

UPT: A Graphical Transformation Language based on a UML Profile

Santiago Meliá
Jaime Gómez
Jose Luís Serrano



Web Engineering & Data Warehouse Research Group



Department of Software and Computing Systems



Universitat d'Alacant
Universidad de Alicante

Context (low standarization)

- Model transformations are recognized as being of crucial importance for MDE proposals
- **[CMU/Lewis2005]** MDE proposals do not make use of transformation standards and therefore can not be successfully shared between tools without losing information

Context (Graphical notations)

- The success of any graphical notation (p.e UML) lies in being able to understand the model at first sight
- The graphical notation applied to the transformation models permits to simplify their specification and thus improve the comprehension

Objectives

- To define a standard transformation language that allows us to share them between the different tools
- To reduce the learning curve of the transformation language using a well-known graphical notation

QVT-Merge: transformation standard

- QVT-Merge [nov2005] is the OMG standard for the representation of Model-to-Model Transformations
- QVT has a user/declarative part called Relation language and a tool/imperative part called Operational mappings
- QVT Relation provides a user-friendly graphical and textual notation

QVT Relation Problems

- QVT Relation language presents the following problems:
 - Its graphical notation is not compliant with the current UML tools
 - It represents the metaclasses instances as objects instead of classes
 - Up to now, there is no tool able to support the QVT graphical notation
 - It requires a learning process before it can be used

UPT (UML Profile 4 Transformations)

- UPT is a simple declarative language based on the UML Class Diagram for specifying metamodel transformations
- UPT presents a small MOF metamodel establishing the necessary elements for its specification so as to be able to define it as a UML profile

UPT main advantages

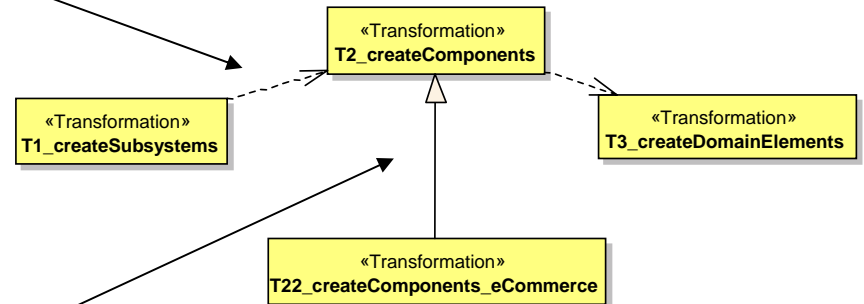
- It can be expressed by a great number of tools able to represent UML models graphically
- UPT sends transformation models to different tools by means of the standard XMI
- UPT introduces new concepts and new semantics to represent transformations through profile mechanisms
- The use of UPT needs a short learning curve to modelers who want to define transformations



UPT Elements: Transformation

- **Transformation:** is a set of relations or transformation rules that have to be carried out within the elements of a set of metamodels.

- the order of execution can be expressed



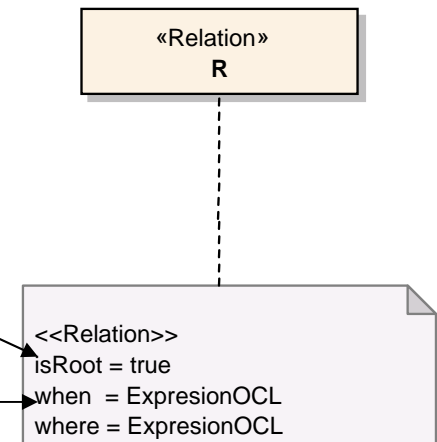
- A Transformation can be extended

UPT Elements: Relation

- **Relation:** indicates the restrictions that the elements of the model candidates must fulfil
- Each relation is made up of two or more domains and two restrictions known as when (or guard) and the clause where.

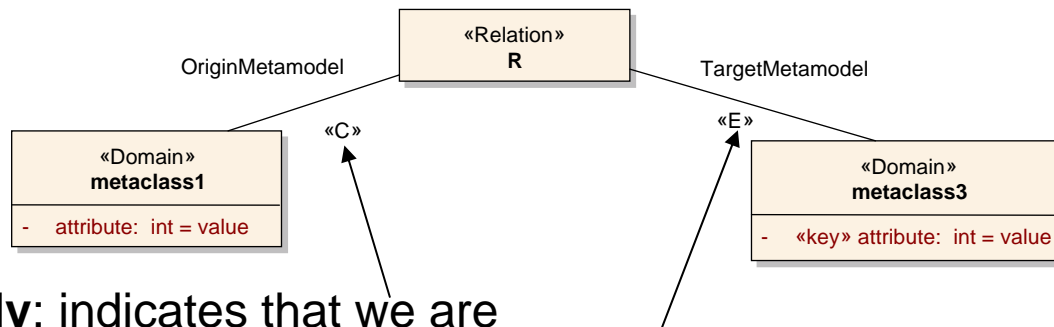
IsRoot: indicates whether the relation is invoked from the transformation or from a where belonging to another relation

When and **Where** are OCL conditions. When must be accomplished by the relation before its execution and Where after its execution



UPT Elements: Domain

- **Domain:** specifies the set of elements of a typed model (referred metamodel) that are of interest for a relation. It is represented by an association relationship between the Relation and the DomainPattern
- There are two types: Checkonly (<<C>>) and Enforceable (<<E>>)



Checkonly: indicates that we are going to make sure that the elements defined in the domain are to be found in the same way in the element of reference

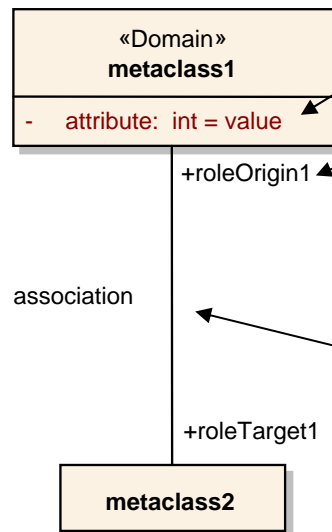
Enforceable: indicating that if the domain elements are not the same as the reference model we must modify them

UPT Elements: DomainPattern

- **DomainPattern:** specifies a template class where the Classes belong to different metaclasses of the domain reference metamodel. A Domain Pattern has the following elements:

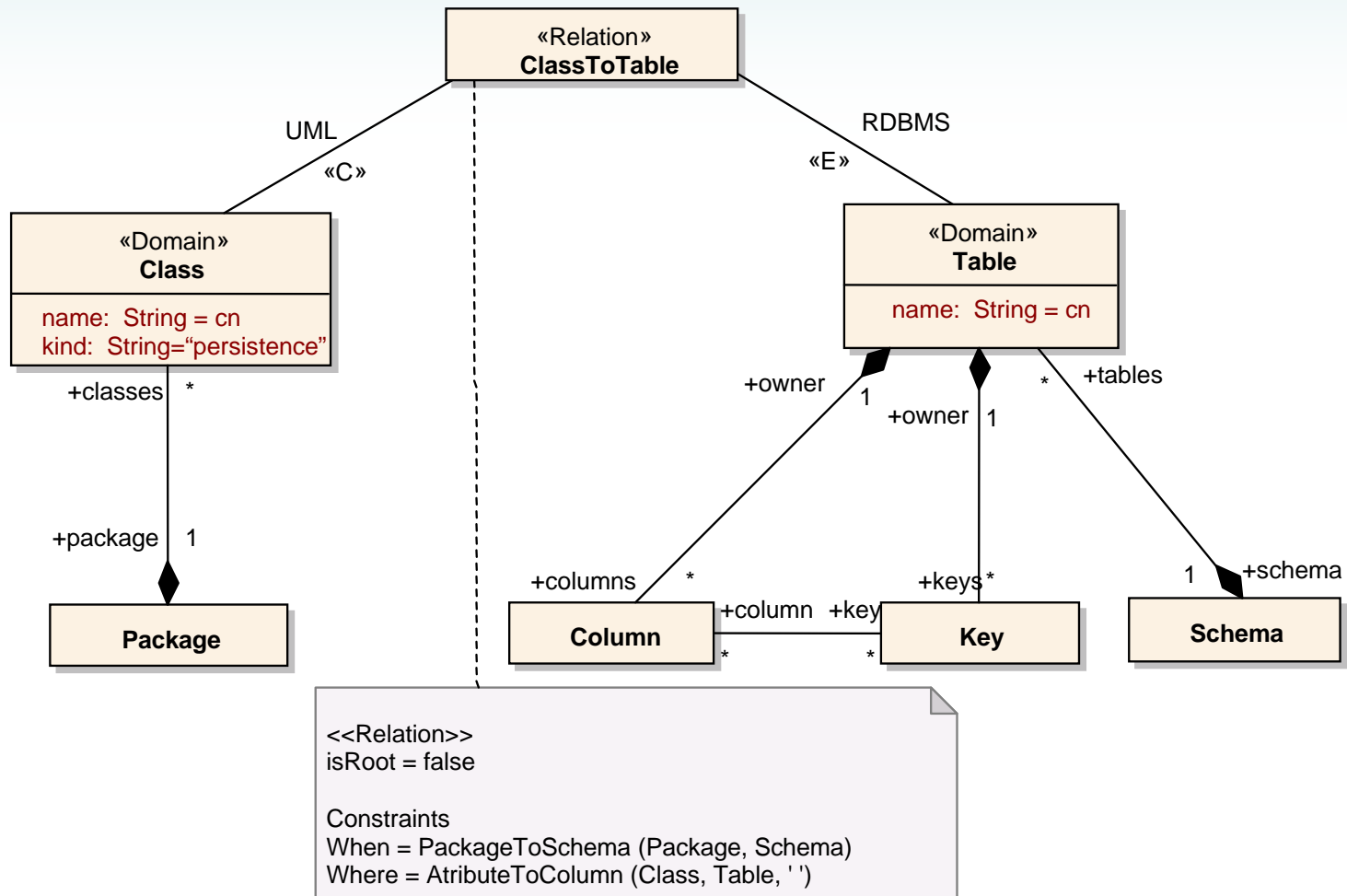
ClassPattern: it receives this name because it refers to a metaclass instance, that is, a class

PropertyPattern: it specifies restrictions in the values that the ClassPattern can take. There are two types: attribute and role



AssociationPattern: establishes a relationship between two ClassPatterns

UPT Example



<<Relation>>
 isRoot = false

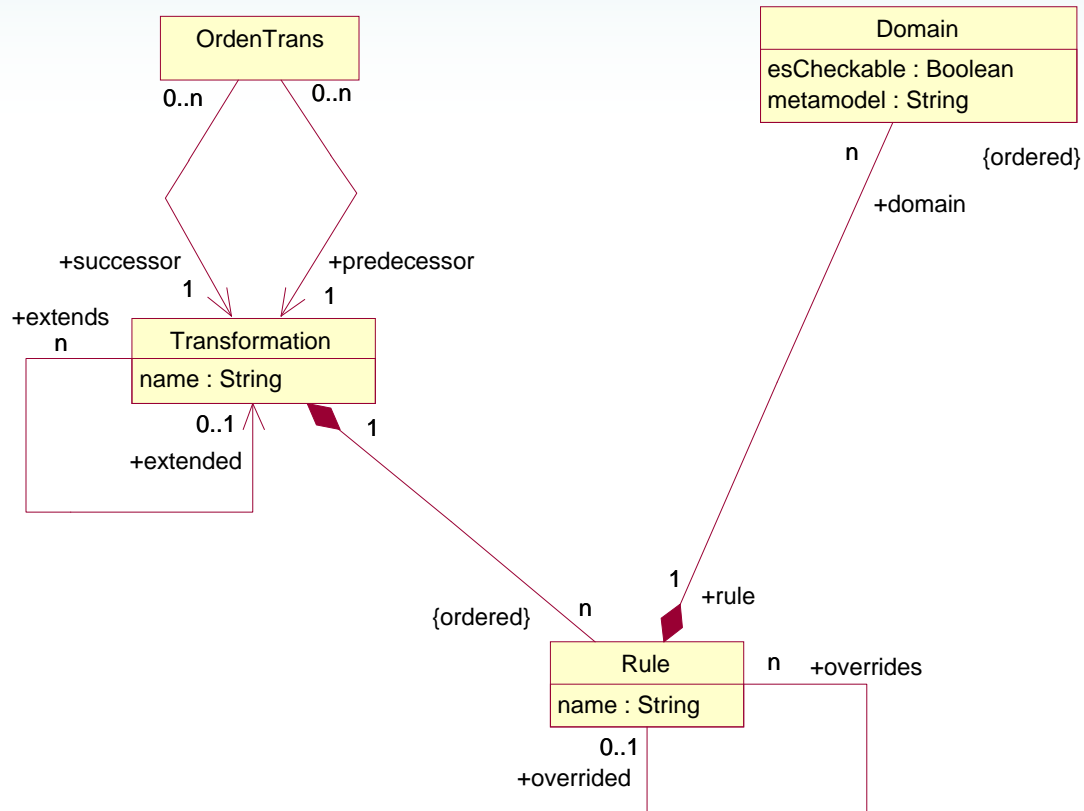
 Constraints
 When = PackageToSchema (Package, Schema)
 Where = AttributeToColumn (Class, Table, ' ')



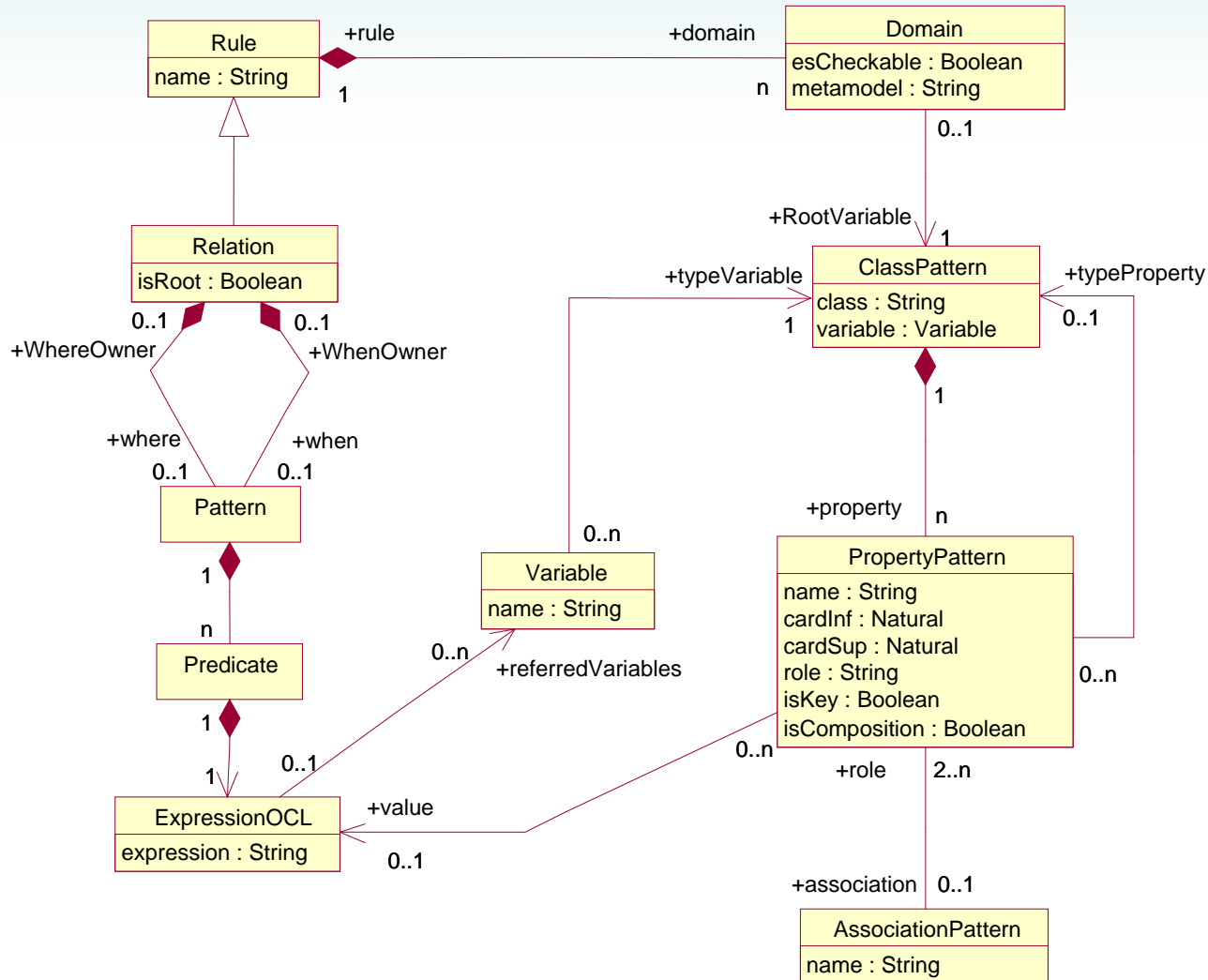
UPT Metamodel

- It is a MOF metamodel
- It formalizes the UPT elements and the relationships that make it possible to represent graphically transformations and relations
- UPT metamodel is simple and compatible with the UML metamodel

UPT Metamodel: Transformation package



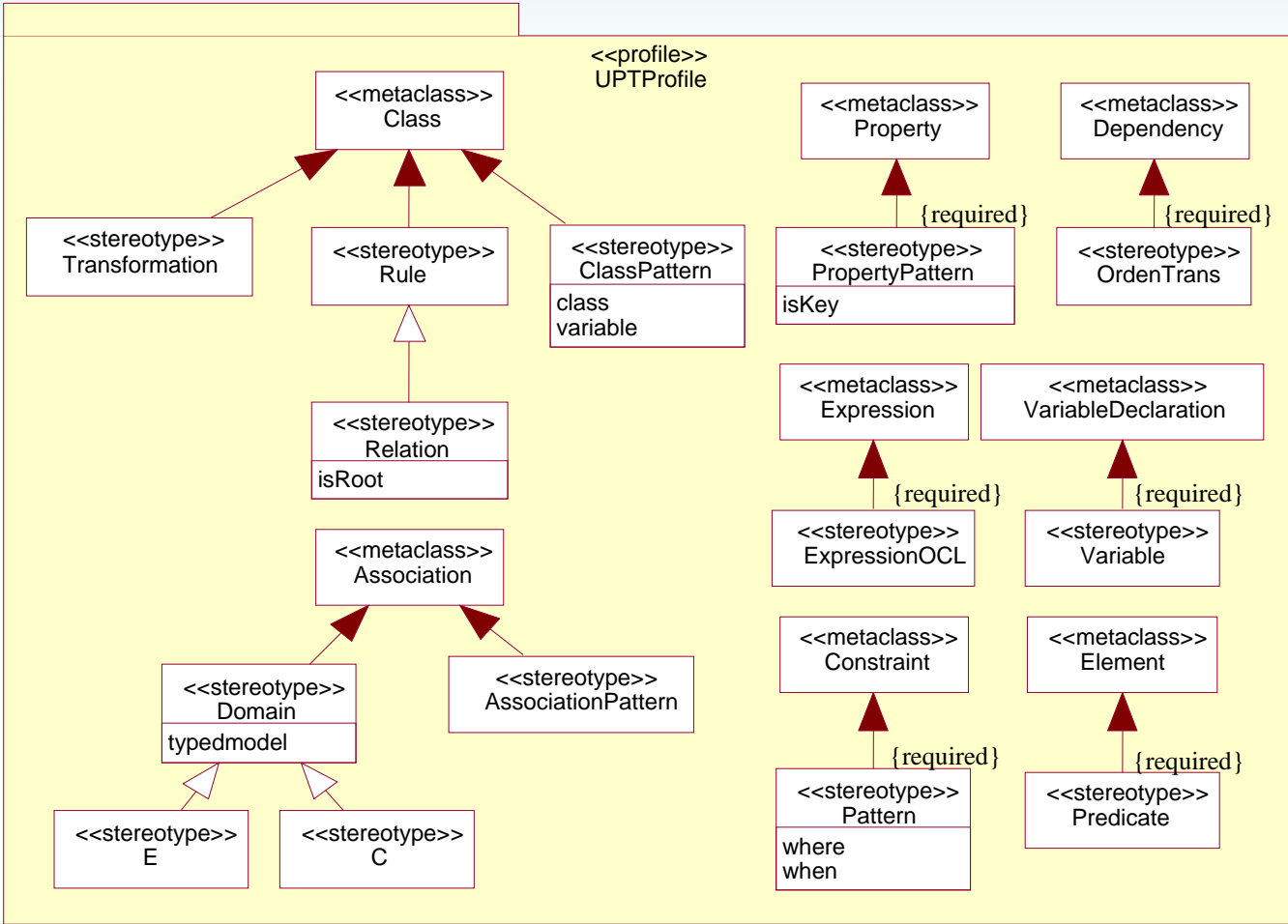
UPT Metamodel: Relation Package



UPT Profile

- It is the adaptation of each one of the concepts defined by the UPT metamodel to the UML and OCL standards
- UPT Profile has followed the alignment proposed by the OCL specification which allows us to insert the different OCL expressions in the UML class diagram

UPT Profile

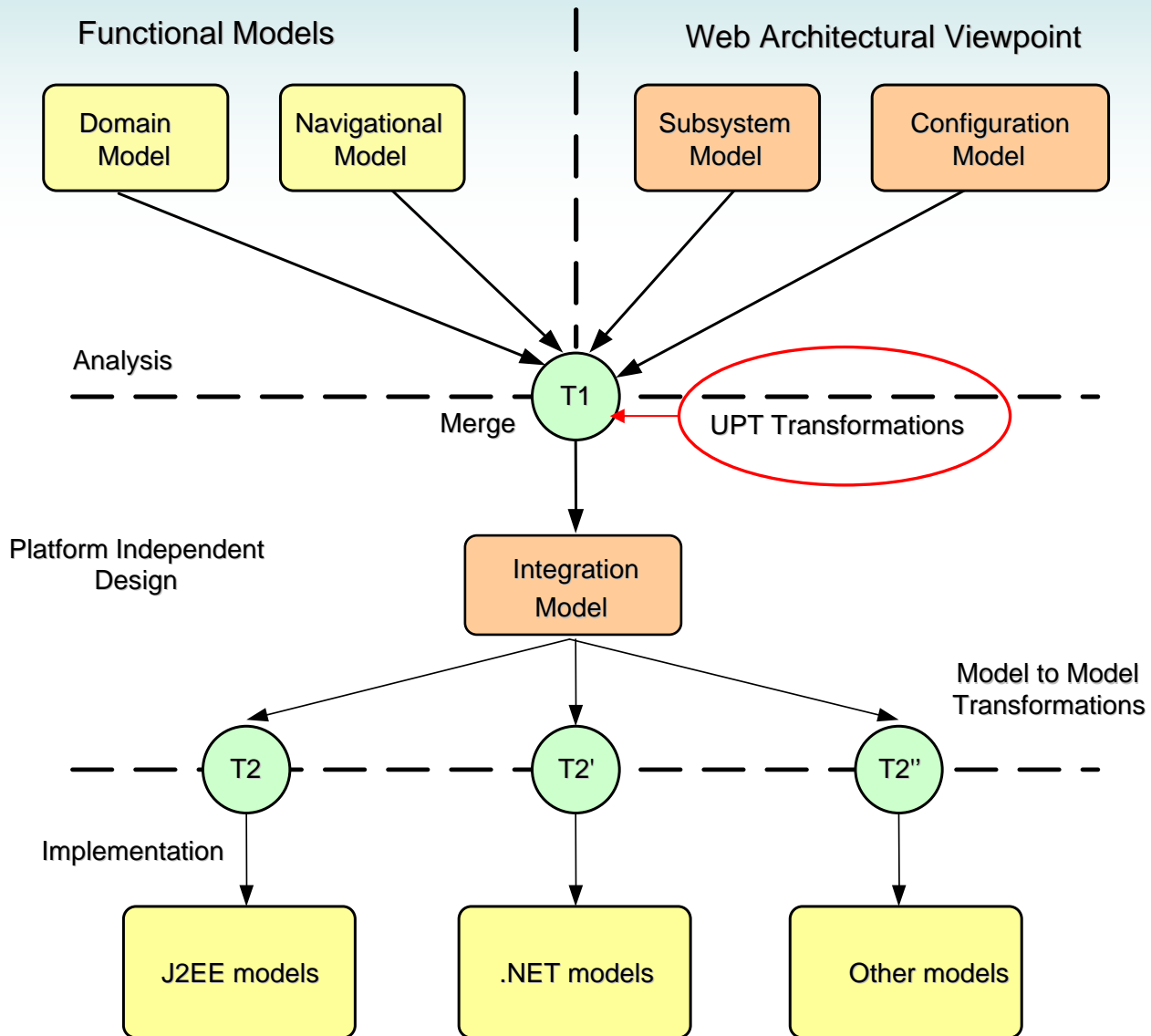


WebSA (Web Software Architecture)

- WebSA [melia2005] is a MDE proposal based on two main aspects:
 - The proposal of a top-down **MDA development process** for the Web application domain to transform models to code
 - The definition of specific **software architectural models and transformations** for Web applications to complement other Web methodologies, such as OO-H or UWE



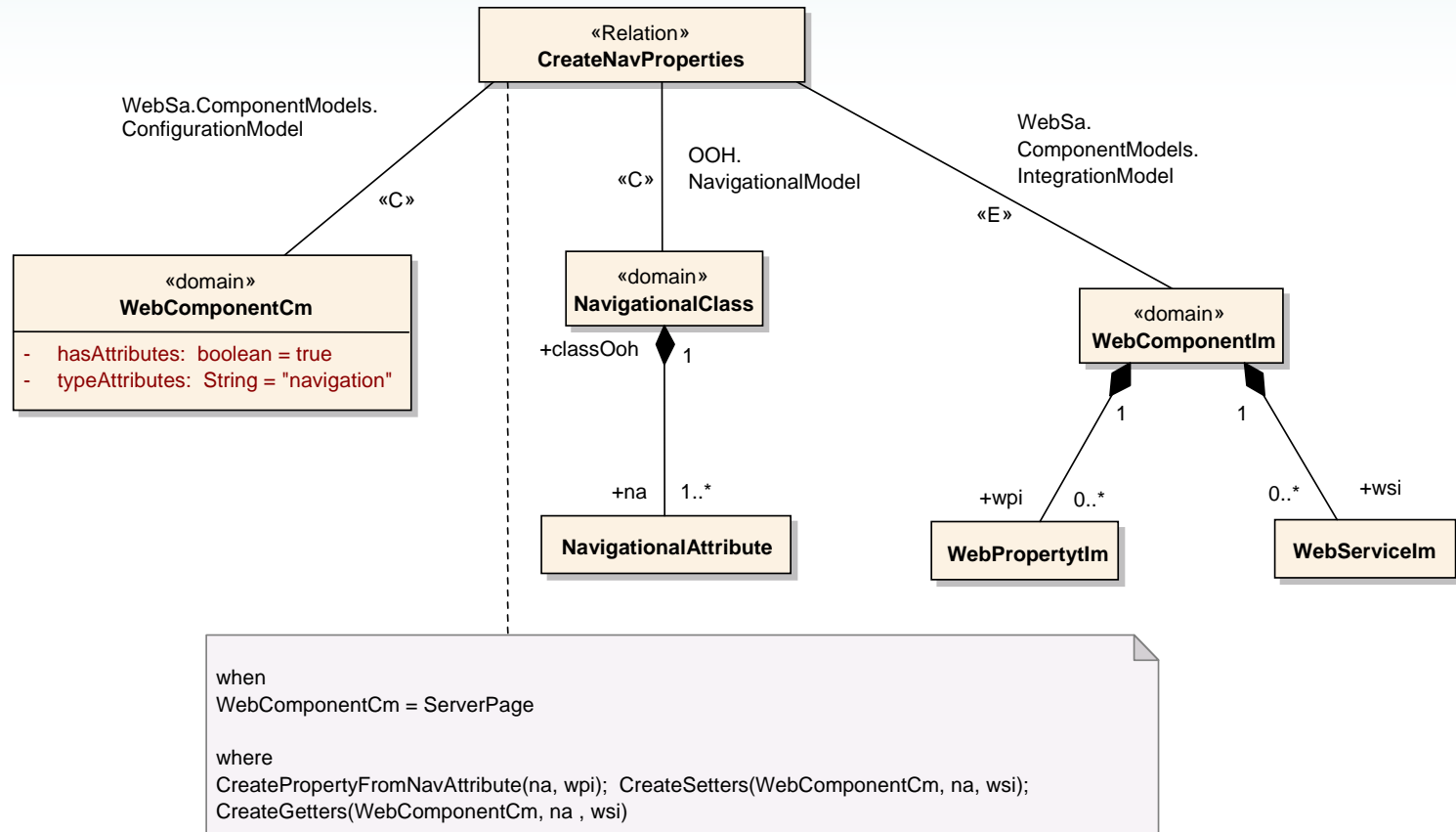
WebSA Development Process



T1 Transformation

- T1 is complex model-to-model transformation which is made up of a sequence of 5 transformations which are composed by 80 relations
- PIM to PIM transformation
- Model Merging transformation (4-to-1)
- Driven by architectural models

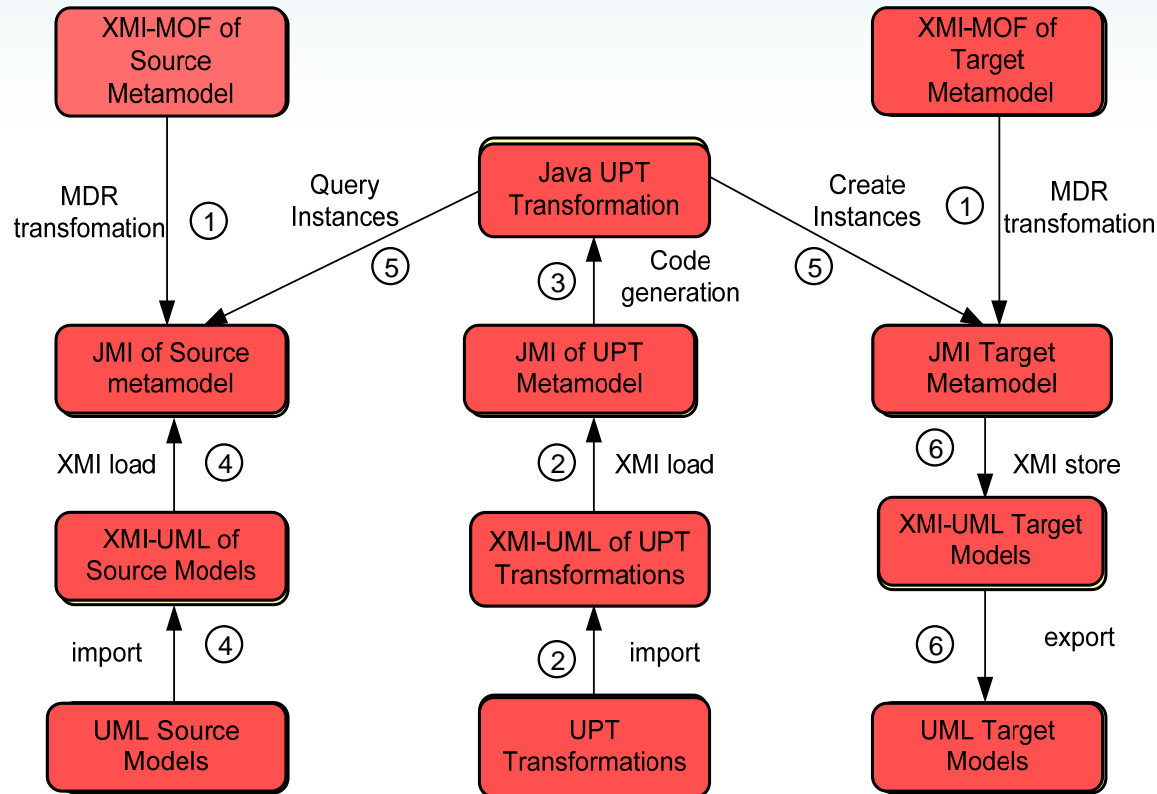
A UPT Relation of T1 transformation of WebSA



UPT Tool

- It is a Web tool developed on J2EE platform which implements the UPT model transformations
- It is based on the OMG standards (UML, XMI, OCL, etc.) for two reasons:
 - Optimize the implementation effort using COTS such as parsers, compilers, etc
 - facilitate the use of any type of UML tool that possesses the support for class diagrams

Generic UPT Tool



WebSA Tool

The screenshot displays the WebSA Tool interface. A central browser window shows the XML output of a transformation:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
- <XMI xmi.version="1.2" timestamp="Fri Jul 07 18:13:15 CEST 2006">
+ <XMI.header>
- <XMI.content>
+ <WebSA.ComponentModels.IntegrationView.IntegrationModel xmi.id="a1">
+ <WebSA.ComponentModels.IntegrationView.WebConnectorIm xmi.id="a106"
  name="">
+ <WebSA.ComponentModels.IntegrationView.WebConnectorIm xmi.id="a108"
  name="assembly3">
+ <WebSA.ComponentModels.IntegrationView.WebConnectorIm xmi.id="a110"
  name="">
+ <WebSA.ComponentModels.IntegrationView.WebComponentIm xmi.id="a112">
<WebSA.ComponentModels.IntegrationView.WebComponentIm
  xmi.id="a117" />
<WebSA.ComponentModels.IntegrationView.WebComponentIm
  xmi.id="a118" />
+ <WebSA.ComponentModels.IntegrationView.WebComponentIm xmi.id="a119"
  name="Model_Cart">
<WebSA.ComponentModels.IntegrationView.WebComponentIm
  xmi.id="a121" />
+ <WebSA.ComponentModels.IntegrationView.WebComponentIm xmi.id="a122"
  name="Model_Customer">
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a124" />
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a125" />
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a126" />
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a127" />
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a128" />
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a129" />
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a130" />
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a131" />
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a132" />
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a133" />
<WebSA.ComponentModels.IntegrationView.WebPortIm xmi.id="a134" />
```

The sidebar on the right contains a table of execution results:

Execution	Status
T1	Loaded
	Loaded
	Loaded
	Loaded
	Loaded
	Loaded
Last Executions	
Integration Model	
Petstore	
Examinar...	
Examinar...	
Execute	



Conclusions & Future Works

- Conclusions
 - UPT offers an easy graphical and standard notation to represent metamodel transformations
 - The UPT transformations can be represented in any UML tool and can also be shared using XMI
- Future Works
 - improve the process of converting the XMI-MOF metamodel into JMI to make this conversion automatically

Thanks for your attention!!



Questions?

For further comments...

santi@dlsi.ua.es

jgomez@dlsi.ua.es

monfor@dlsi.ua.es