Model Transformation and Weaving Formalisms and Applications

(not completely true)

Alfonso Pierantonio

Università degli Studi dell'Aquila
I-67100 L'Aquila – Italy

alfonso@di.univaq.it

dipartimentoinformatica
Università degli Studi dell'Aquila

Model Transformation and Weaving Formalisms and Applications (not completely true)

Alfonso Pierantonio

Università degli Studi dell'Aquila
I-67100 L'Aquila – Italy

alfonso@di.univaq.it

Outline

- Introduction
- Model Driven Architecture
- Model Transformations
- Application domains
  - Data-intensive web applications I, II
  - Middleware-based applications
- Conclusions

Introduction

» A well-known and long established idea

> separating the specification of the operation of a system from the details of the way that system uses the capabilities of its platform

» MDA provides an approach for, and enables tools to be provided for

> specifying a system independently of the platform that supports it

> specifying platforms

> choosing a particular platform for the system, and

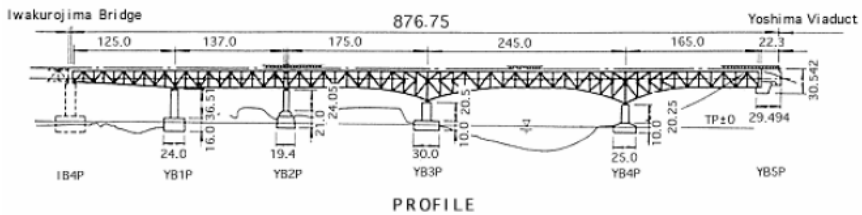
> transforming the system specification into one for a particular platform

Introduction: What is a Model?

» A model can be characterized as follows

> A model has a purpose

> A model describes some entity that exists or is intended to exist in the future

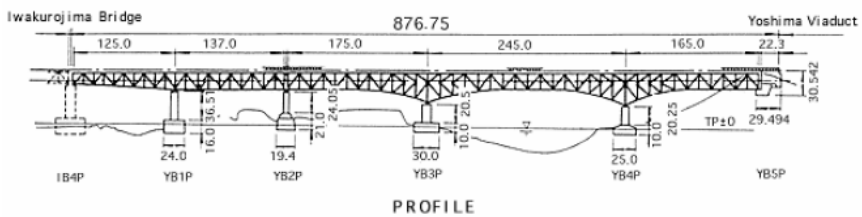


Introduction: What is a Model?

» A model can be characterized as follows

> A model has a purpose

> A model describes some entity that exists or is intended to exist in the future



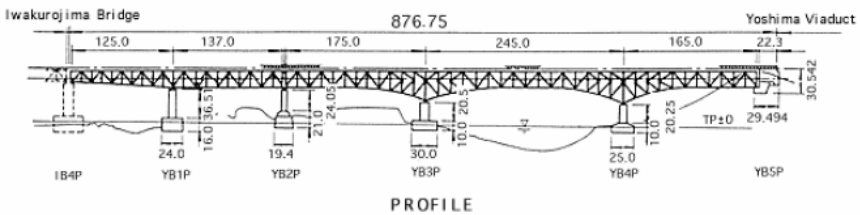
Introduction: What is a Model?

» A model can be characterized as follows

> A model has a purpose

> A model describes some entity that exists or is intended to exist in the future

> A model is an abstraction, it does not describe details of the entity that are not of interest to the audience of the model



Model Driven Architecture (MDA)

- Defined by OMG (2000) and based on modeling and automated mappings of models to implementations
- MDA is a specific Model Driven Development deployment effort around industrial standards, such as MOF, UML, QVT, etc.
- The artifacts are
  - formal models, ie. models that can be understood by computers
  - first class objects
- It separates the specification of system functionality from the specification of the implementation on a given technology platform
  - model once, generate anywhere
  - enable enterprises to preserve investments in business logic

Model Driven Architecture (MDA)

- Defined by OMG (2000) and based on modeling and automated mappings of models to implementations
- MDA is a specific Model Driven Development deployment effort around industrial standards, such as MOF, UML, QVT, etc.
- The artifacts are
  - formal models, ie. models that can be understood by computers
  - first class objects
- It separates the specification of system functionality from the specification of the implementation on a given technology platform
  - model once, generate anywhere
  - enable enterprises to preserve investments in business logic

Model Driven Architecture (MDA)

- Defined by OMG (2000) and based on modeling and automated mappings of models to implementations
- MDA is a specific Model Driven Development deployment effort around industrial standards, such as MOF, UML, QVT, etc.
- The artifacts are
  - formal models, ie. models that can be understood by computers
  - first class objects
- It separates the specification of system functionality from the specification of the implementation on a given technology platform
  - model once, generate anywhere
  - enable enterprises to preserve investments in business logic

MDA Terminology

- PIM (Platform Independent Model) is an abstract model independent from any technology
- PSM (Platform Specific Model) specifies how the functionality described in a PIM is realized on a given platform
  - A PIM can be transformed into one or more PSMs
- PIMs and PSMs are expressed in UML profiles or metamodels
- The ultimate goal is to generate the system implementation (among other views) by means of model transformations

MDA: does it make ene ?

» A PIM has to be expressed in terms of concepts and relationships among them which are familiar to those operating on that specific domain

» The domain is formalized by means of a metamodel which has to be tailored for the domain

» The more the domain is defined within crispy boundaries, the better the metamodel can be effective and induces a beneficial impact on the overall process

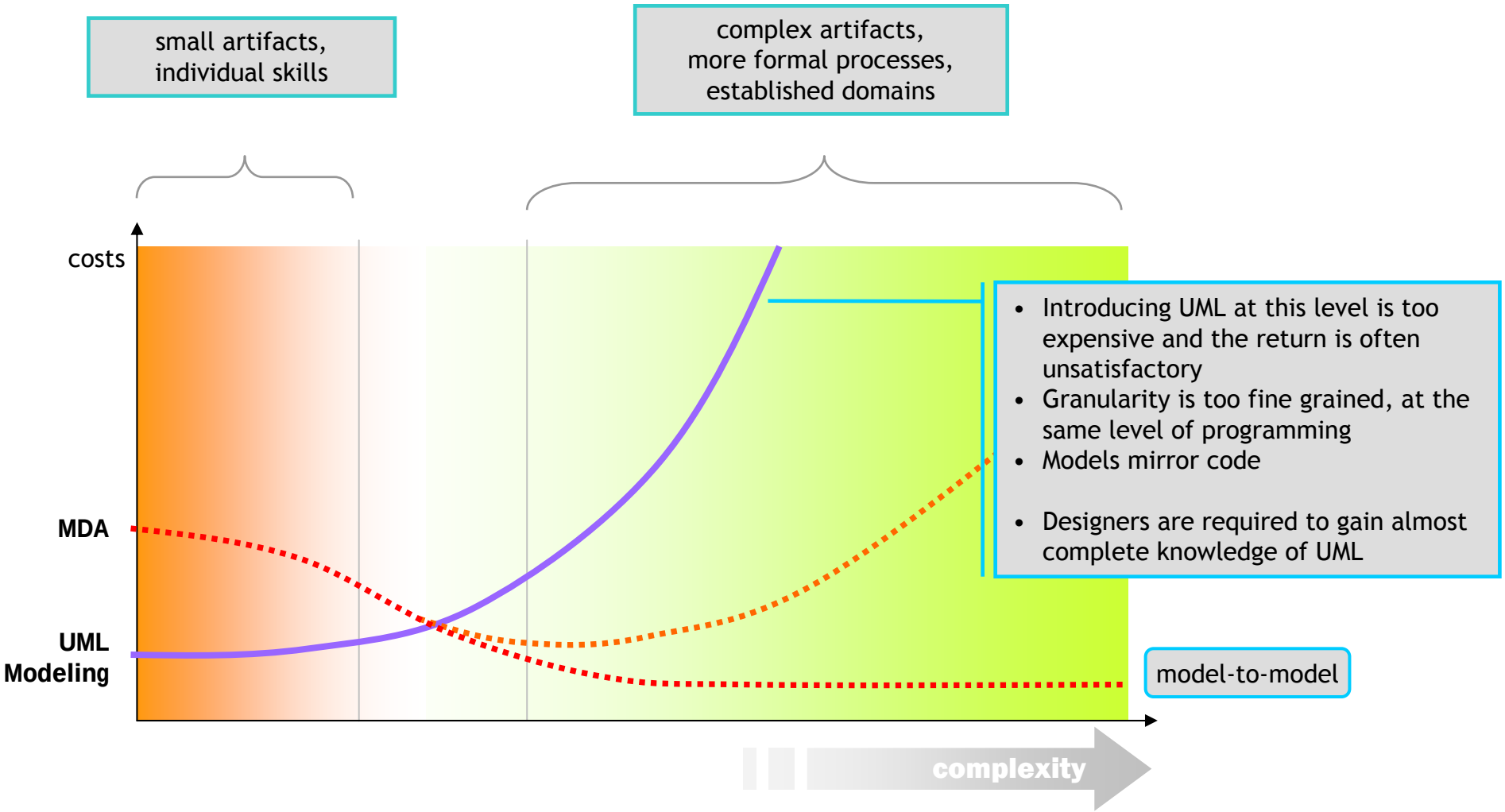
MDA: does it make ene ?

» A PIM has to be expressed in terms of concepts and relationships among them which are familiar to those operating on that specific domain

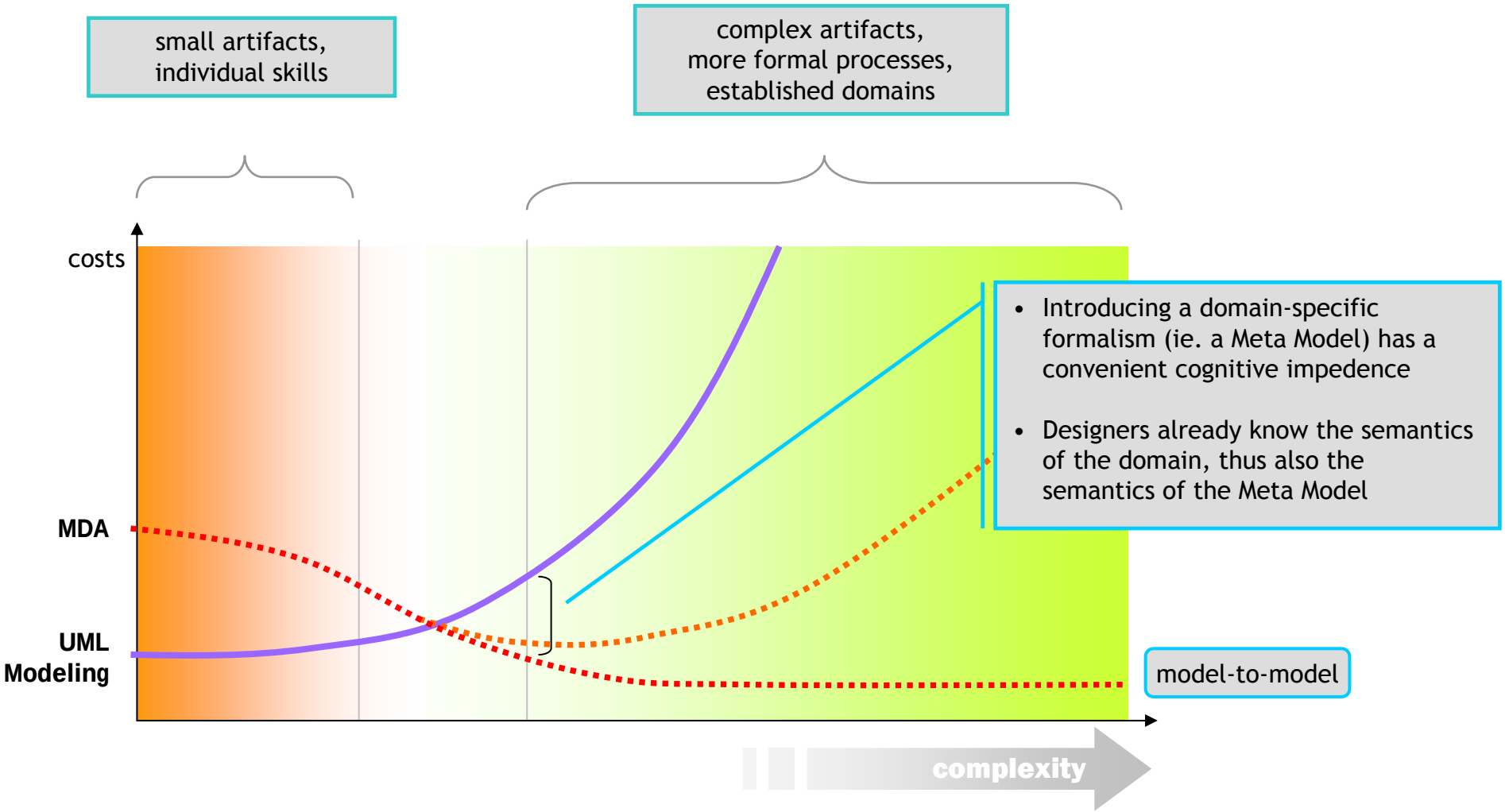
» The domain is formalized by means of a metamodel which has to be tailored for the domain

» The more the domain is defined within crispy boundaries, the better the metamodel can be effective and induces a beneficial impact on the overall process

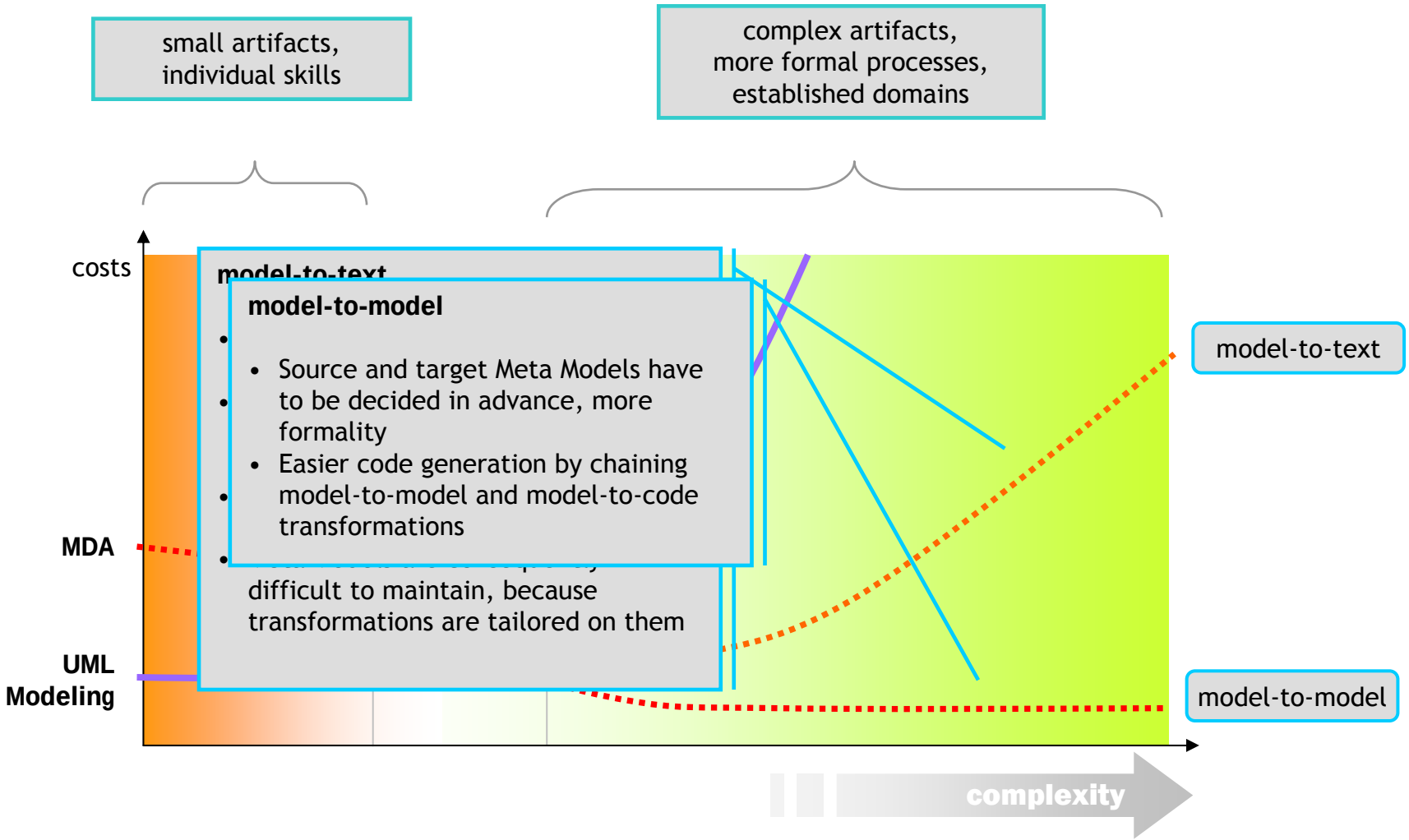
UML Modeling vs Domain-specific Modeling (MDA)



UML Modeling vs Domain-specific Modeling (MDA)



UML Modeling vs Domain-specific Modeling (MDA)



A few considerations

» Most of the approaches does not generate business rules, but just code schemata, up to 80-90% of code lines, but considerably less design efforts !

» How to improve it ?

1. Designing a specific metamodel which has to be tailored over the application domain is crucial
2. Adopting a model-to-model transformation language which offers usability and enough pragmatic qualities but also rigor
3. Composition mechanisms, it should be possible to adapt, extend, modify and combine source metamodels and consequently the transformation program

A few considerations

» Most of the approaches does not generate business rules, but just code schemata, up to 80-90% of code lines, but considerably less design efforts !

» How to improve it ?

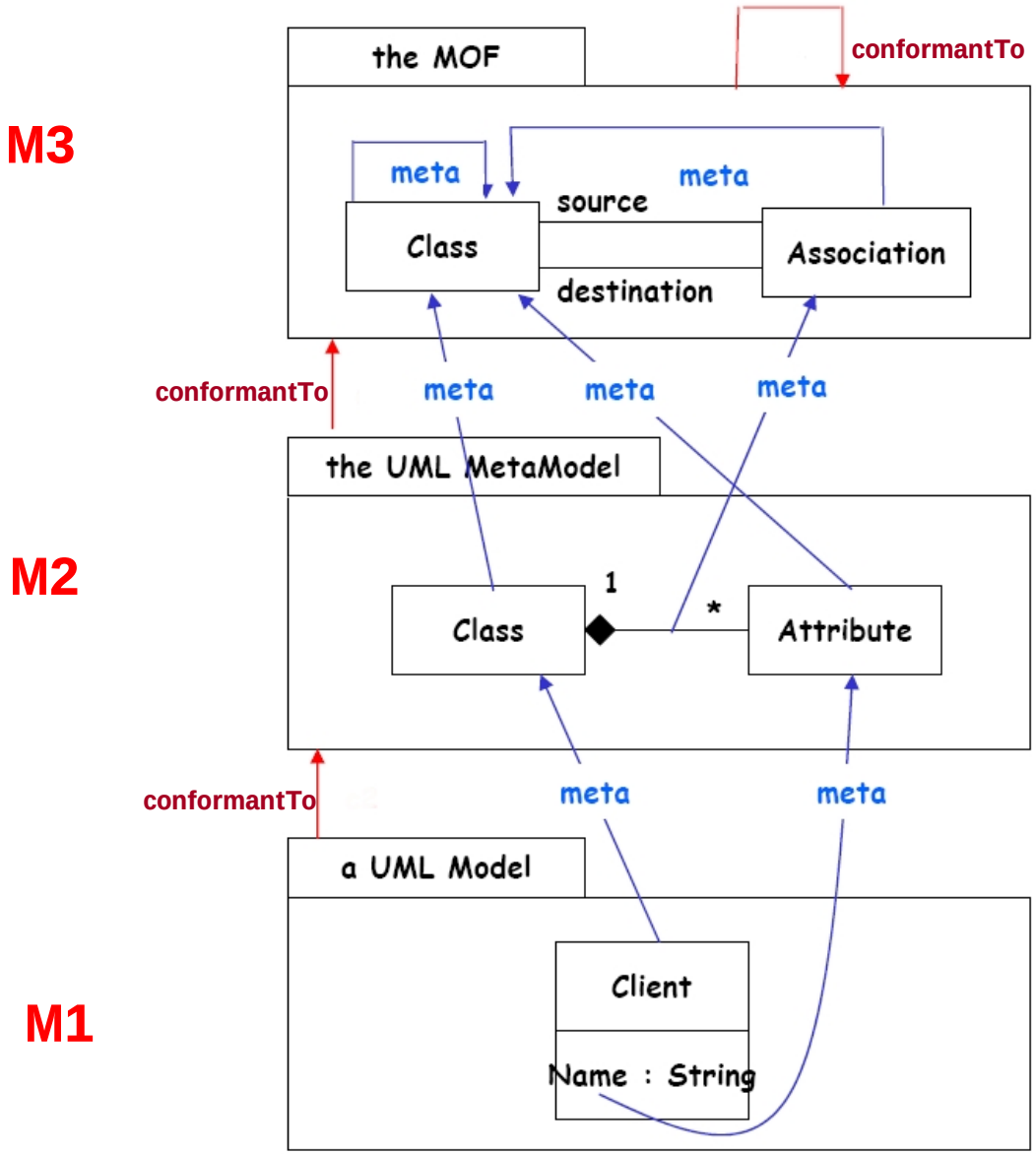
1. Designing a specific metamodel which has to be tailored over the application domain is crucial
2. Adopting a model-to-model transformation language which offers usability and enough pragmatic qualities but also rigor
3. Composition mechanisms, it should be possible to adapt, extend, modify and combine source metamodels and consequently the transformation program

MDA: Architecture with four meta-layers

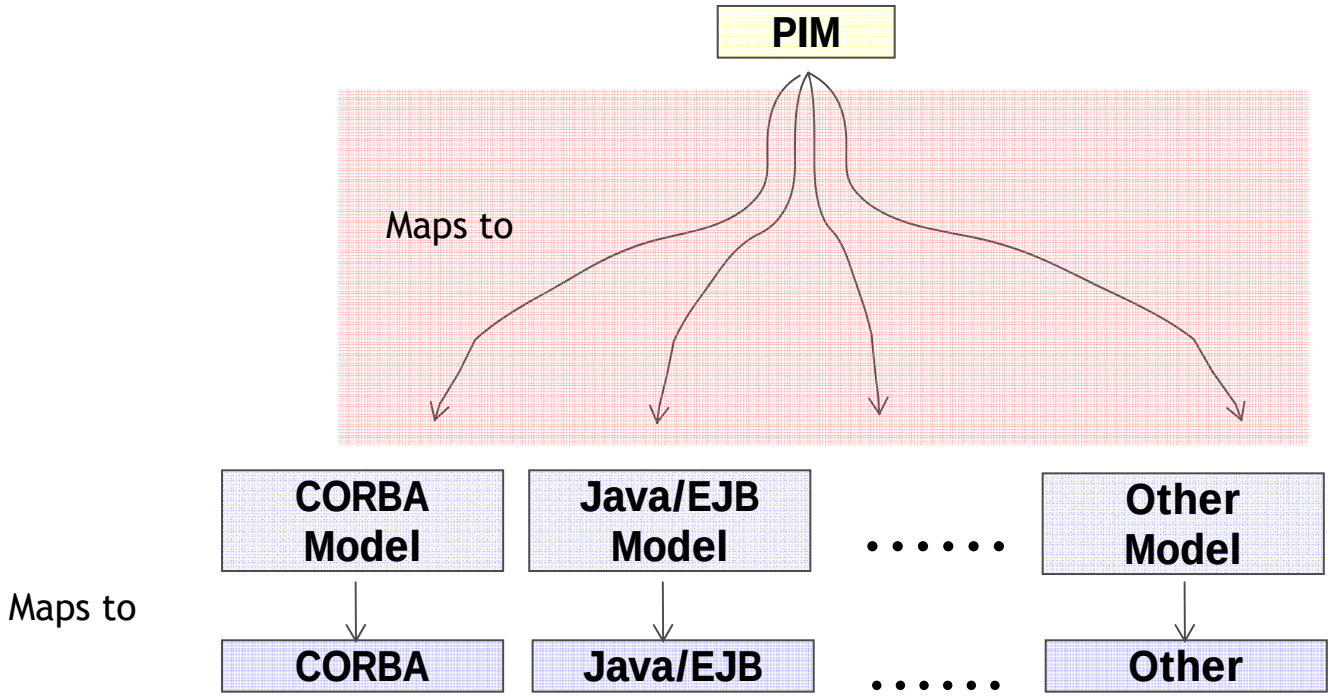
» OMG standards which provide a well-established foundation for defining PIMs and PSMs and model transformation languages as well

Meta Level	MOF terms	Examples
M3	Metametamodel	MOF models
M2	Metametadata, metamodel	UML Metamodels, UML profiles
M1	Metadata, model	UML Models (eg. Class diagrams)
M0	Data	Modeled systems

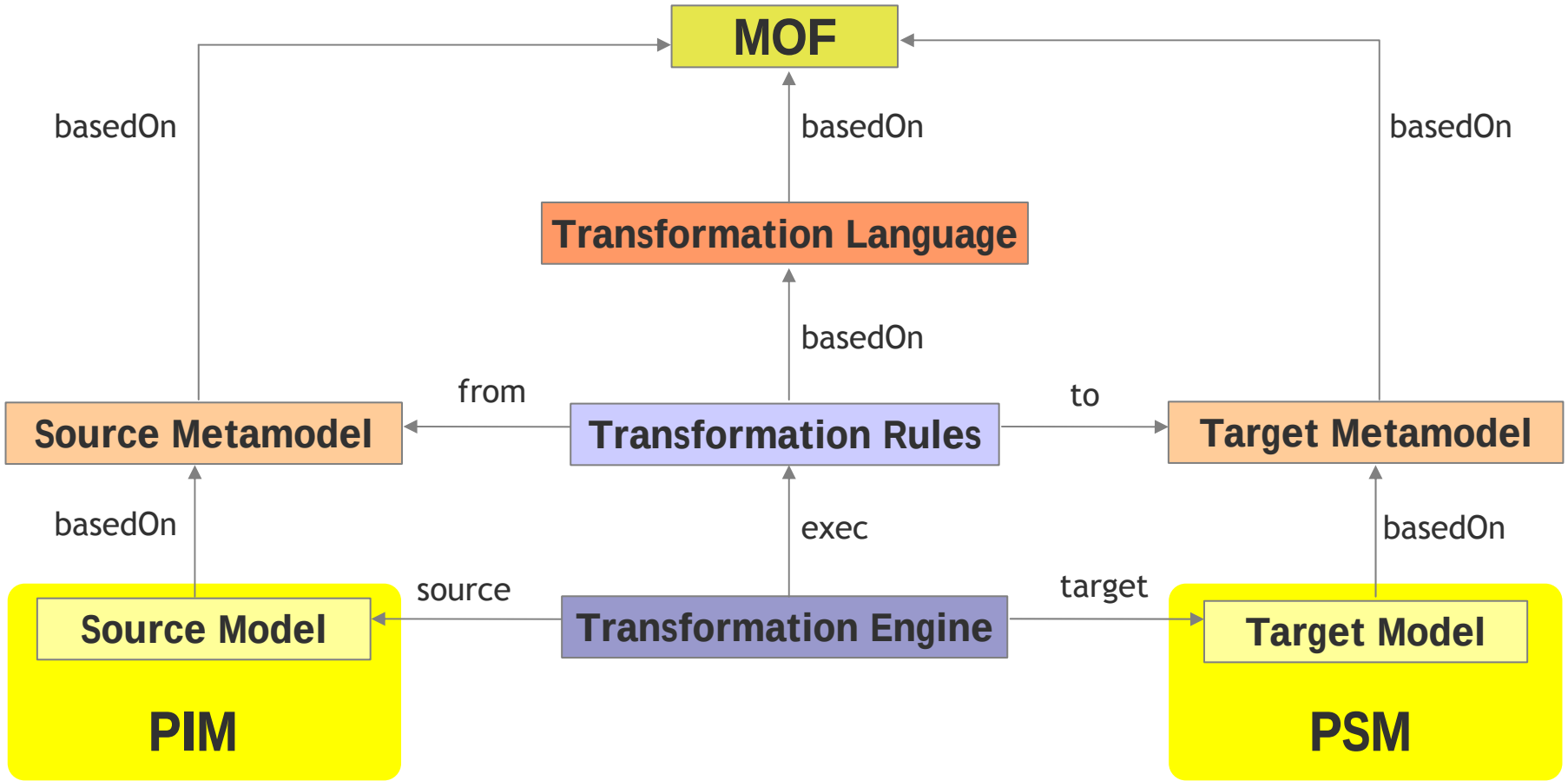
OMG Architecture example



MDA: Model Transformations



MDA: Model Transformations



Abstract State Machines (ASMs)

- ASMs tend to bridge the gap between specification and computation by providing more versatile Turing-complete machines
- Ability to simulate algorithms on their natural levels of abstraction without implementing them
- ASMs is a variant of first-order logic with equality, where the fundamental concept is that functions are defined over a set *U* and can be changed point-wise
- ASM can be given as a metamodel [Riccobene et al `04]
- Extended literature on high-level system design and analysis (see [Bőrger `03])

Abstract State Machines

» Systems of finitely many transition rules of form

if Condition **then** Updates

which transform abstract states where:

» Condition: arbitrary first-order formula without free variables

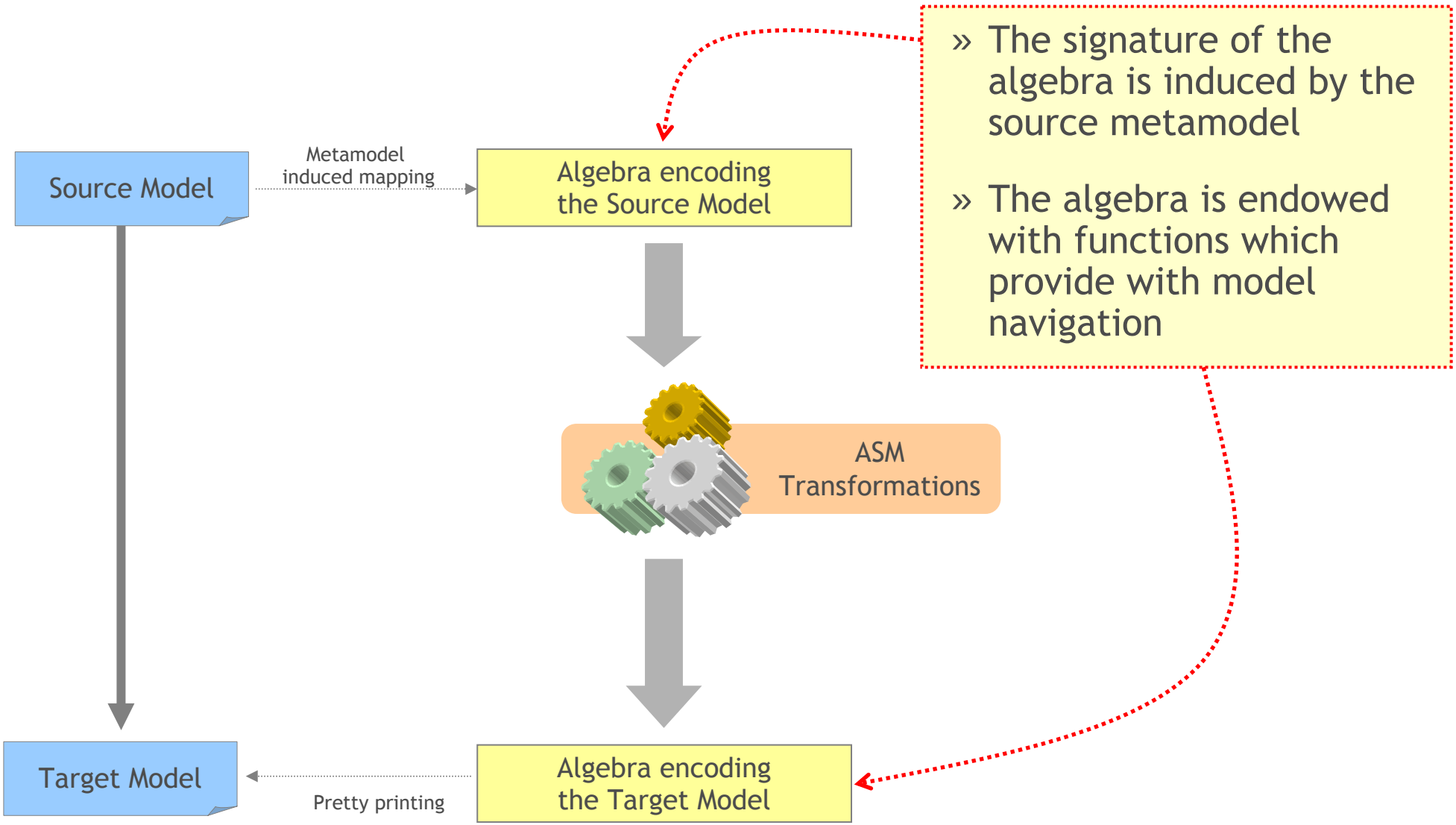
» Updates: finite set of function updates of form

f(t1,...,tn):=t

simultaneously executed when Condition is true

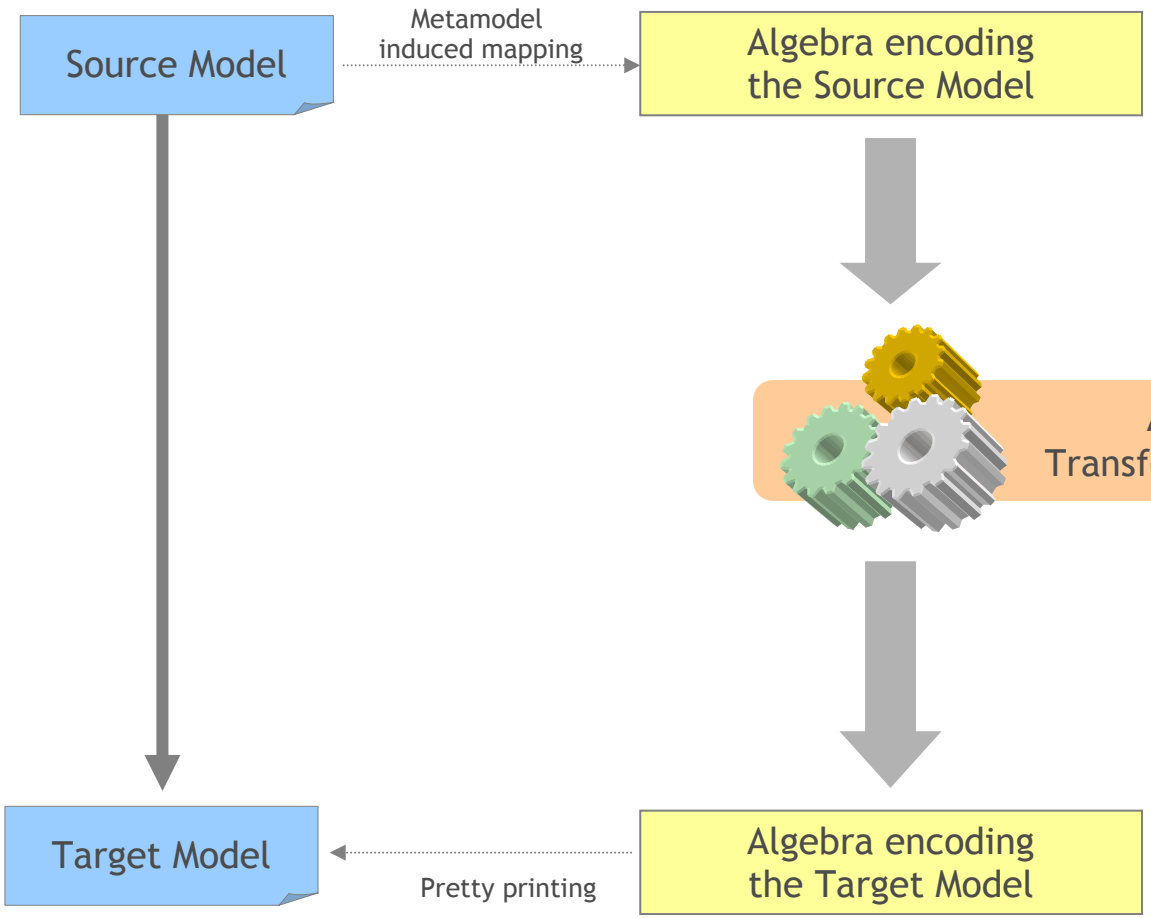
» A mathematically rigorous form to capture fundamental operational intuitions of computing

ASM and Model Transformation



- The signature of the algebra is induced by the source metamodel
- The algebra is endowed with functions which provide with model navigation

ASM and Model Transformation



Transformation process

- Target algebra initialization
- Iterative extensions of the target algebra by transforming the elements of the source algebra as specified by the transformation rules

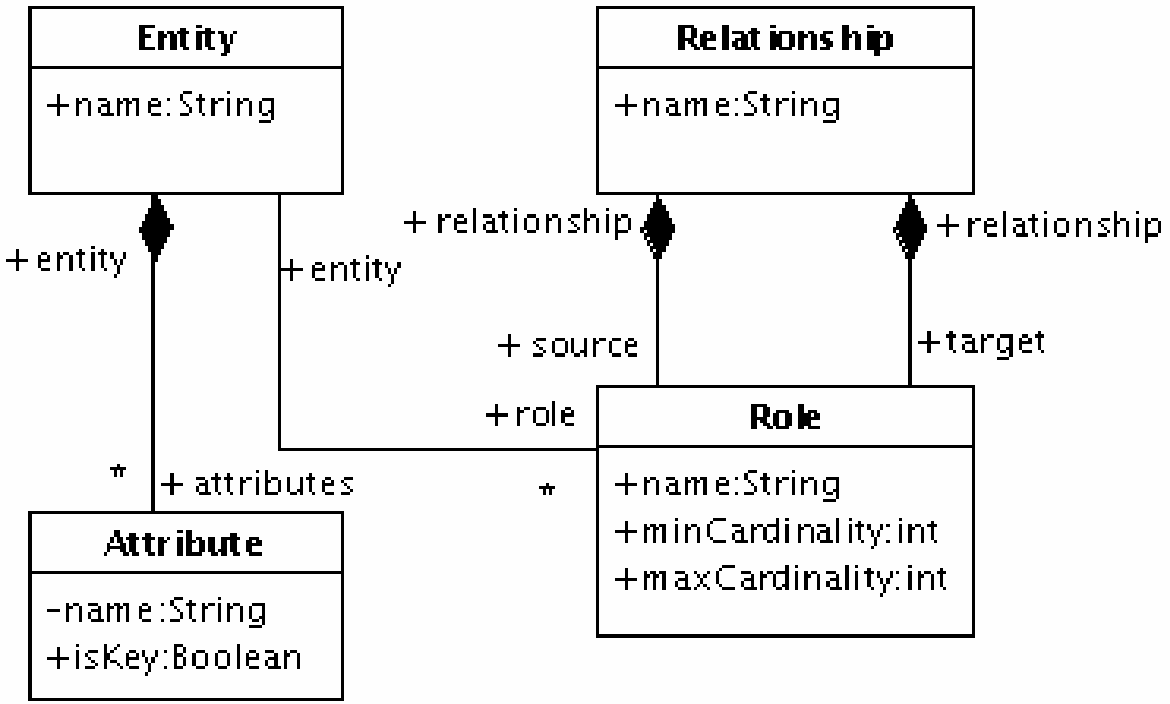
The Algebraic Encoding

» The signature of the algebras encoding a model it is canonically defined by the metamodel the models are conformant to

» The encoding can be enriched with additional information in order to accomplish more complex computations

» Example: Entity/Relationship diagrams

The Algebraic Encoding – The ER metamodel



Algebraic encoding – The induced Signature

$$\Sigma = (S, OP)$$

S := { *Entity, Relationship, Attribute, Role* }

OP :=

name : Entity → String

name : Attribute → String

entity: Attribute → Entity

name : Relationship → String

source: Relationship → Role

target: Relationship → Role

name : Role → String

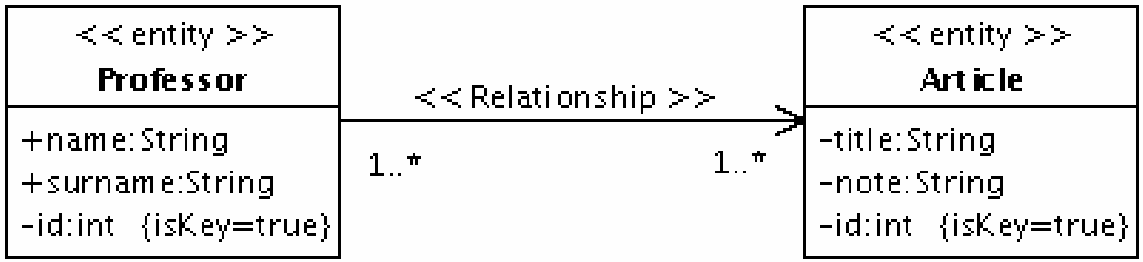
relationship: Role → Relationship

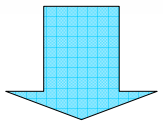
entity: Role → Entity

minCardinality: Role → Int

maxCardinality: Role → Int

Algebraic encoding - Simple ER model

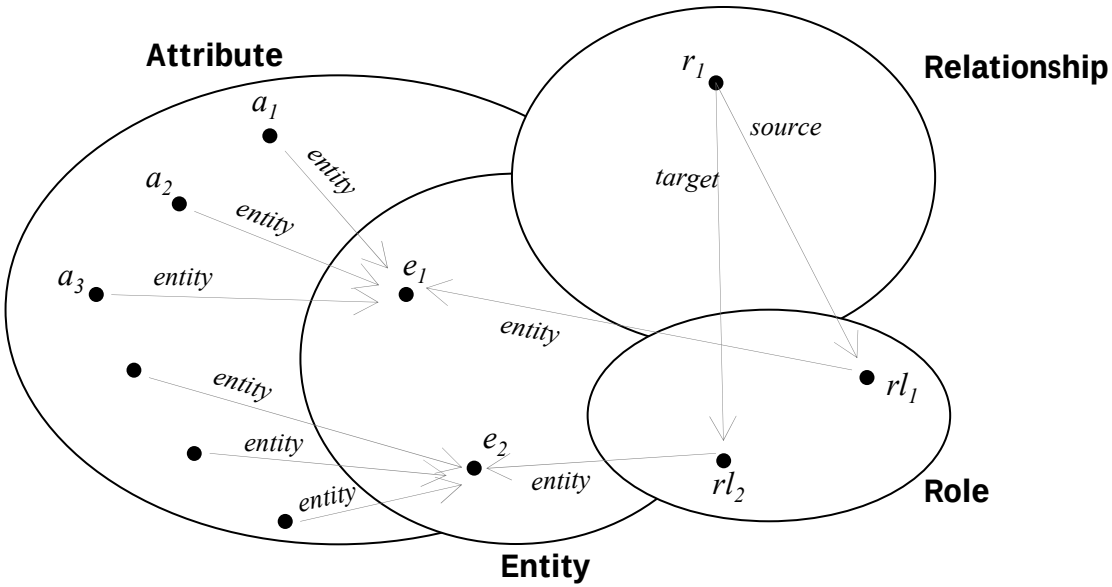




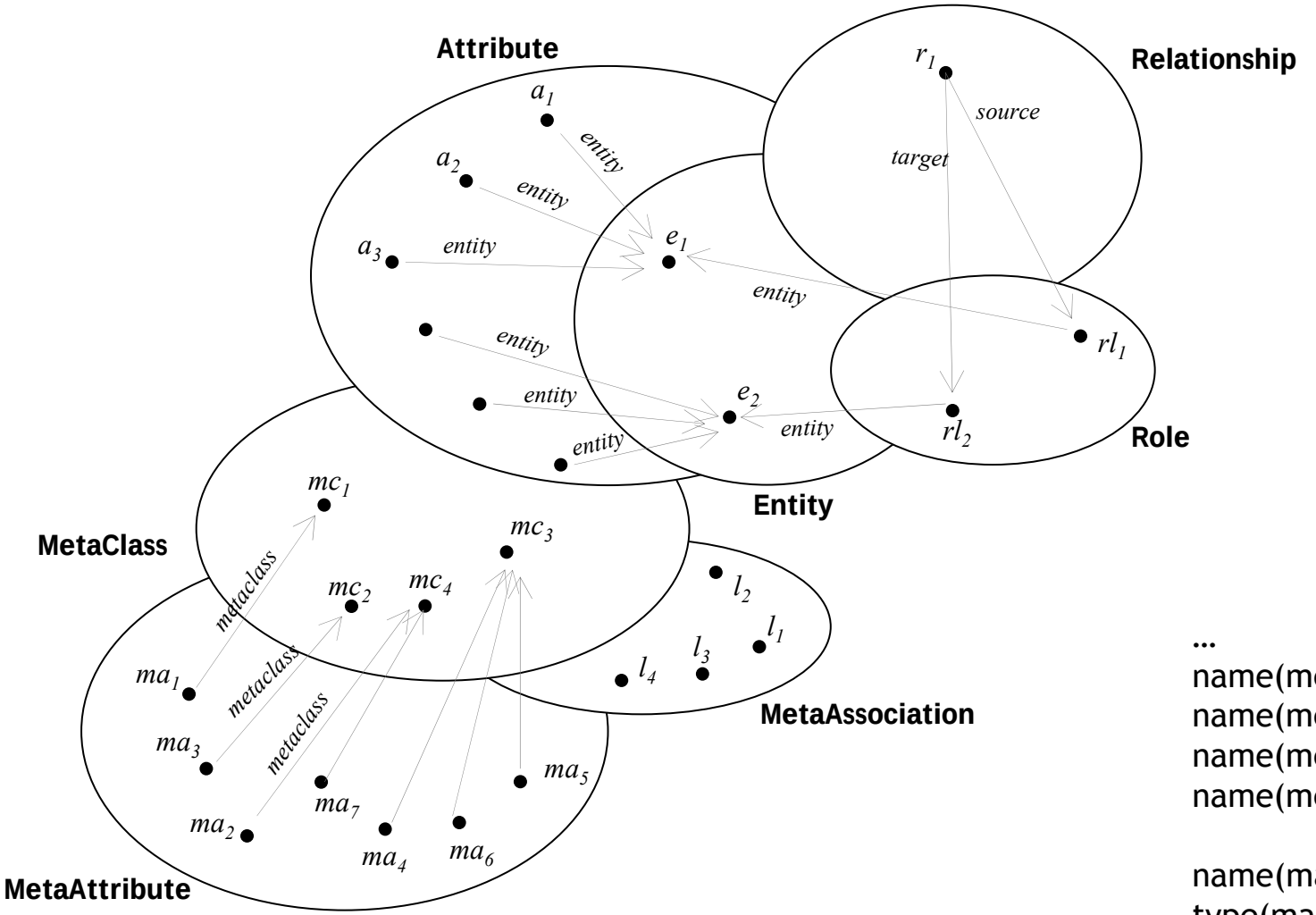
name(e_1) = “Professor”
name(e_2) = “Article”

source(r_1) = rl_1
entity(rl_1) = e_1
target(r_1)=rl_2
entity(rl_1)=e_2

name(a_1)=“name”
type(a_1)=“String”
…
name(a_3)=“id”
tyep(a_3)=“int”
isKey(id)=true
…



Enriched algebraic encoding of the ER model



…
name(mc_1) =“Entity”
name(mc_2) = “Attribute”
name(mc_3) = “Role”
name(mc_4) = “Relationship”

name(ma_1)=“name”
type(ma_1)=“string”
metaclass(ma_1)=mc_1
…

Recap

» A Meta Model induces the signature of algebras, such algebras can contain additional (auxiliary) information, eg about the Meta Model itself

» ASMs present a good combination of declarativeness and pragmatics

<Query> : <Transformations>

> queries are expressed by means of first order predicates

> transformations are procedurally defined to change the population of universes and to point-wise change functions

» Transformations may make use of asynchronous and recursive submachine performing complex computations (without side effects)

» ASMs can be used for specifying transformations and weaving operation

Recap

» A Meta Model induces the signature of algebras, such algebras can contain additional (auxiliary) information, eg about the Meta Model itself

» ASMs present a good combination of declarativeness and pragmatics

Rules = {(<Query> : <Transformations>)}

> declarativeness

> queries are expressed by means of first order predicates

> matching algorithm is implicit (no control-flow needs to be specified)

> procedurality

> transformations are procedurally defined to change the population of universes and to point-wise change functions

» Transformations may make use of asynchronous and recursive submachine performing complex computations (without side effects)

» ASMs can be used for specifying transformations and weaving operation

Domain 1

Data intensive Web Applications (I)

[CAISE05]

Data-intensive Web applications

» Data-intensive web applications are conveniently described as a hybrid between hypermedia and an information system

» Problem dimensions

> Data management

> Content delivery

> Navigation

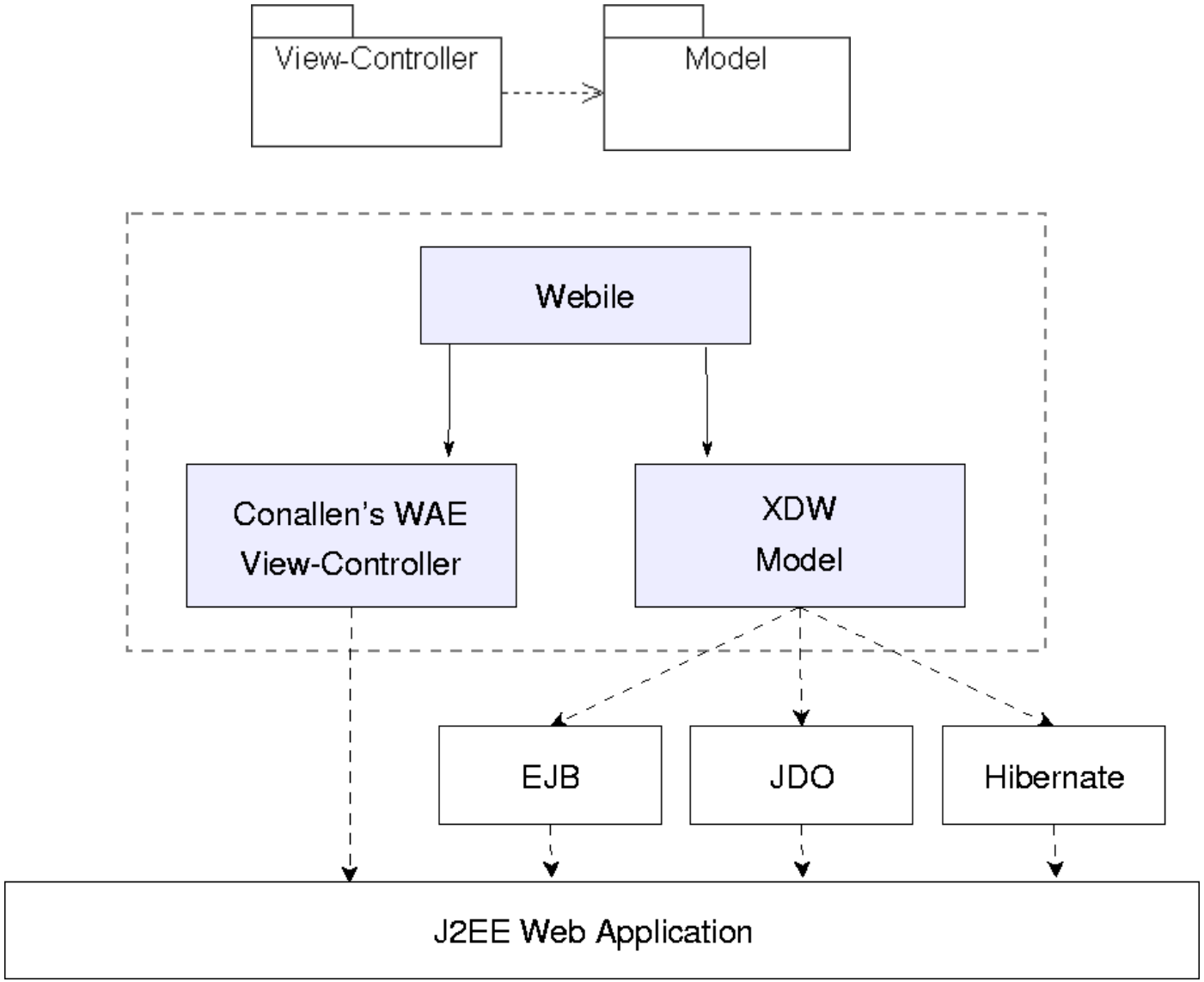
» We are interested in having conceptual descriptions of Web applications which do not refer to any technological asset

Model-View-Controller

It is an architectural pattern which aims at minimizing the degree of coupling between elements to relate the user interface to underlying data models

- Model, holds all data relevant to domain entity or process and performs behavioral processing on that data
- View, displays data contained in the model and maintains consistency in the presentation when the model changes
- Controller, is the glue between view and model reacting to significant events in the view, which may result in manipulation of the model

Problem Architecture



Source profile: Webile

» Webile is a UML profile to describe in a uniform and conceptual way proper aspects of data-intensive Web applications [IJWET04]

» Descriptions encompasses several concerns by capturing data, pages and navigation into extended class diagrams

» Webile can be used to specify PIMs for data-intensive Web applications

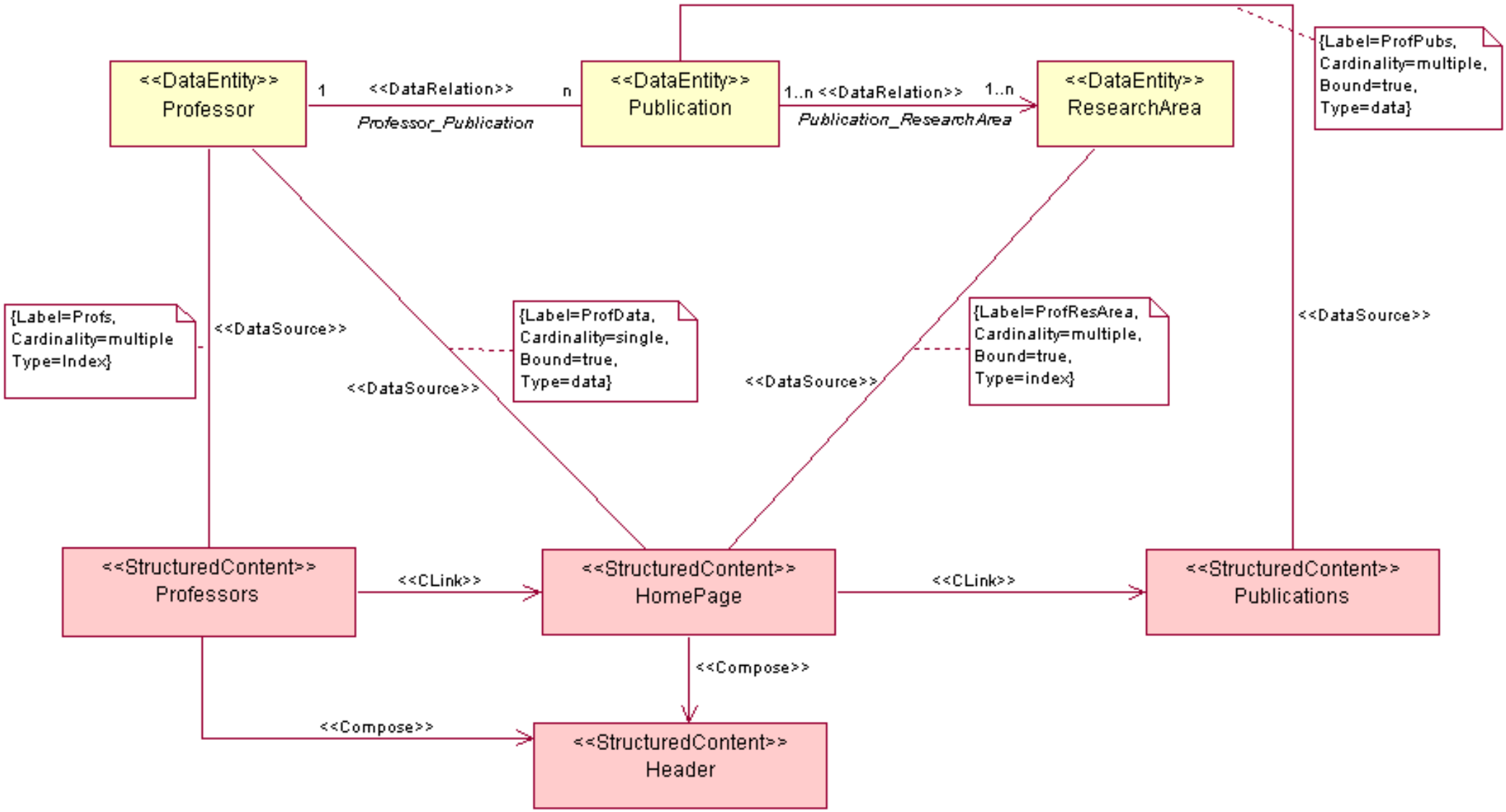
Class stereotypes

> Data Entity
> Structured Content
> Auth Structured Content
> External Content
> …

Association Stereotypes

> Data Relation
> Contextual Link
> Non Contextual Link
> Data Source
> Compose
> …

Source profile: a PIM



Target profile: Conallen's WAE

» The Web Application Extension (WAE) is an extension of UML for modeling Web applications proposed by J.Conallen

» WAE intends a web application as systems (web servers, networks, HTTP, browsers, etc) where user input affects the state of the business

» The idea of using Conallen's WAE for specifying PSMs is not novel (eg. see [Bezevin '03] and [Vallecillo '04])

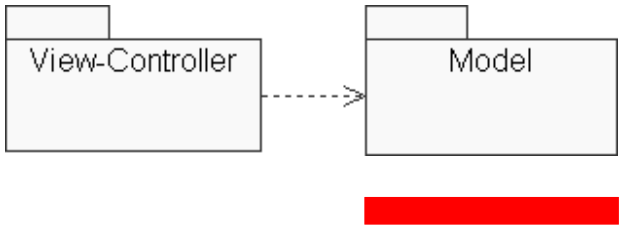
Class stereotypes

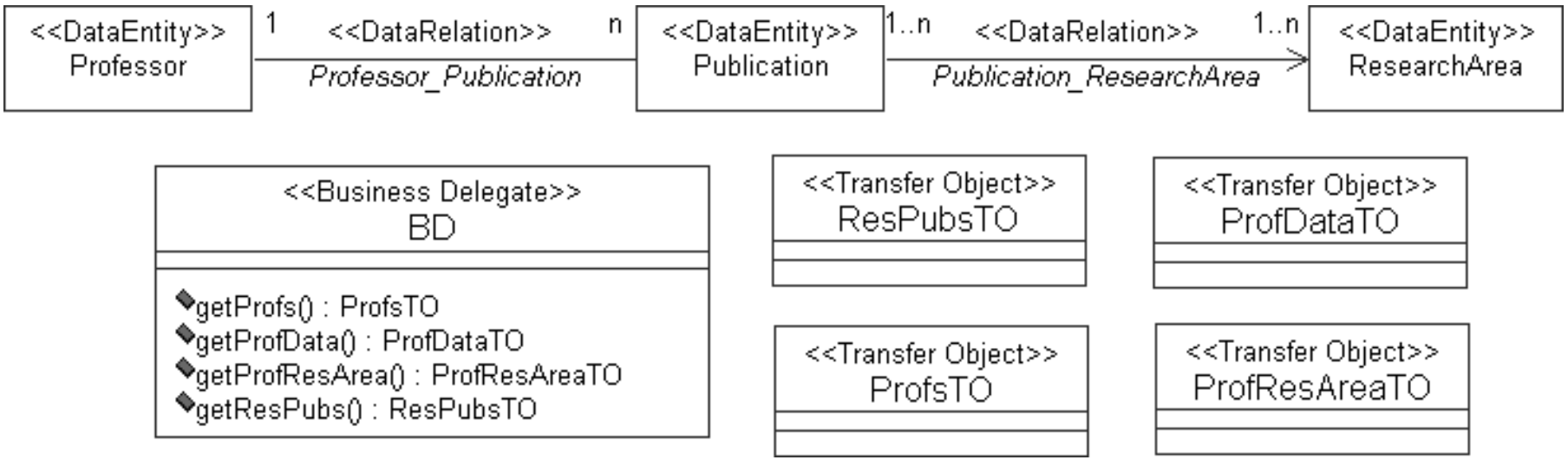
> Server Page
> Client Page
> Form
> Frameset, Target, etc.

Association Stereotypes

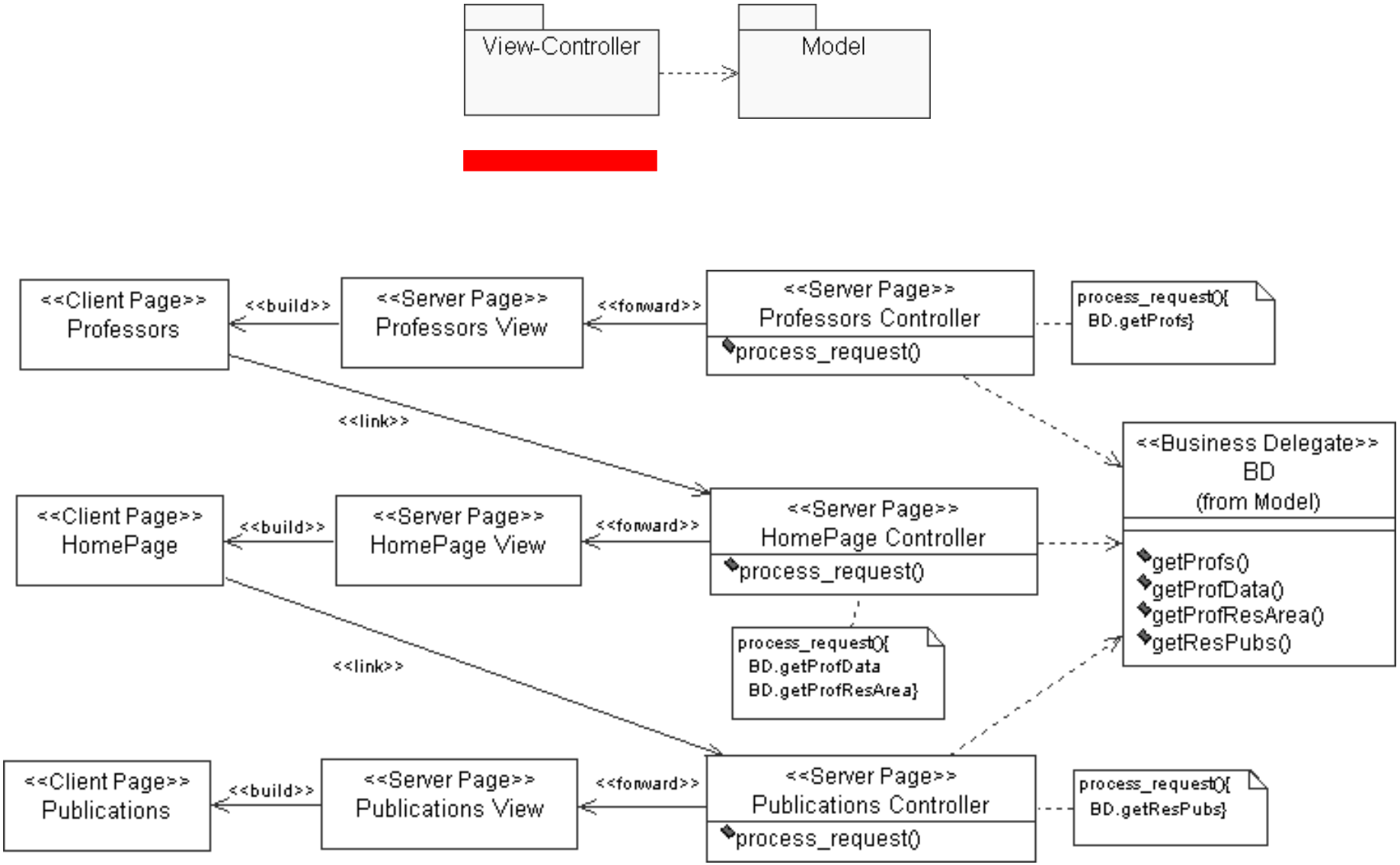
> Link
> Builds
> Frame Content, Submit, etc.

Target profile: the "model" PSM

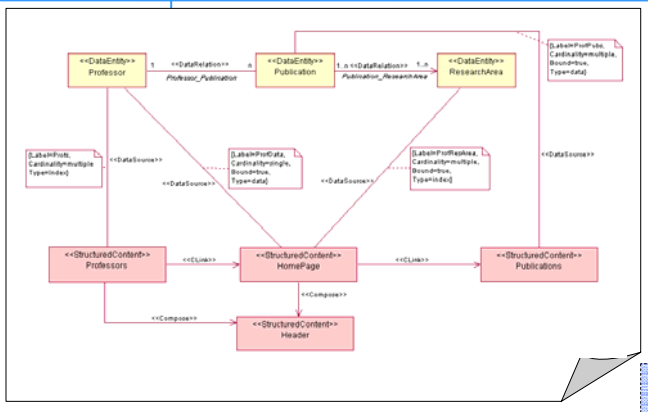


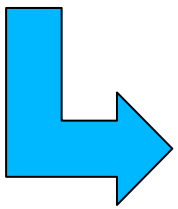


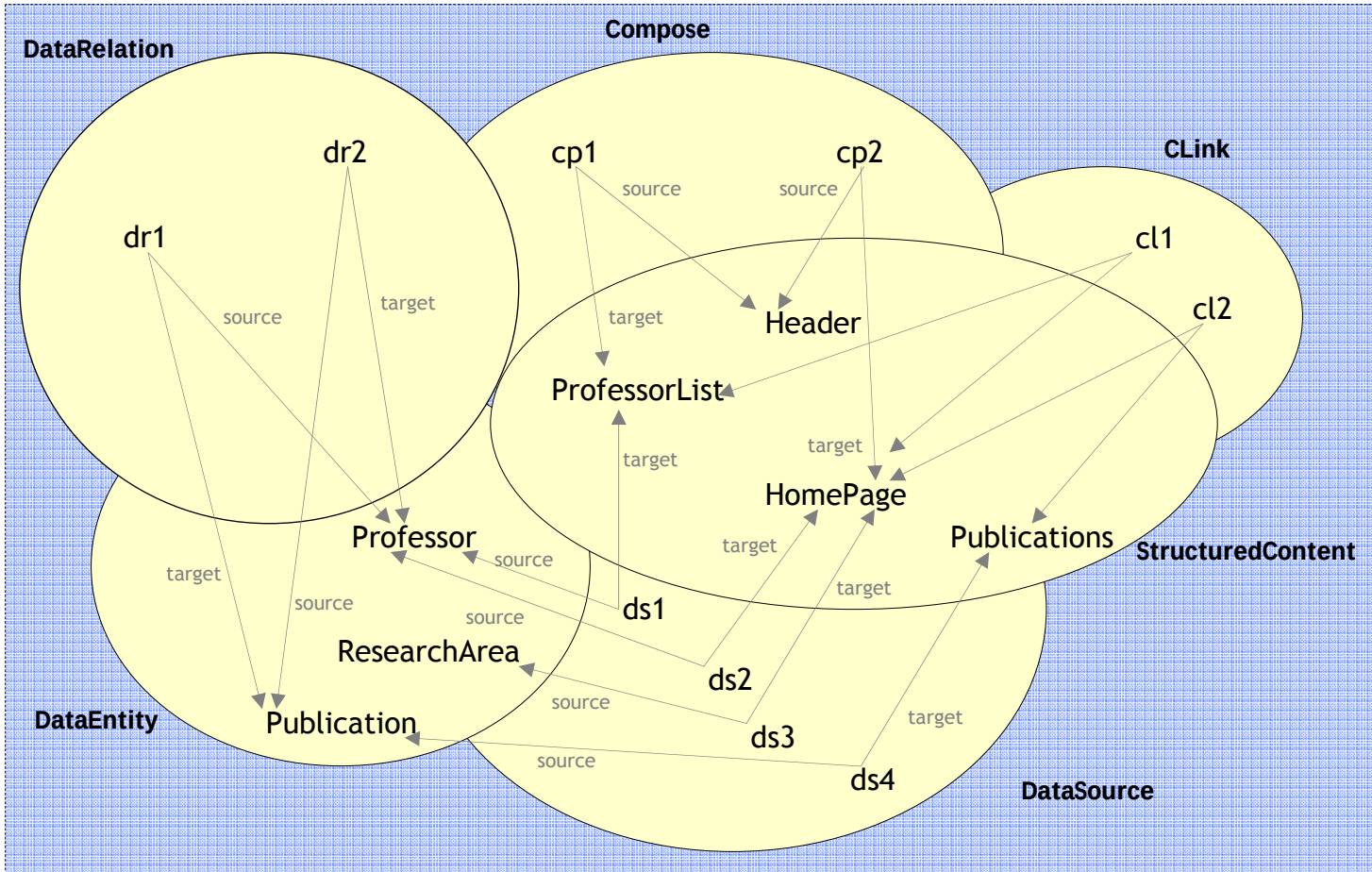
Target profile: the "view-controller" PSM



Source Model: canonical encoding







An ASM rule (1)

```
asm StructuredContent is
  do forall x in StructuredContent
    extend ServerPage with s1,s2 and ClientPage with c and Build with b and
           Forward with r and Use with u
      source(b) := s1
      target(b) := c
      source(r) := s2
      target(r) := s1
      source(u) := s2
      target(u) := bd
      controller(x) := s2
      serverView(x) := s1
      clientView(x) := c

      generatedFrom({s1,s2,c,b,r}) = {x}

    endextend
  enddo
endasm
```

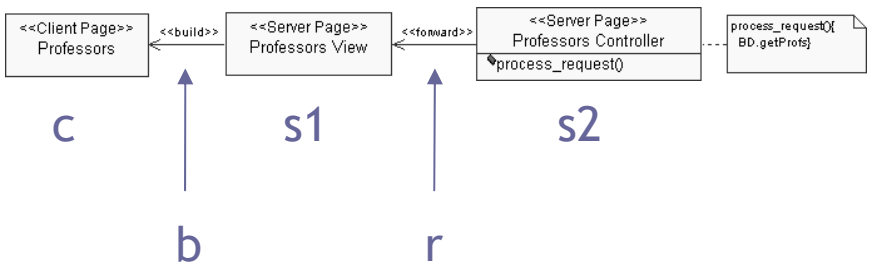
An ASM rule (3)

```
asm StructuredContent is
  do forall x in StructuredContent
    extend ServerPage with s1,s2 and ClientPage with c and Build with b and
           Forward with r and Use with u
      source(b) := s1
      target(b) := c
      source(r) := s2
      target(r) := s1
      source(u) := s2
      target(u) := bd
      controller(x) := s2
      serverView(x) := s1
      clientView(x) := c

      generatedFrom({s1,s2,c,b,r}) = {x}

    endextend
  enddo
endasm
```

For each StructuredContent in the source algebra, creates in the target algebra representatives for the following



An ASM rule (4)

```
asm StructuredContent is
  do forall x in StructuredContent
    extend ServerPage with s1,s2 and
           Forward with r and Use wi
      source(b) := s1
      target(b) := c
      source(r) := s2
      target(r) := s1
      source(u) := s2
      target(u) := bd
      controller(x) := s2
      serverView(x) := s1
      clientView(x) := c

      generatedFrom({s1,s2,c,b,r}) = {x}

    endextend
  enddo
endasm
```

Persistent transformation

Explicit tracking information for change propagation, usually implicit.

Domain 1

Data intensive Web Applications (II)

[SAC06]

Design methodologies of Web applications (1)

» Many design methodologies have been proposed to cope with the technical intricacy of Web applications

» They share a common metamodel enabling the description of such systems under three different views:

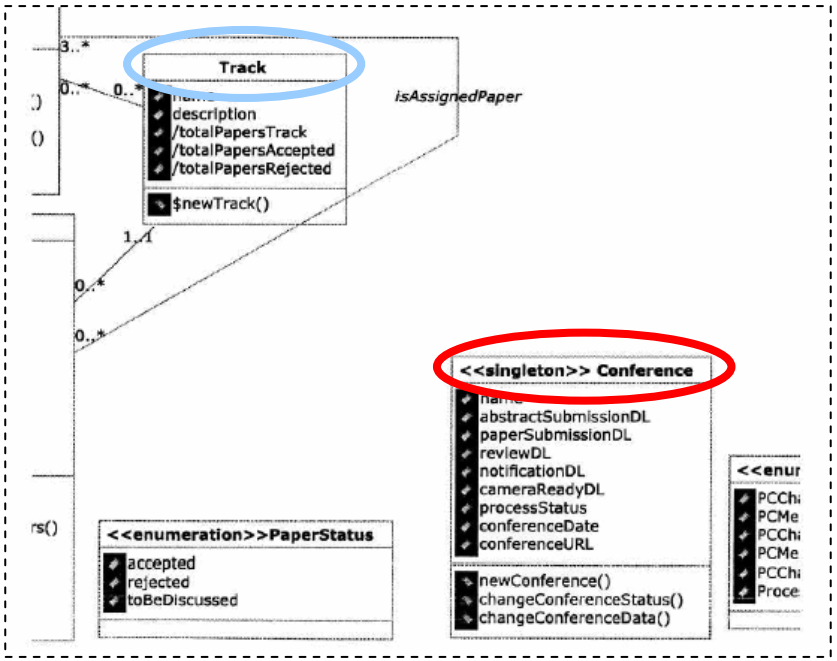
> data,

> navigation, and

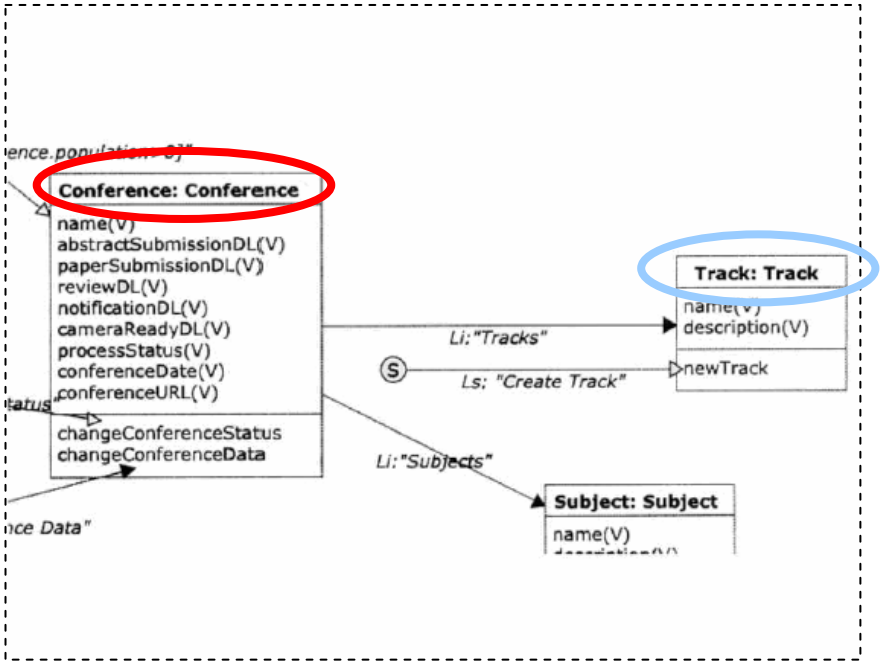
> presentation

» Consistency among the views is guaranteed by less formal relations being essentially based on name conventions and/or ad-hoc tool support

Design methodologies of Web applications (2)



Conceptual data model



Navigation model

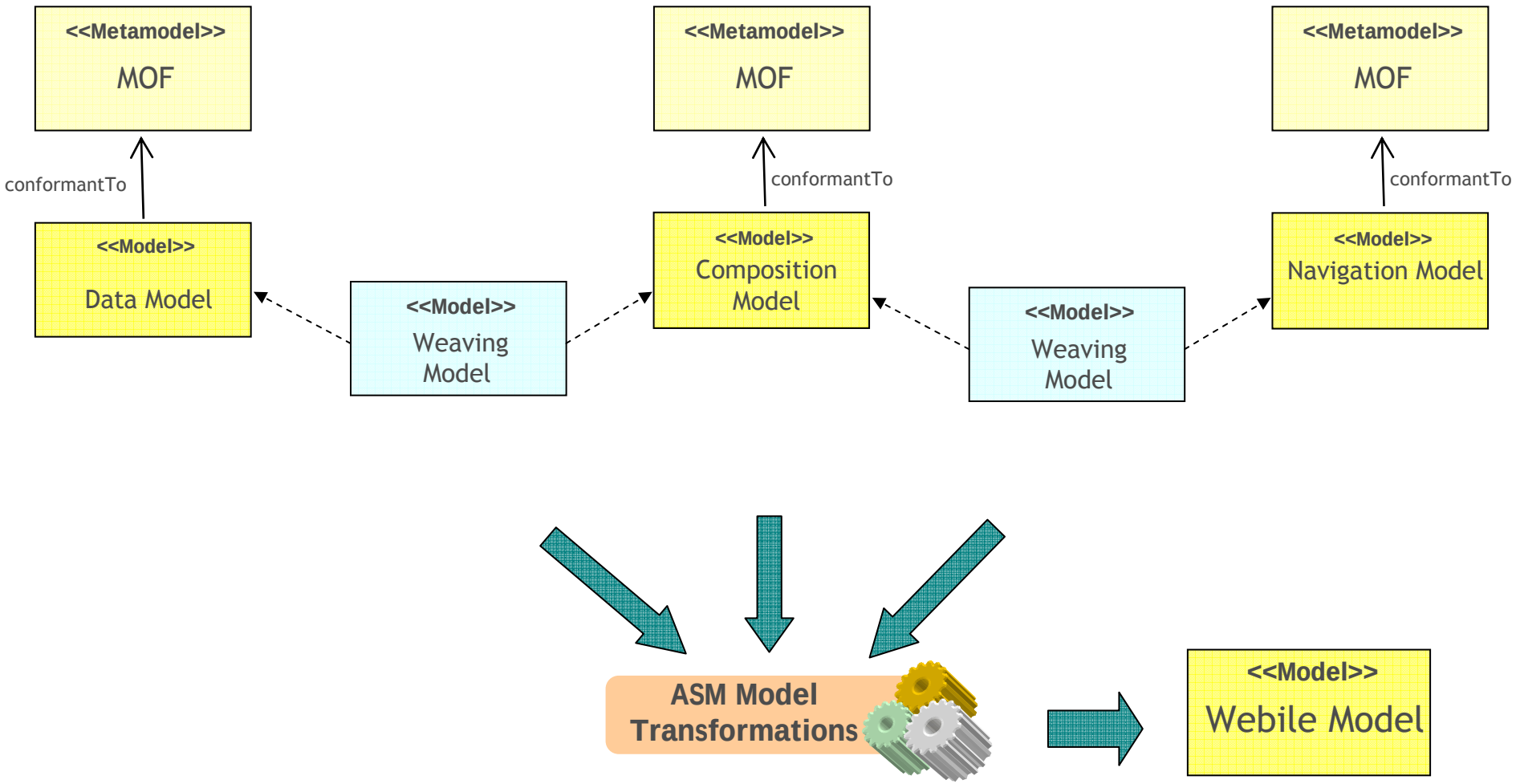
Weaving Operation

» Typically exploited for database metadata integration and evolution [Bernstein`03]

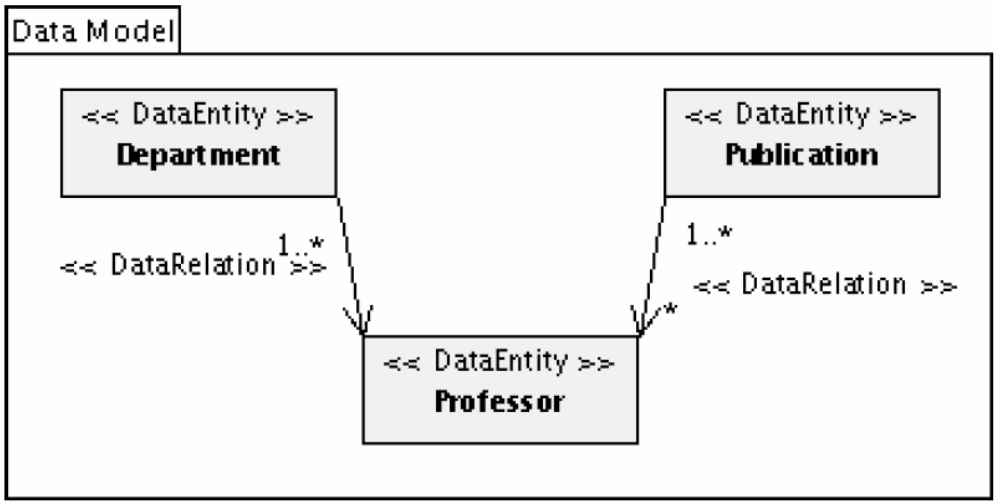
» It can be used for setting fine-grained relationships between models or metamodels and executing operations according to the link semantics [Bézivin`04]

» Weaving models can be used to define rigorous connections between the different artifacts achieving a clear separation of views and their connections

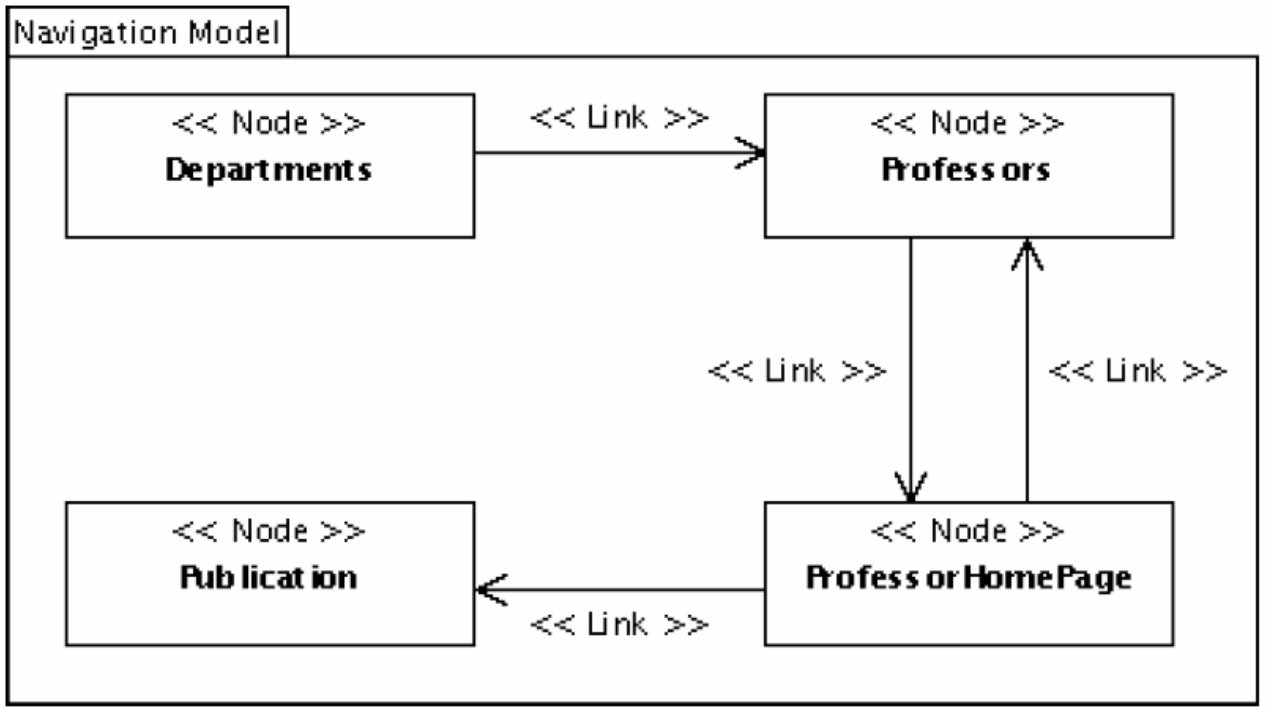
Proposed approach



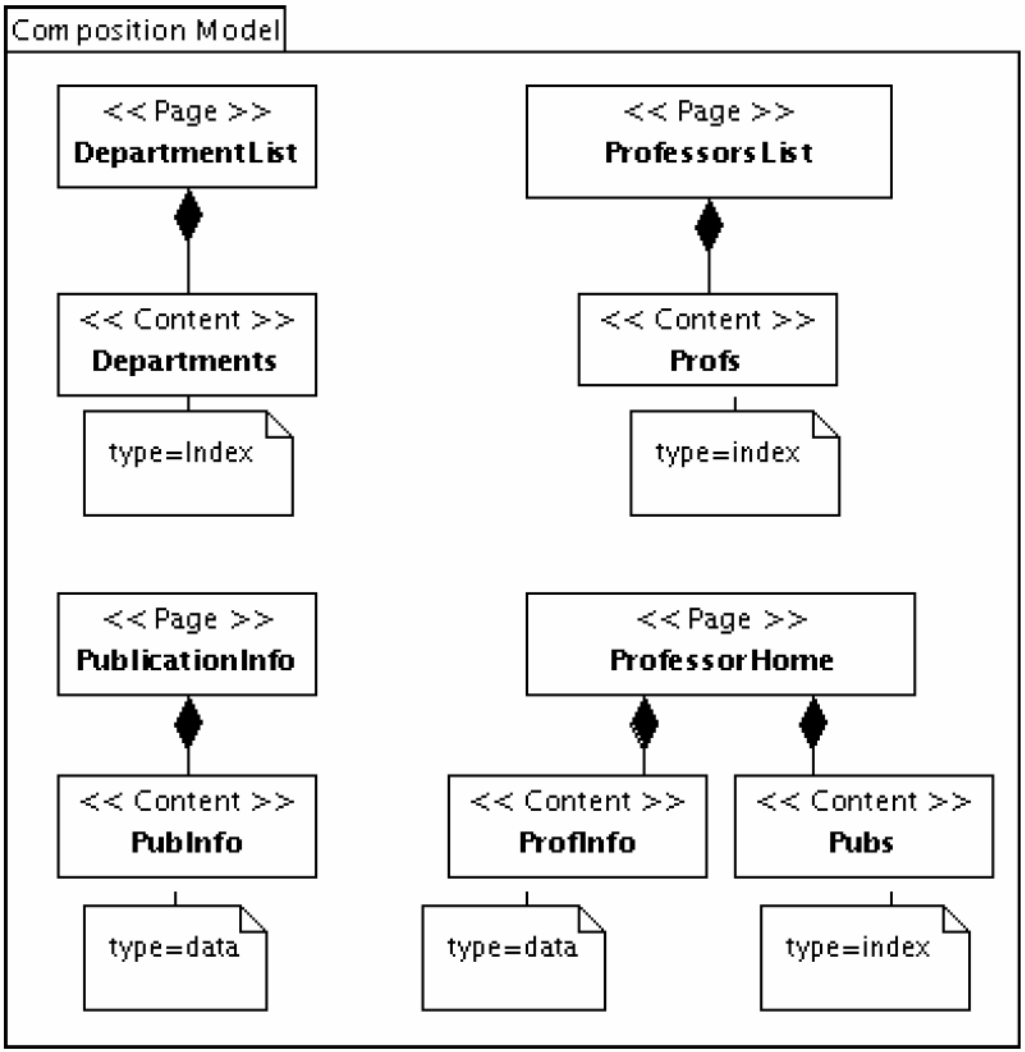
Example – Data Model



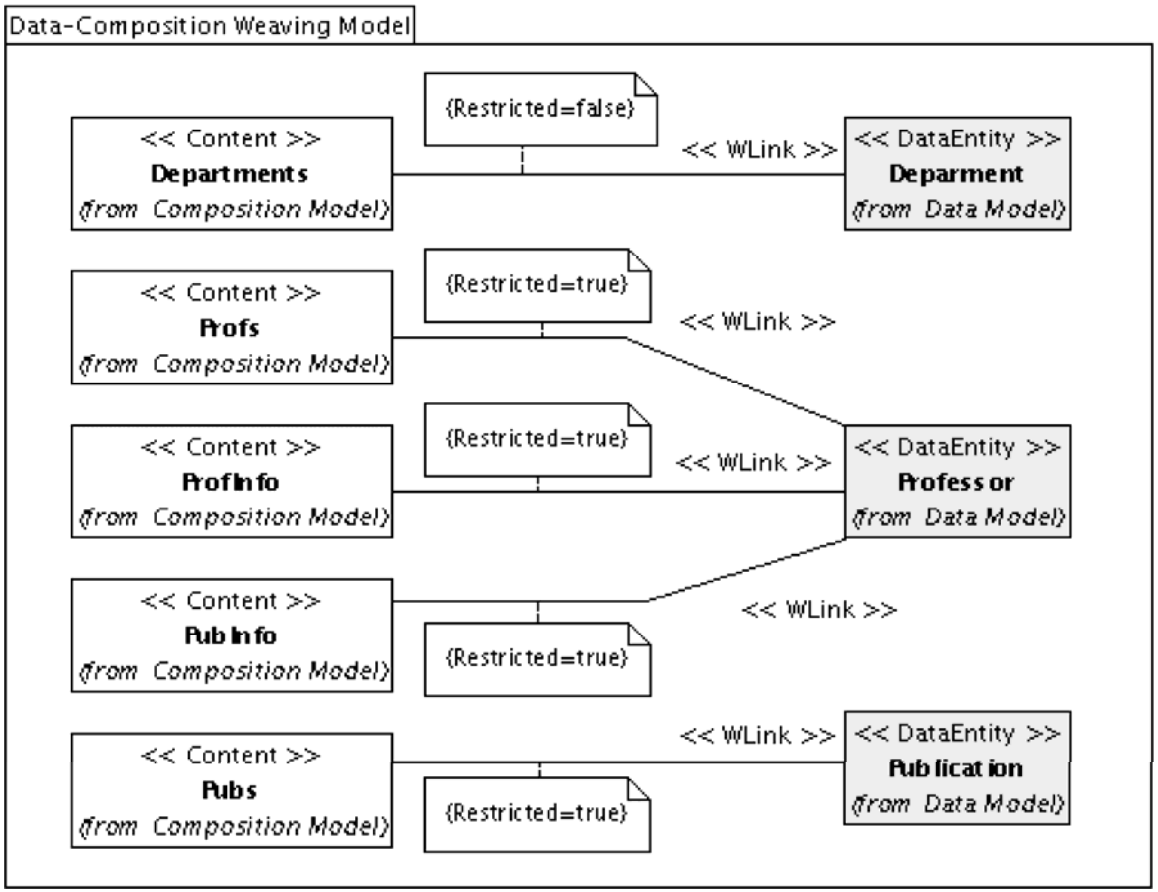
Example – Navigation Model



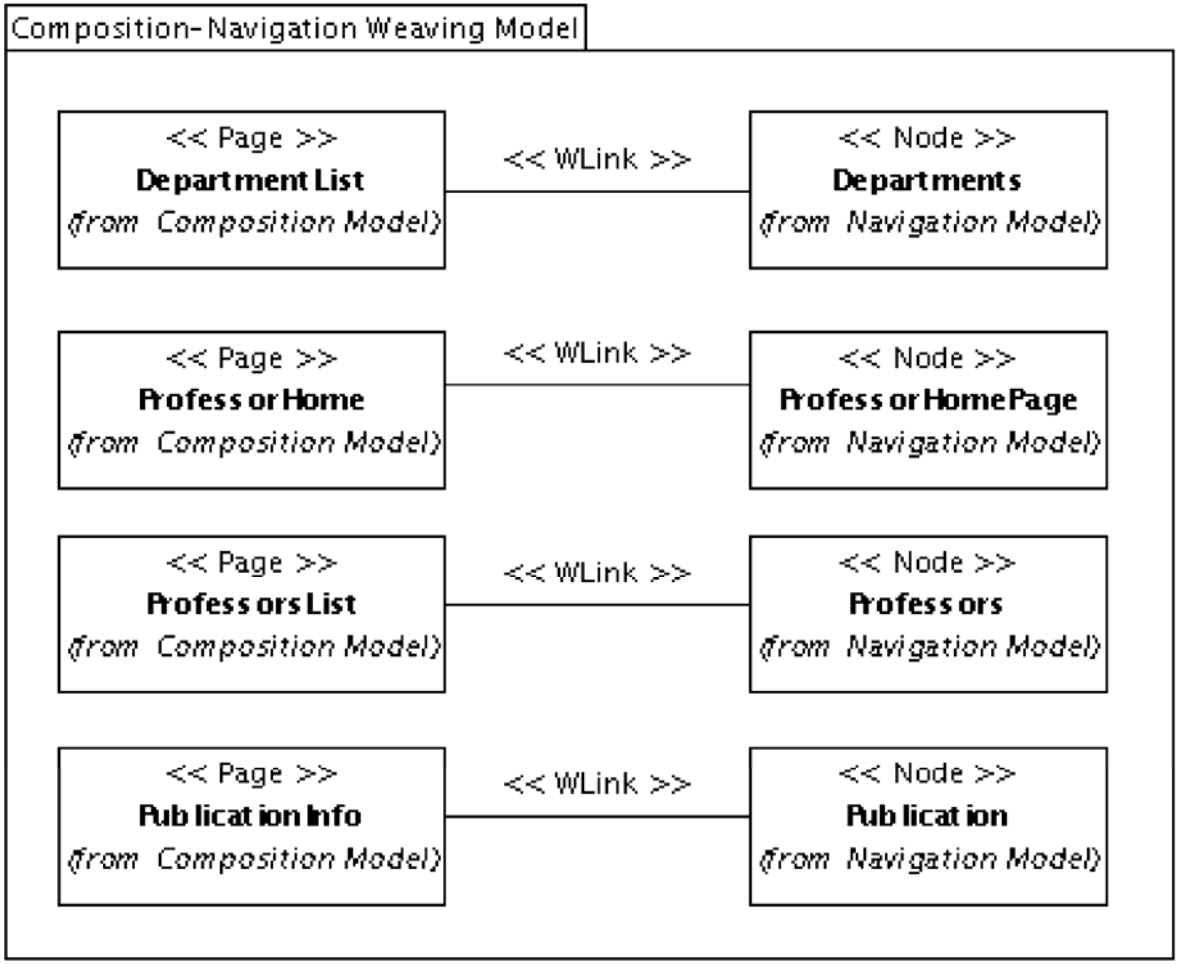
Example – Composition Model



Weaving Data and Composition Models



Weaving Composition and Navigation Models



Target model generation

asm MAIN **is**

…
DataDerivation
StructuredContentDerivation
DataSourceDerivation
LinksDerivationn
…

endasm

Quite simple transformations because of the direct correspondences among the source and target elements

Target model generation

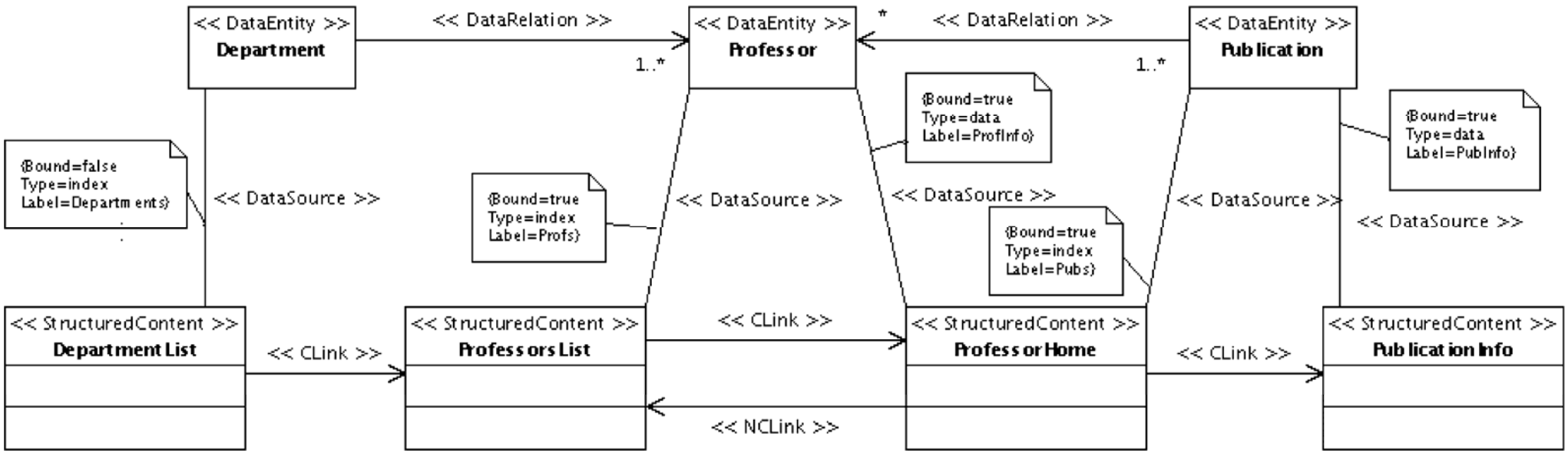
asm MAIN **is**

…
DataDerivation
StructuredContentDerivation
DataSourceDerivation
LinksDerivation
…

endasm

Complex transformation requiring the navigation of all source models for collecting the required information to generate the target elements

Generated target model



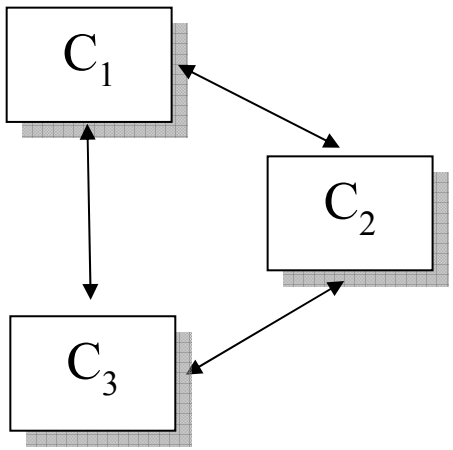
Domain 2

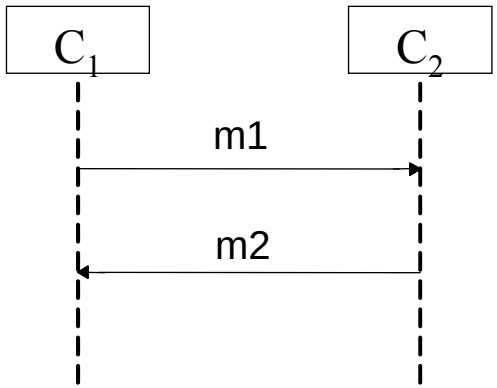
Compositional Reasoning for analyzing Middleware-based Applications

[EWSA05]

Setting the context

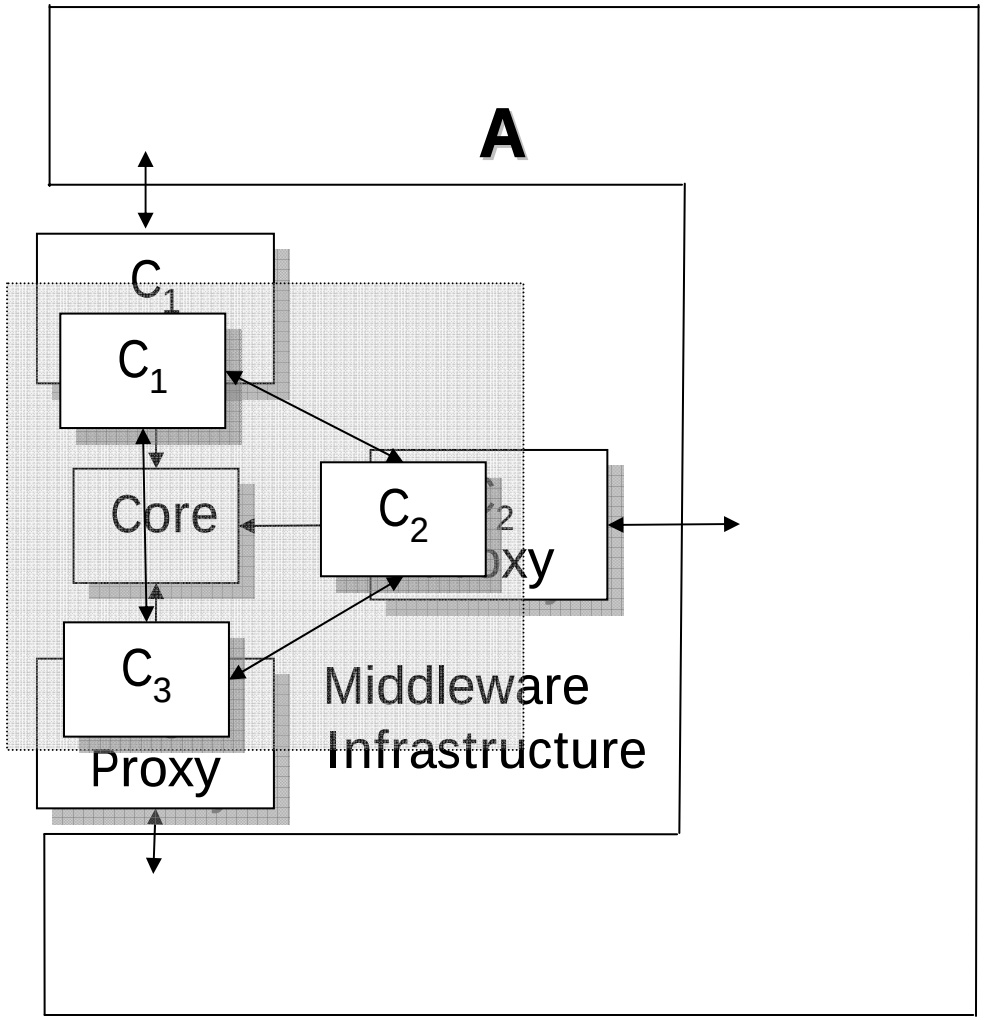
Application **A** $\quad$ **A** $\models$ **Z** $\quad$ Property **Z**





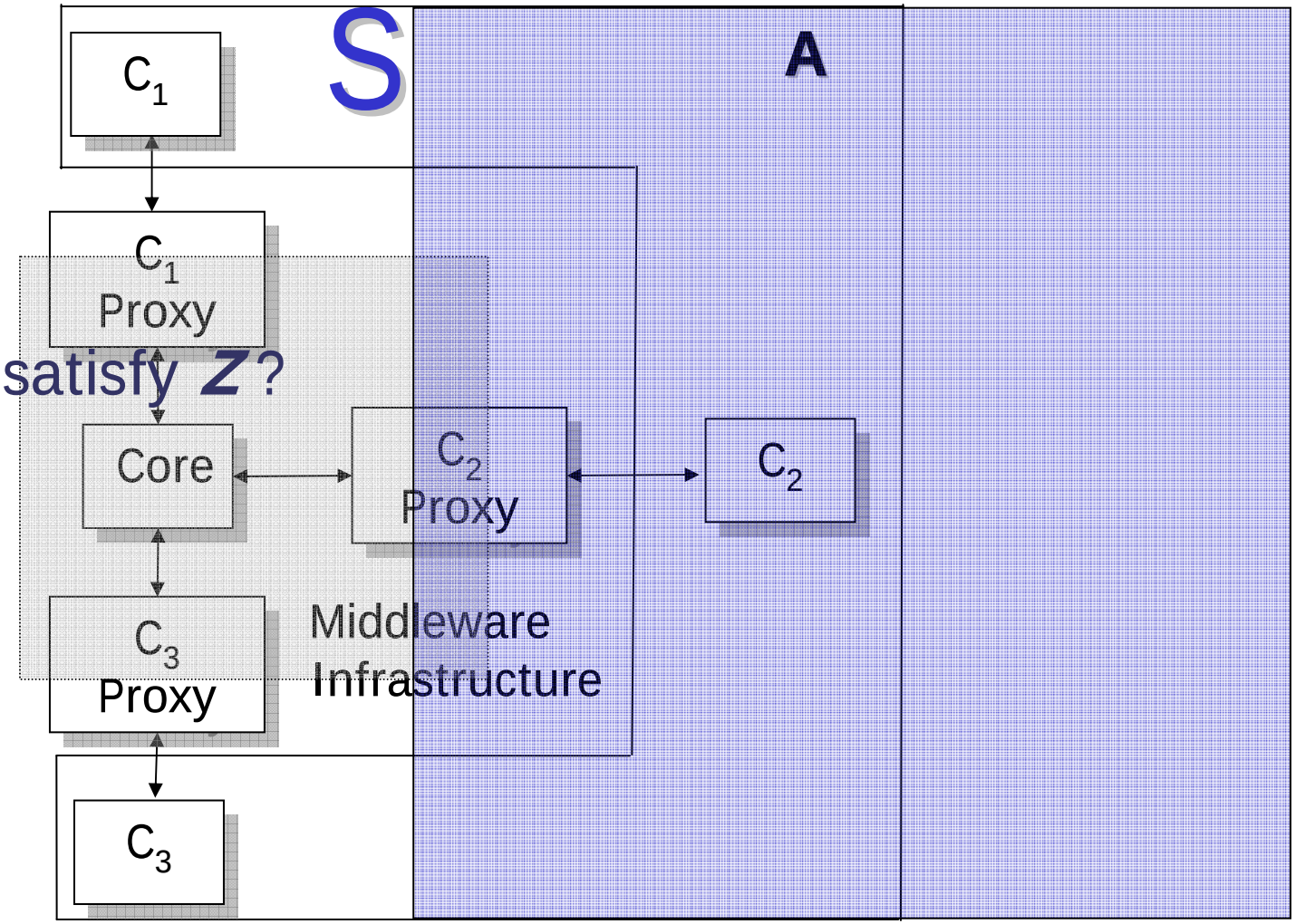
[](m1 -> <> m2)

Setting the context



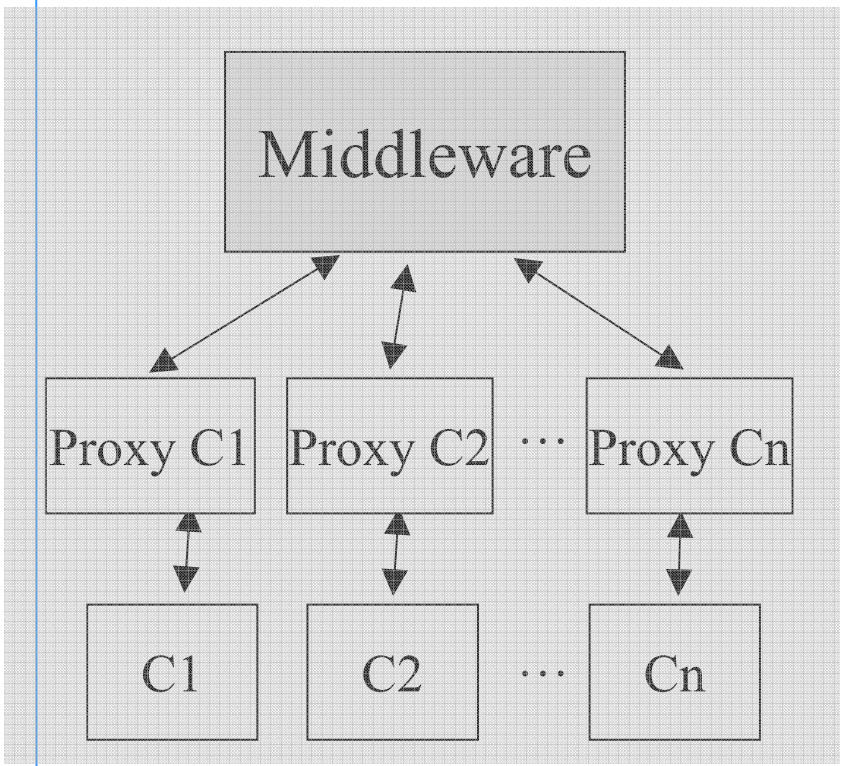
Setting the context

Does **S** still satisfy ***Z***?

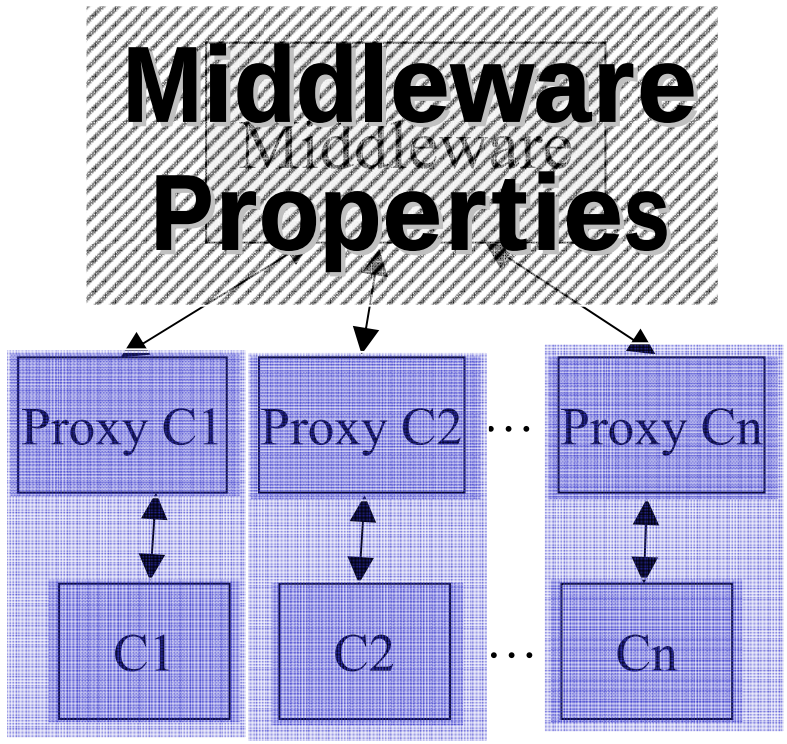


Middleware-based Software Architecture

Classical Approach



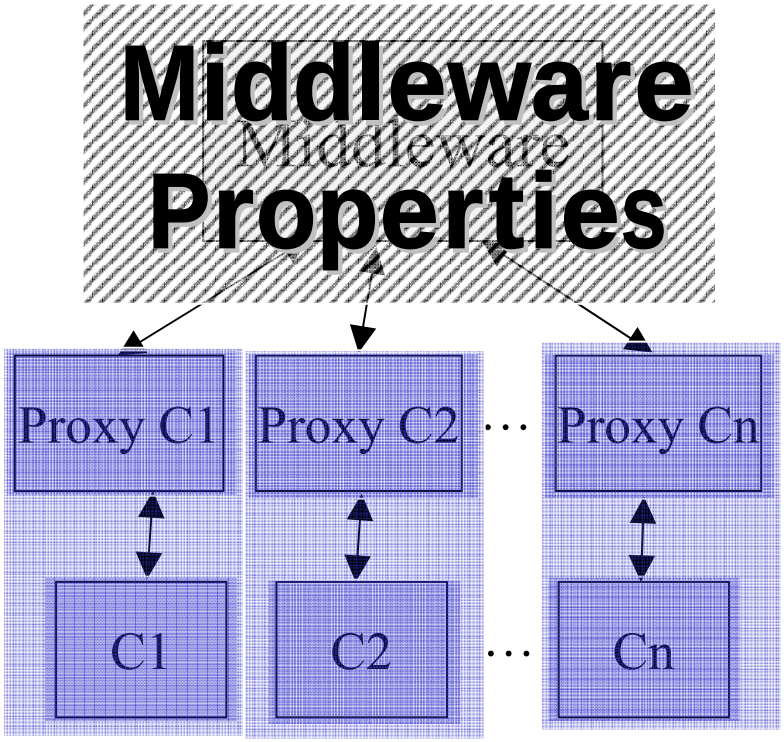
Compositional Approach



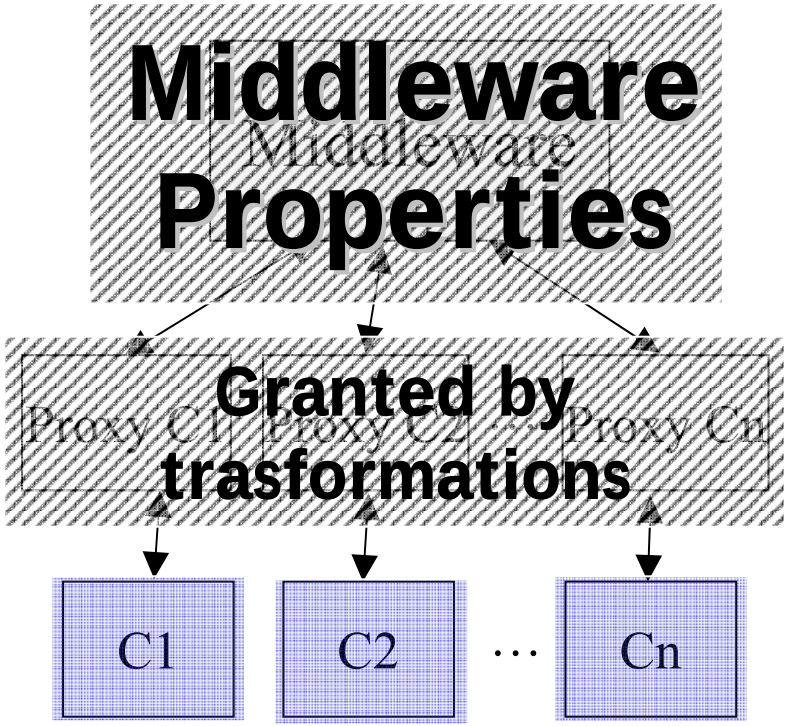
[ICSE04]

Proposed approach

Compositional Approach

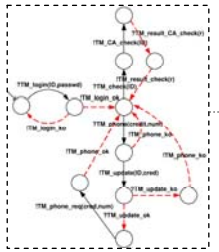


Compositional Approach with Model Transformations



[EWSA05]

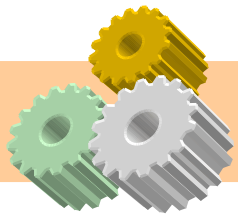
Proxy generation



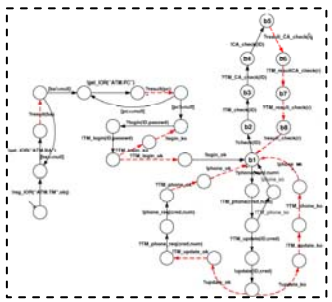
Source
Component behavior
Description model

Algebra encoding the source model

ASM Transformations



Algebra encoding the target model



Target
Proxy behaviour
Description model

Transformation process:

» Target algebra initialization

» Iterative extensions of the target algebra by transforming the methods of the source algebra

> the methods are distinguished into *received* and *sent;*

> the transformation rules are designed in order to preserve the message sequences given in the source model;

Conclusions

» Model transformations play a key role in the OMG's Model Driven Architecture initiative and they are intrinsically difficult

» The presented approach provides with some pragmatics for defining formal model transformations

> Models are queried in a declarative way by means of first-order predicates

> The views are procedurally manipulated

> Rule-based

» It has advantages over structural approaches, ie graph transformations

> Complex computations, eg transitive closure of associations

Things to be done

» The algebraic foundations provides the machinery for investigating more complex issues and properties

> composition of transformation programs

> sequential and parallel independence of model transformations

> system design as model rewriting

> model merging + weaving + difference

» Validation against industrial-size case studies