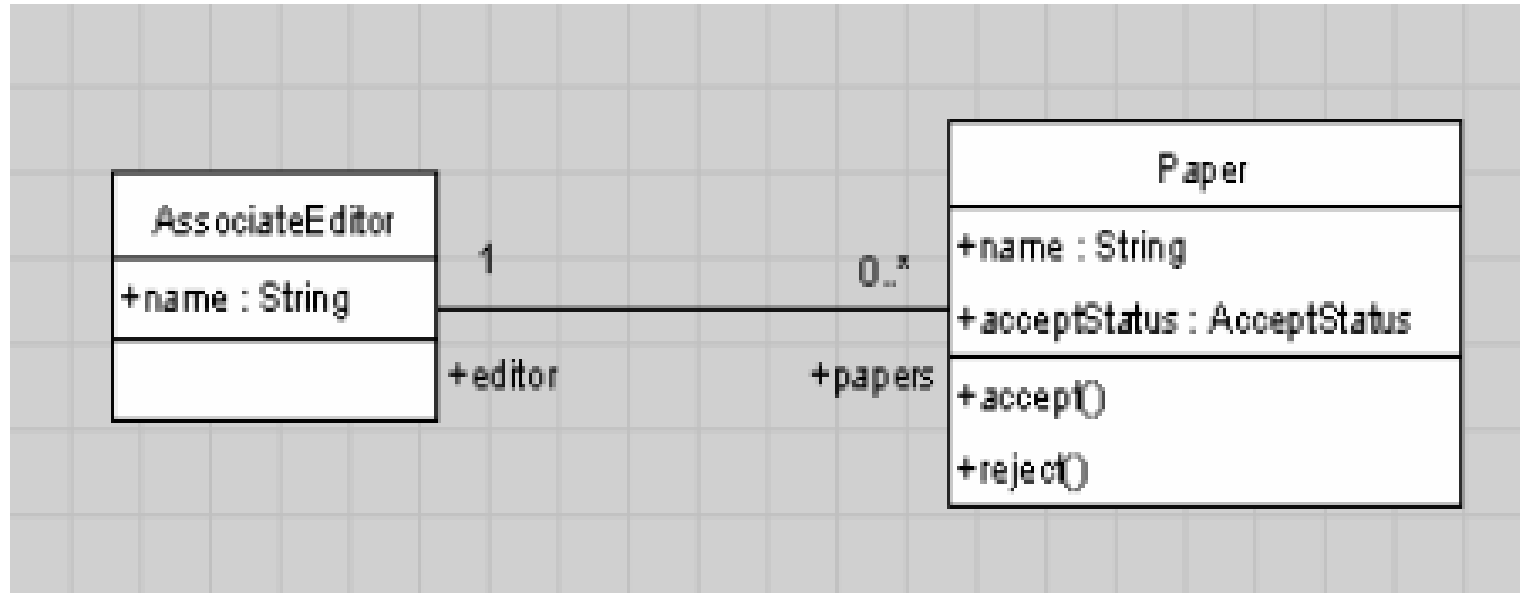
Agenda

- MDA Basics
- Metamodelling
- **Model Transformation**
- Tools
- Demonstration
- MDE
- Discussion

Model transformations

- Used to compile Models
 - Programmatic PIM to PSM transformation
- Models are used to scope complexity e.g. high-level models vs. low-level models
 - Models are defined for specific viewpoints
- Model evolution is possible
 - PIM to PIM refactoring

Simple Transformation Example

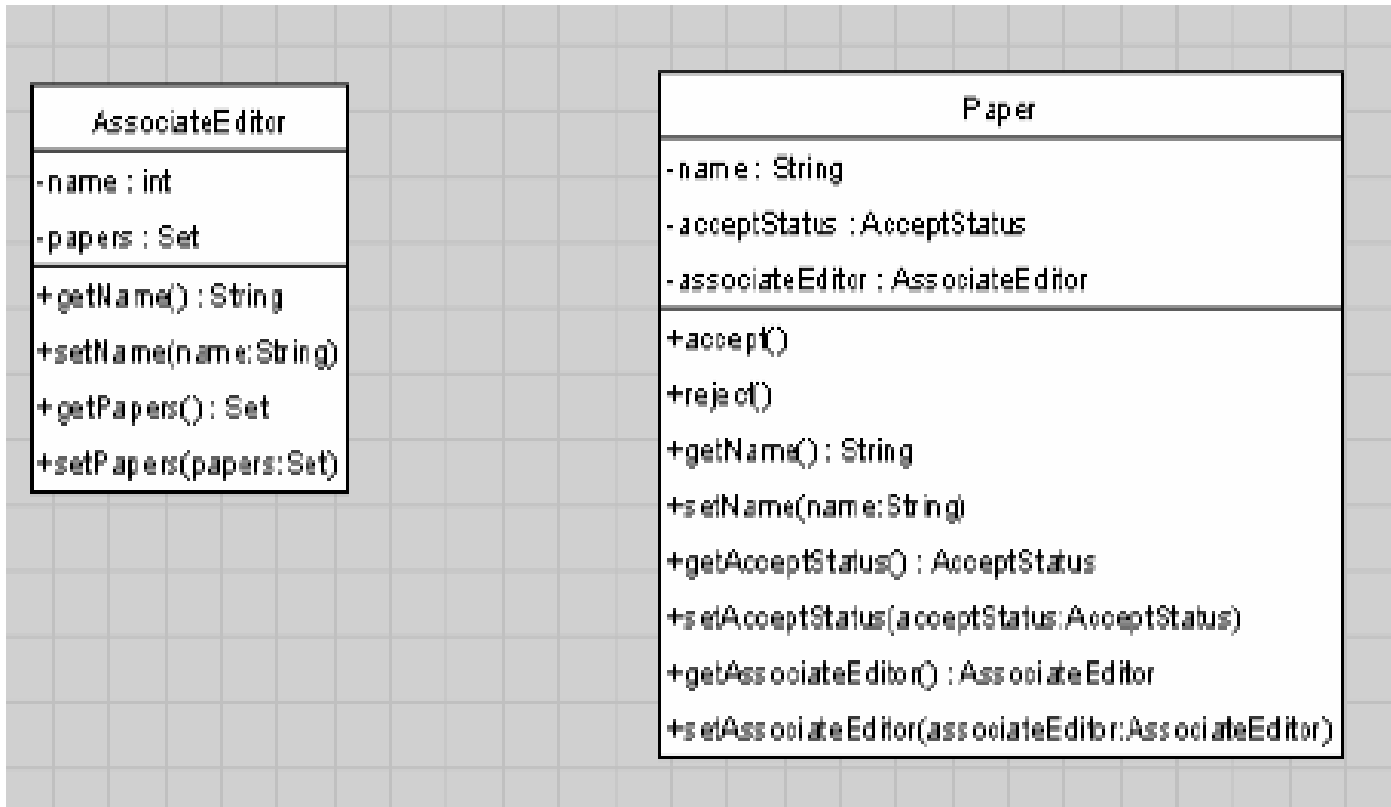


- We will transform this PIM into a PSM

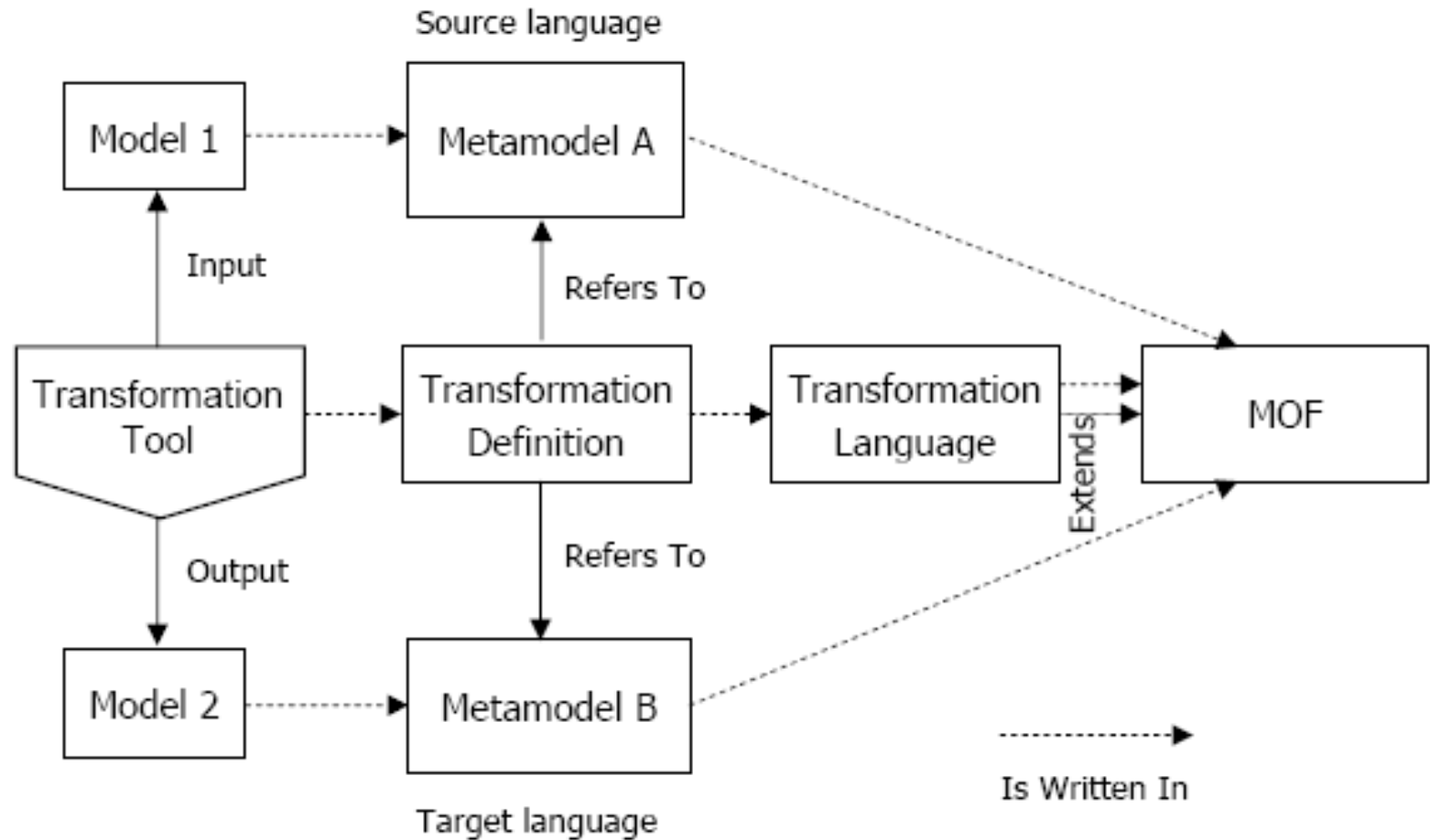
Transformation rules

- All public attributes become private
 - We must provide access methods for these attributes
- Association ends are to be modelled as new private attributes
 - We must provide access methods for these new attributes as well
- Additional requirements:
 - How do we resolve cardinality > 1 ?
 - How do we resolve n:m associations?
 - How do we resolve association classes?

The resulting PSM looks like...



General Transformation Model



Transformation Discussion

- OMG has made a RFP for a standard for defining transformations named QVT (Query-View-Transformation)
- So far 8 proposals have been made
- Take a look at Inria's ATL which is a subproject of the Eclipse GMT project (Jean Bezivin)
- Most MDA tools provide model-to-code transformations based on code templates e.g. Rose Technical Developer (formerly known as Real-Time)
- Some tools provide model-to-model transformations e.g. OptimalJ, Compuware

Agenda

- MDA Basics
- Metamodelling
- Model Transformation
- **Tools**
- Demonstration
- MDE
- Discussion

MDA Tool criteria 1

- Modelling and meta-modelling
 - UML support
 - Profile support
 - Support for creating MOF meta-models
 - Action Language definition
 - Import/export using XMI

MDA Tool criteria 2

- Model Transformation
 - Support for standard model-to-model transformations
 - Support for parameterisation and customisation of transformations
 - Support for user-defined transformations
 - Support for automatic and interactive transformations
 - Support for code, test and documentation generation
 - Synchronisation between models
 - Ability to modify the result of transformations

MDA Tool criteria 3

- Additional capabilities
 - Support for specific target platforms
 - DSL support
 - Integration of new platforms
 - Versioning and concurrent development
 - Support for reverse engineering (?)

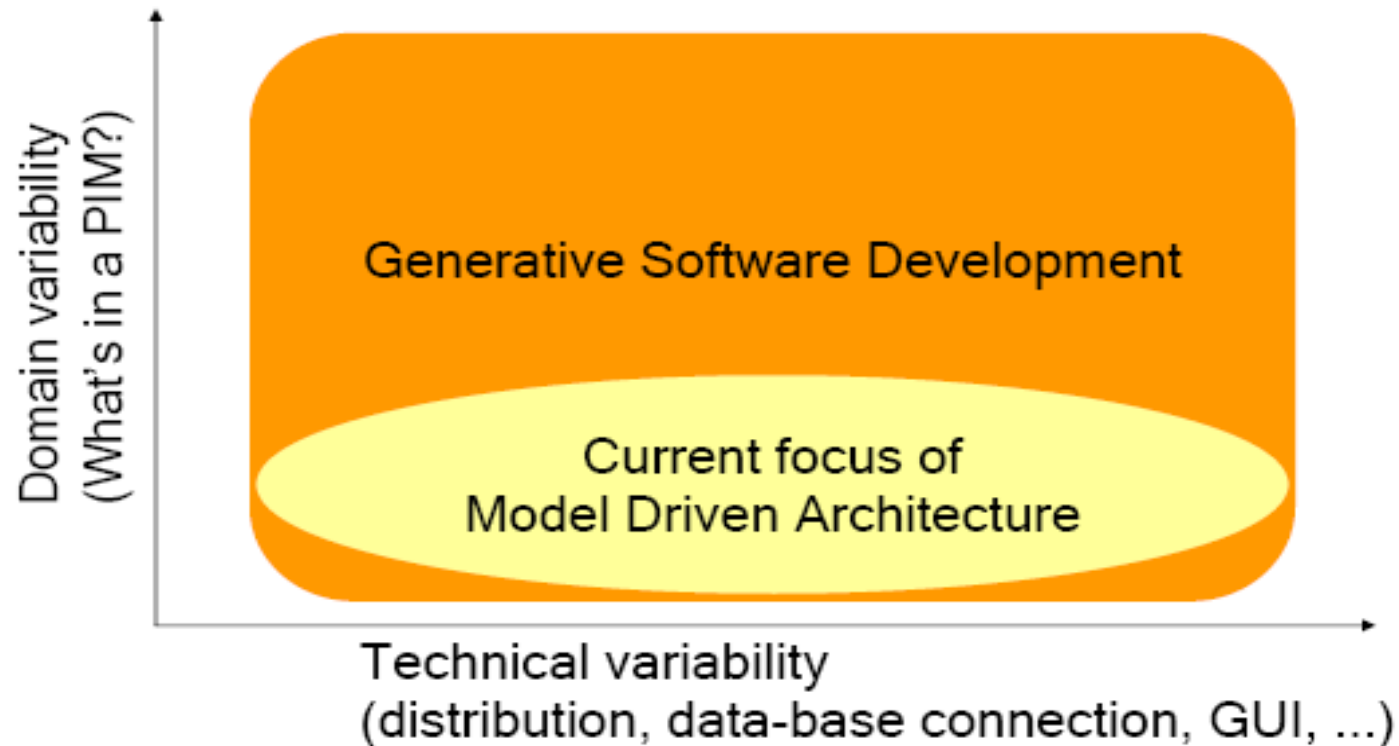
MDA Tools overview

- Following Tools have been identified
 - Commercial
 - Business Applications (J2EE): OptimalJ, Codagen Architect, ArcStyler, Rational XDE...
 - Embedded: BridgePoint, iUML, Artisan Real-Time Studio, Rhapsody, Aonix STP, Rational Technical Developer, MetaEdit+, Aonix...
 - Open Source:
 - UMT, UMLAUT / MTL, ATL, ModFact / TRL, OpenMDX, AndroMDA, Jamda, GMT,...

Additional Tools

- Generator Tool Database
www.codegeneration.org
- Other
www.modelbased.net
- MS Visual Studio 2005

Where we stand today



MDA Tools today

- Tools cover more or less the defined MDA criteria
- Use Tools to generate CRUD (create, read, update, delete) functionality
- Keep in mind: MOF and XMI promote metadata interoperability

Agenda

- MDA Basics
- Metamodelling
- Model Transformation
- Tools
- **Demonstration**
- MDE
- Discussion

Demonstration summary 1

- Tool: Rational Technical Developer (RT)
- Support for UML2
- No support for Profiles, MOF
- Action Language is C, C++ or Java
- Import/export via Rational Rose to XMI

Demonstration summary 2

- Model Transformation
 - No support for model-to-model transformations
 - Limited support for parameterisation and customisation of transformations
 - Limited support for user-defined transformations
 - No support for automatic and interactive transformations
 - Support for code, test and documentation generation
 - No synchronisation between models (one way gen.)
 - Ability to modify the result of transformations

Demonstration summary 3

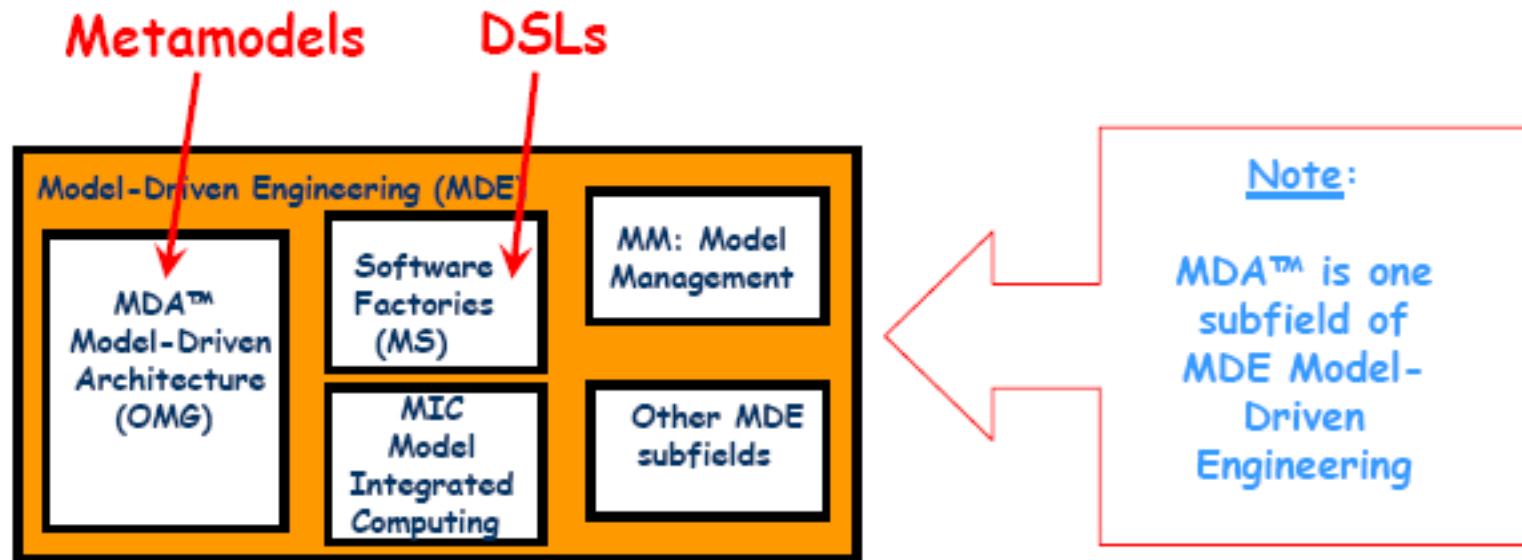
- Additional capabilities
 - Support for specific target platforms
 - No DSL support
 - Limited integration of new platforms
 - Versioning and concurrent development
 - No support for reverse engineering
- Note: Component replication needed to fully isolate models from platform specifics

Agenda

- MDA Basics
- Metamodelling
- Model Transformation
- Tools
- Demonstration
- **MDE**
- Discussion

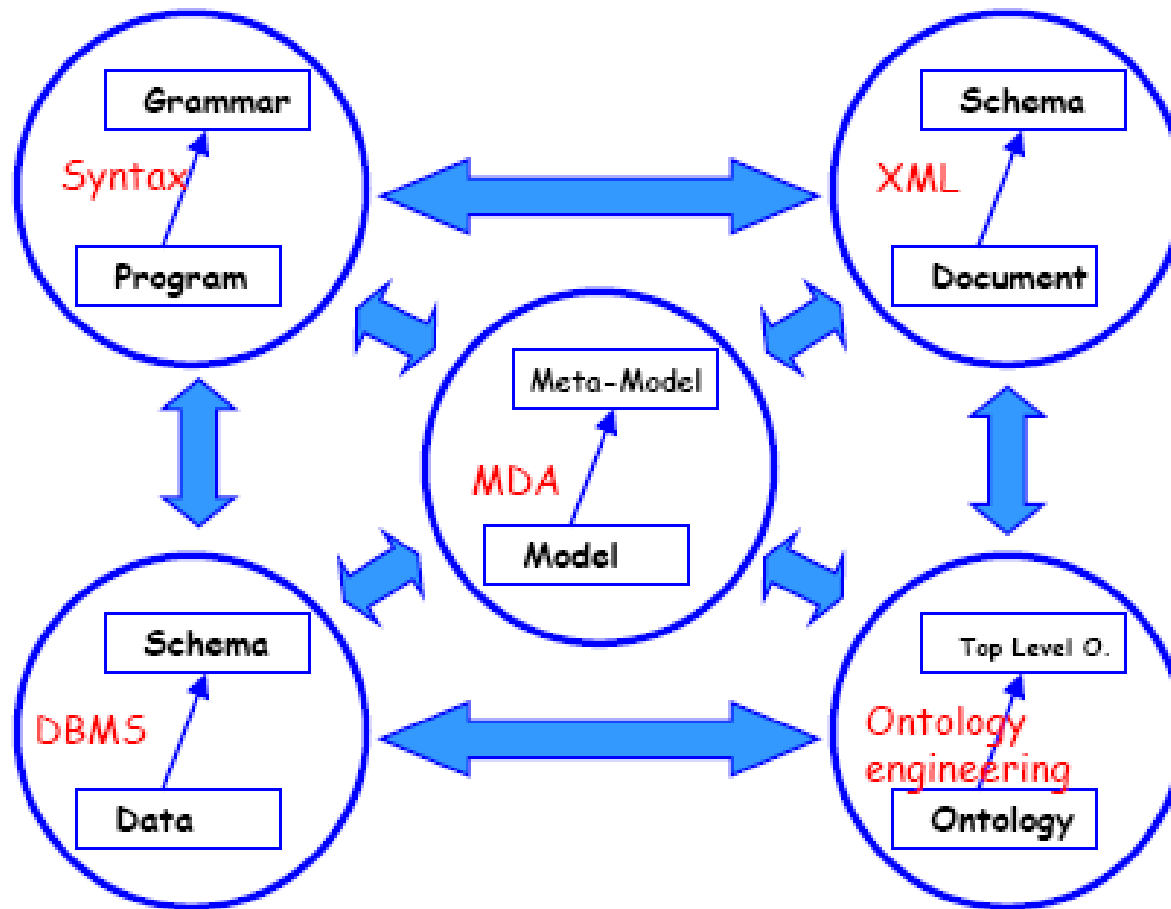
MDE

MDA™ is one specific example of MDE based on some OMG standards such as MOF, UML, CWM, QVT, etc.



- Source: Jean Bezivin

Technology spaces – Jean Bezivin



- Any TSpace is organized around an explicit or implicit "meta-meta-model"
- TSpaces are linked by bridges
- A Tspace is organized around a set of concepts
- TSpaces are similarly organized and operationally interoperable

Further reading - Books

- Frankel. "Model Driven Architecture: Applying MDA to Enterprise Computing." Wiley, 2003
- Kleppe, Warmer, & Bast. MDA Explained: The Model Driven Architecture--Practice and Promise. Addison-Wesley, 2003
- Hubert. "Convergent Architecture: Building Model Driven J2EE Systems with UML." Wiley 2001
- Grose, Doney, & Brodsky. Mastering XMI: Java Programming with XMI, XML, and UML. Wiley, 2002
- Czarnecki & Eisenecker, "Generative Programming: Methods, Tools, and Applications." Addison-Wesley, 2000
- Greenfield & Short. "Software Factories: Automating Component Design, Implementation, and Assembly." Wiley, 2004
- "*Domain Driven Development*" – *Bacvanski & Graff & Mitchell*, to be published late 2004

Further reading - Online

- MDA Guide
 - www.omg.com/mda
- DSTC MOF Pages
 - www.dstc.edu.au/Research/Projects/MOF/
- Online collection of metamodels
 - <http://mdr.netbeans.org/metamodels.html>
- Tool websites
 - www.codegeneration.org
 - www.modelbased.net/
- Open source MOF repositories
 - <http://mdr.netbeans.org>
 - <http://nsuml.sourceforge.net/>

Agenda

- MDA Basics
- Metamodelling
- Model Transformation
- Tools
- Demonstration
- MDE
- **Discussion**