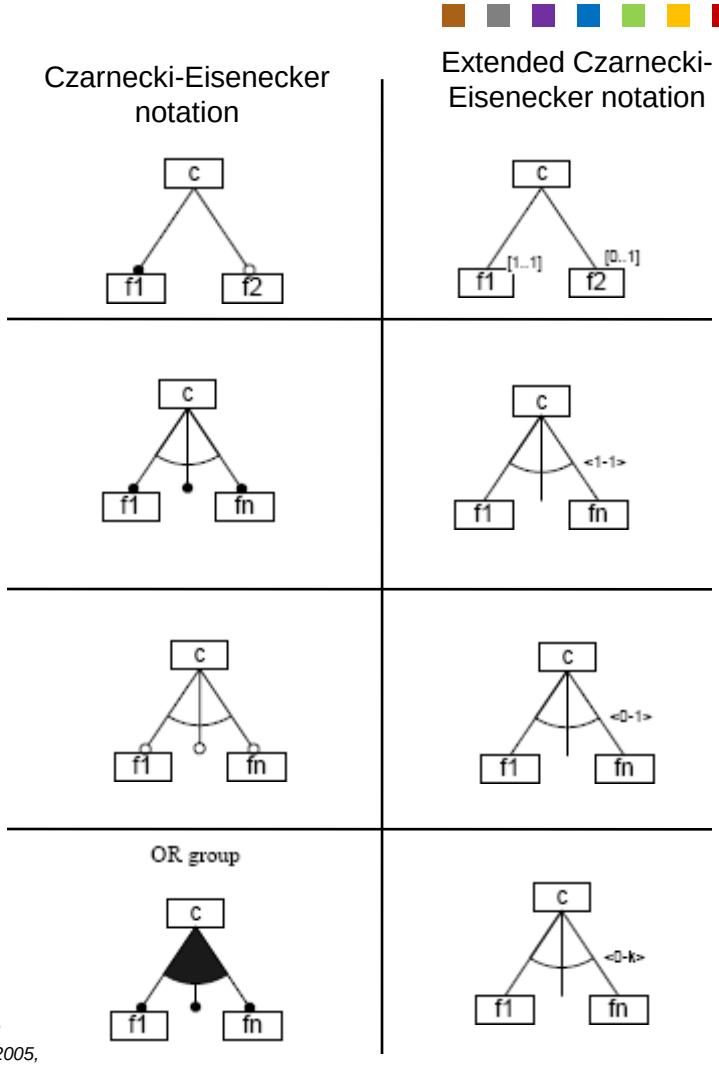
Representing feature models as class diagrams

Abel Gómez, Isidro Ramos
Technical University of Valencia.
Departamento de Sistemas Informáticos y Computación.

Metamodels

Notation

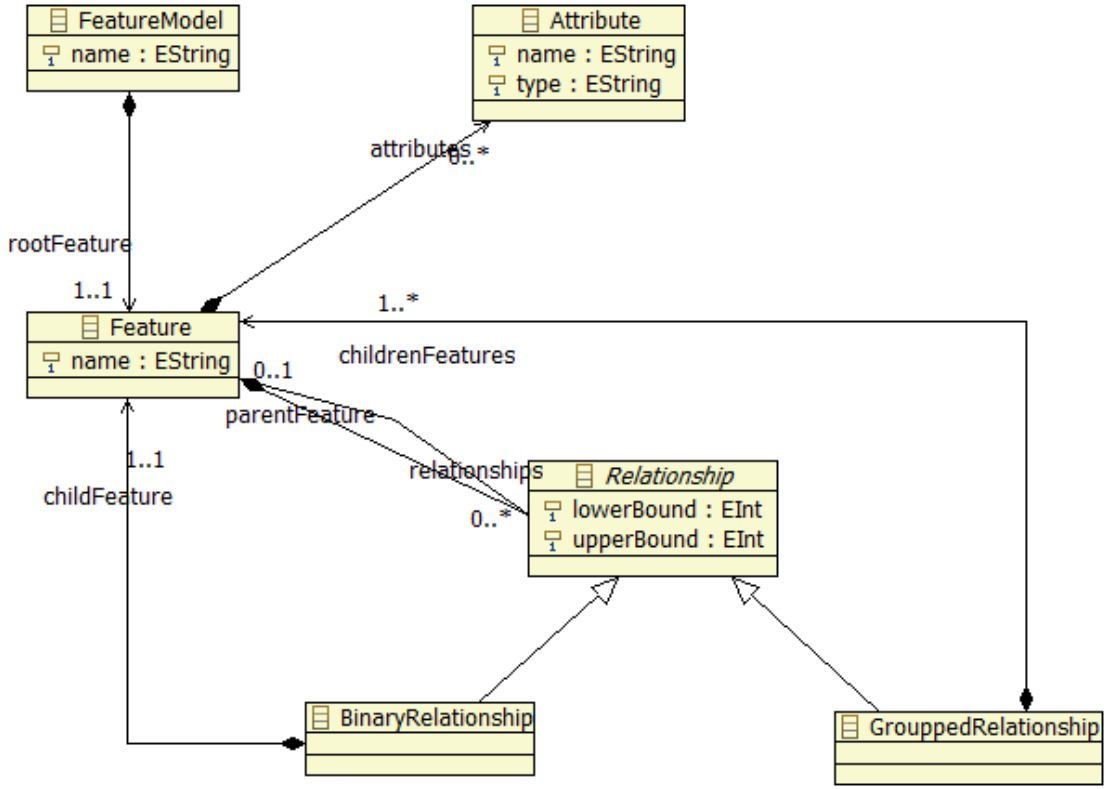
- Binary relationships:
  - Mandatory
  - Optional
- Groupped relationships:
  - Alternative (XOR)
    - XOR
    - XOR with optional subfeatures
  - Optional (OR)



* Figures from *ŠÍPKA, M.: Exploring the Commonality in Feature Modeling Notations. In: IIT.SRC 2005 – Informatics and Information Technology Student Research Conference. Bratislava, April 2005, V. Vranić (supervisor), – pp. 139–144.*

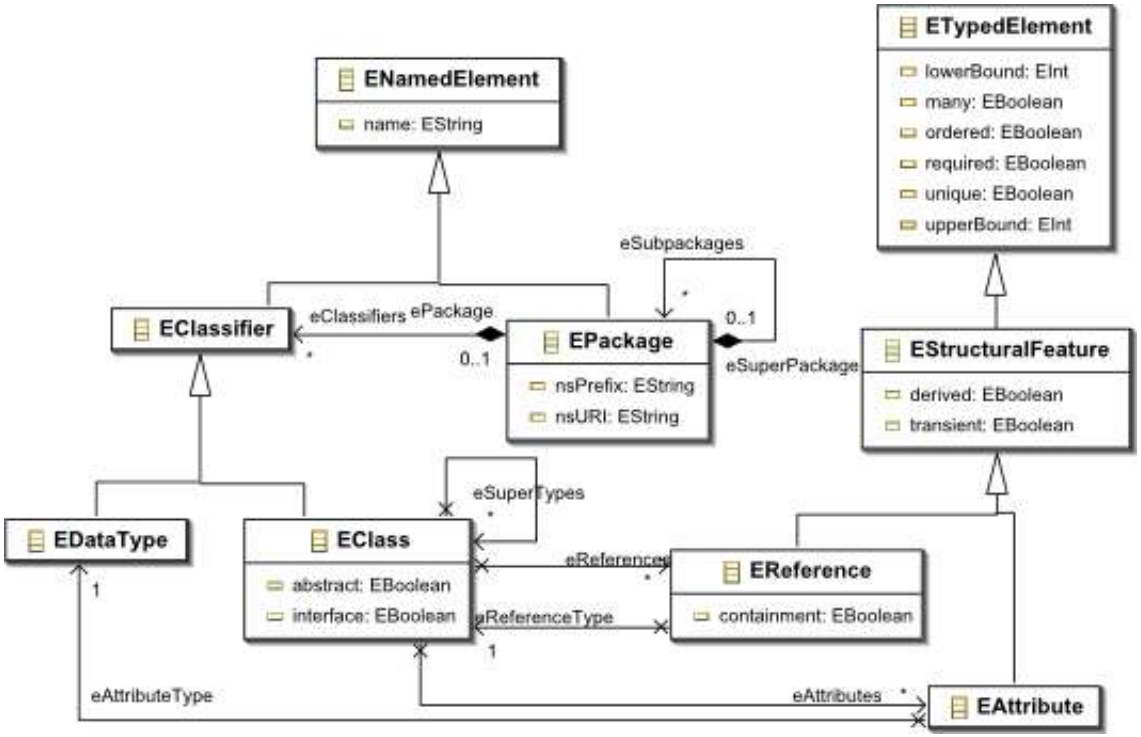
Feature metamodel

- In the feature metamodel the previous relationships are described by using the *lowerBound* and *upperBound* attributes.
- We have also extended the metamodel in order to enrich the features with attributes.



Simplified UML-Class diagram metamodel

- We use Ecore as a simplified version of the UML Class diagram metamodel.



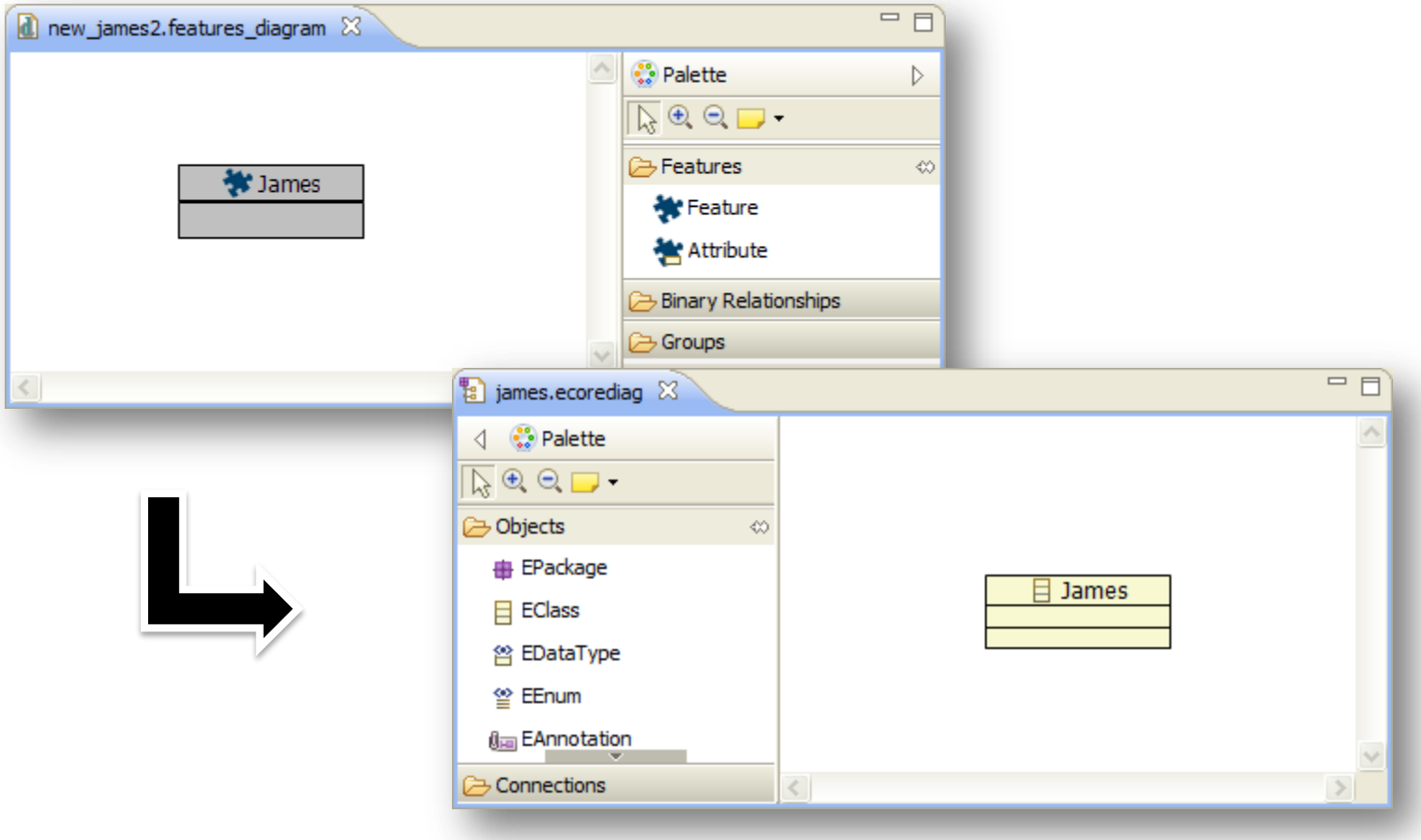
* Simplified subset of the Ecore metamodel

Transformation

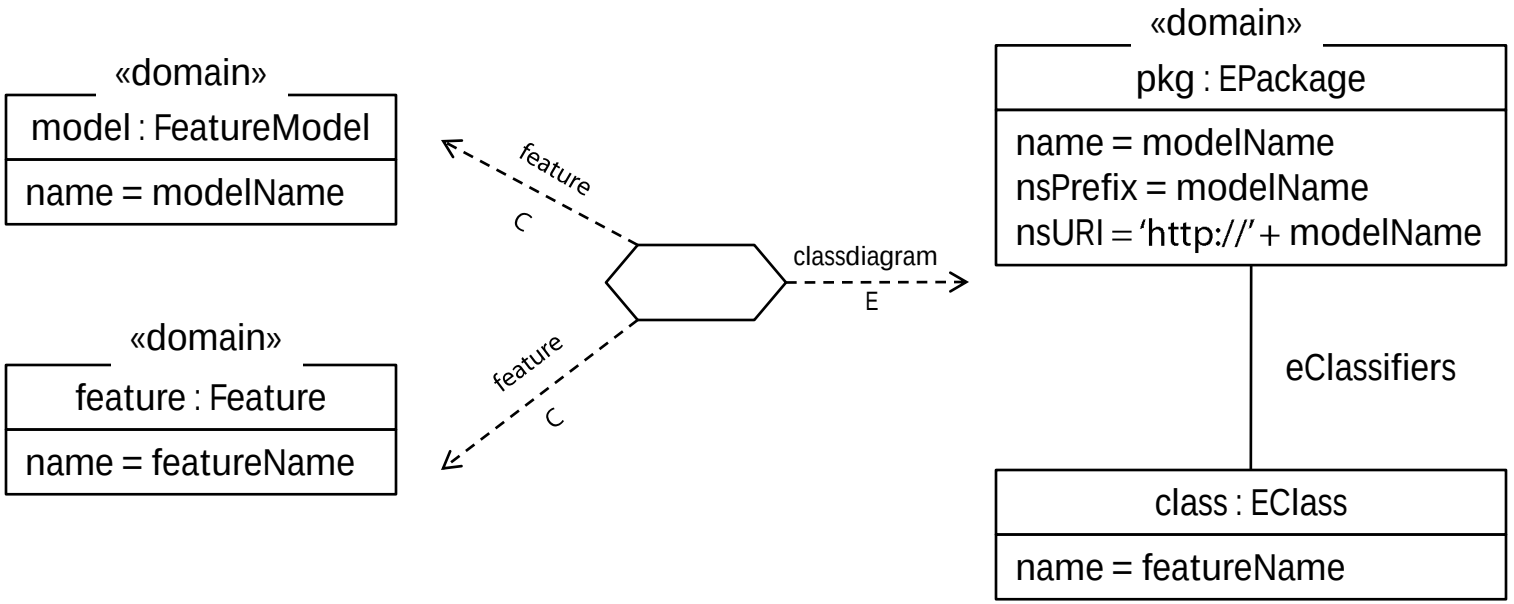
From feature models to class diagrams

Relation Feature2Class

- Each *feature* is transformed to a *class* (Eclass).

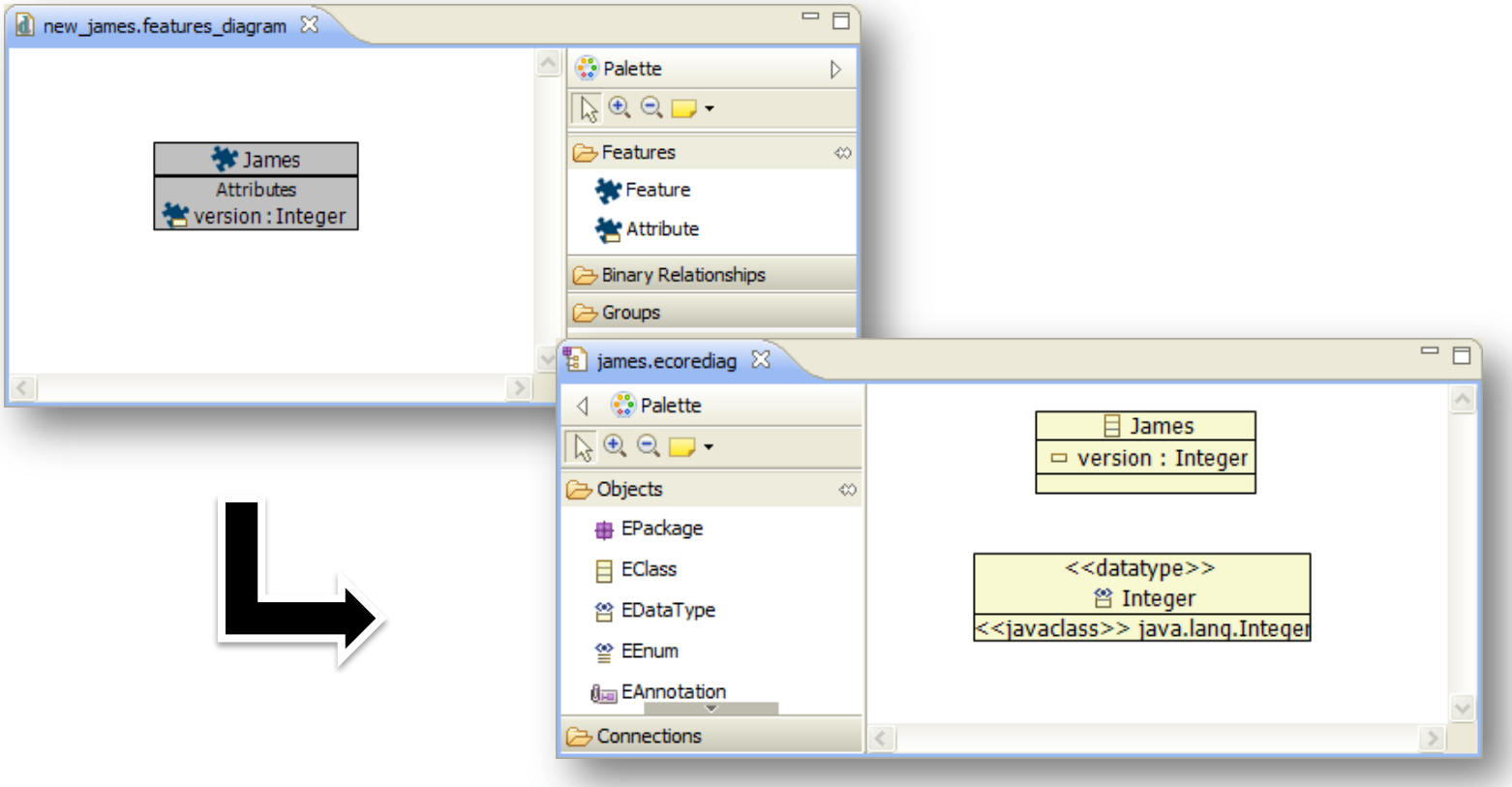


Relation Feature2Class

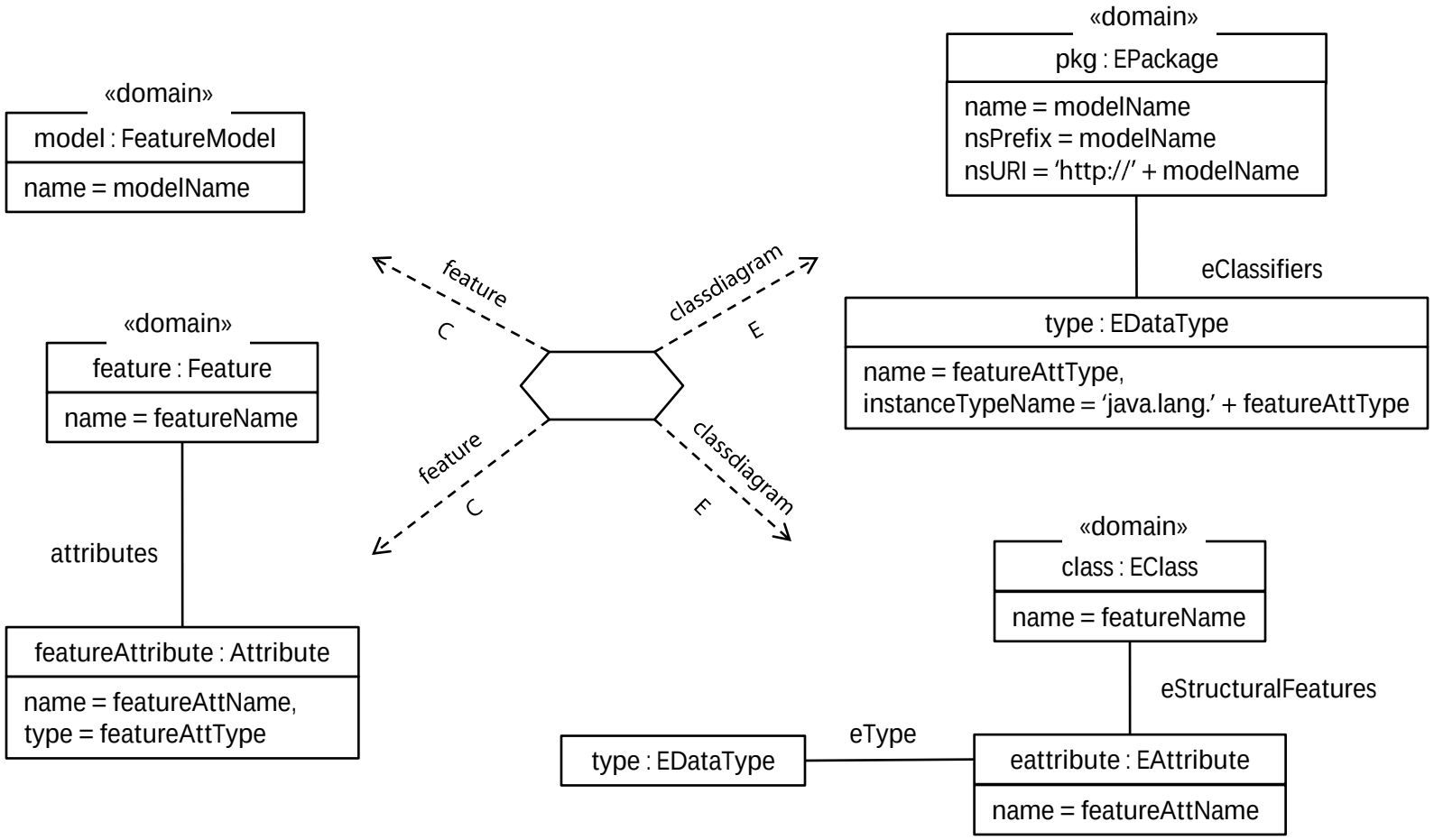


Relation FeatureAttribute2ClassAttribute

- Each feature *attribute* is transformed to a *class attribute* (EAttribute).
- The *type* of the attribute is transformed to a *EDataType*.

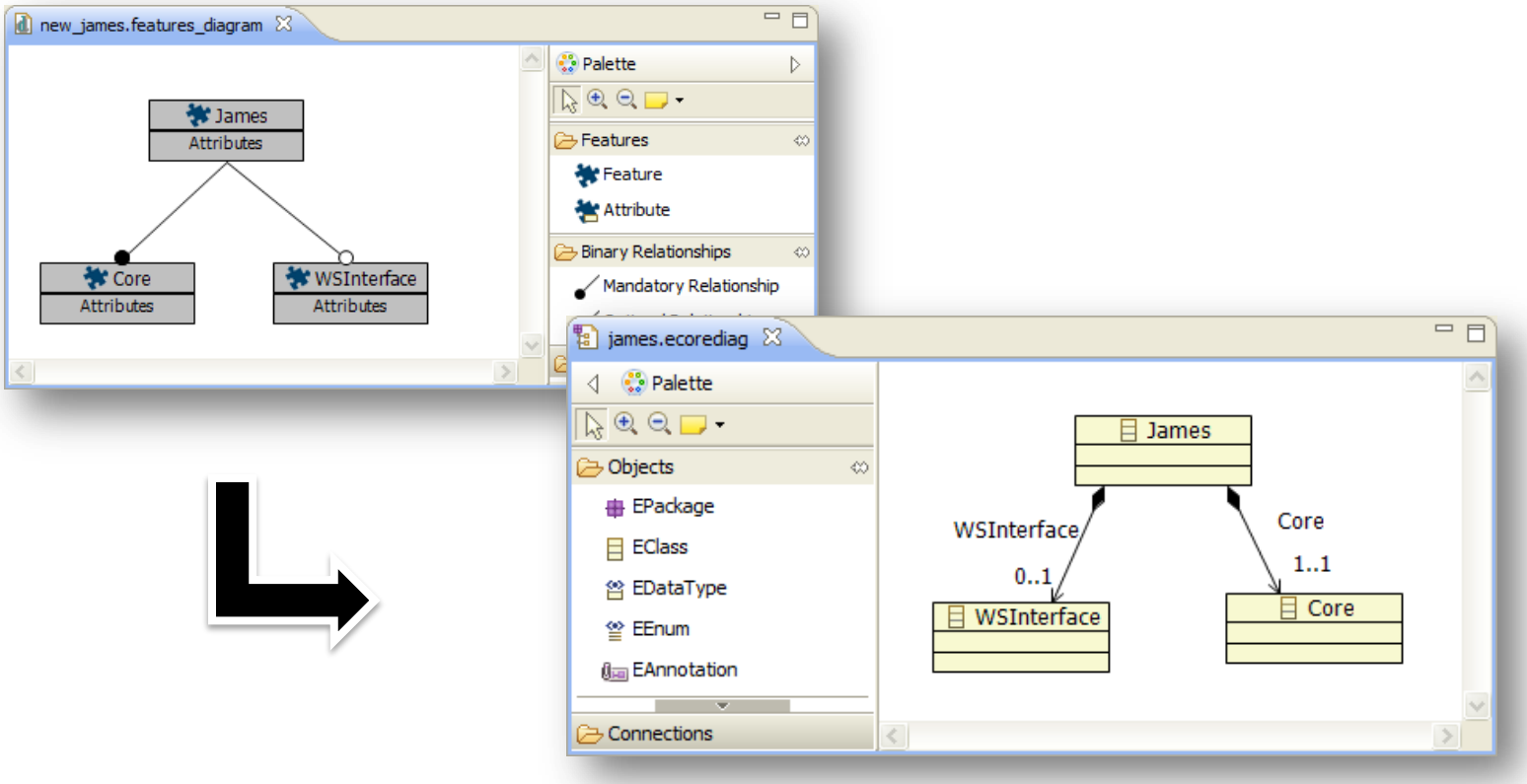


Relation FeatureAttribute2ClassAttribute

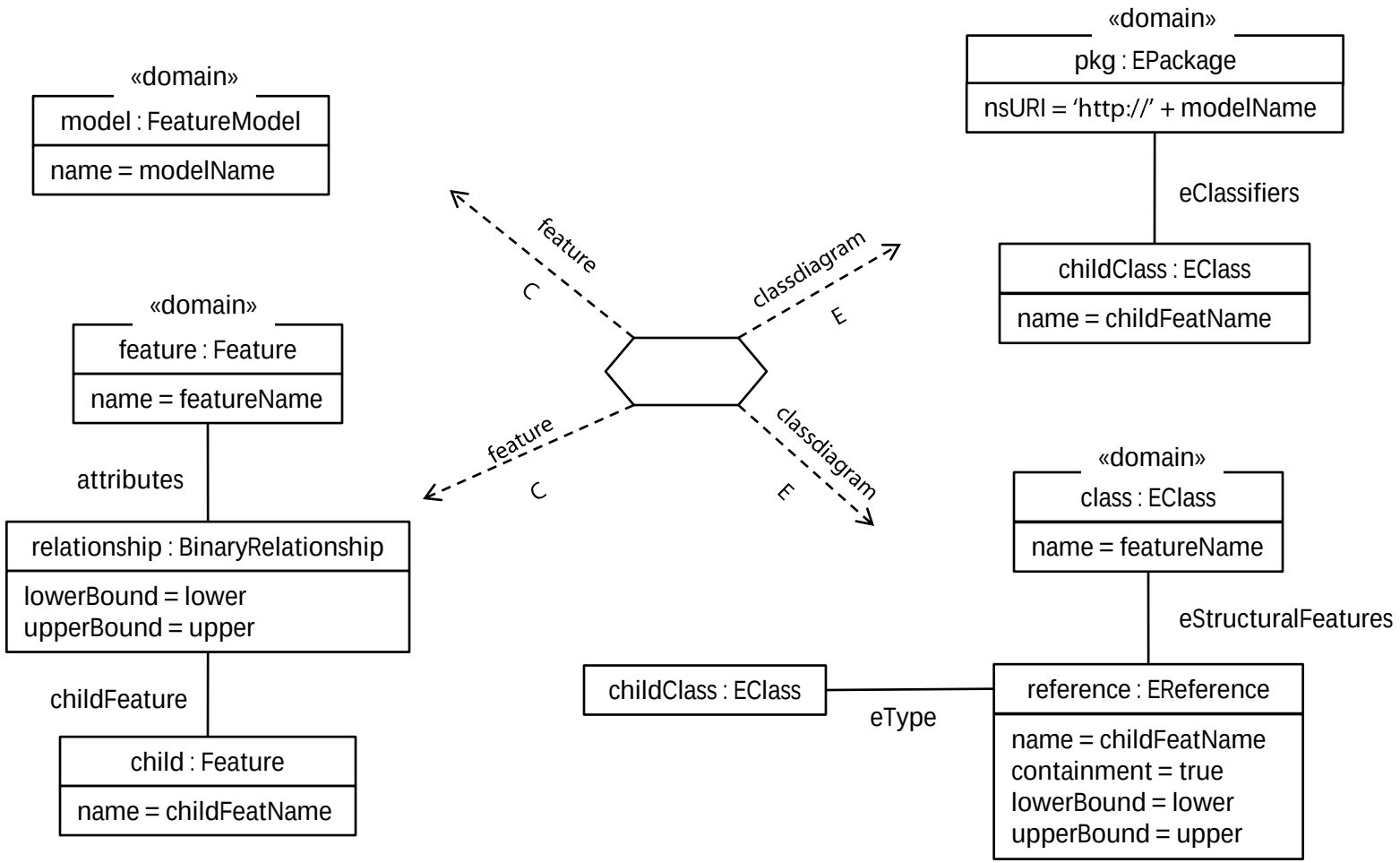


Relation SimpleRelationship2Reference

- Each *BinaryRelationship* is transformed to a UML composition (*containment EReference*).
- The type of relationship determines de multiplicity of the *EReference*.

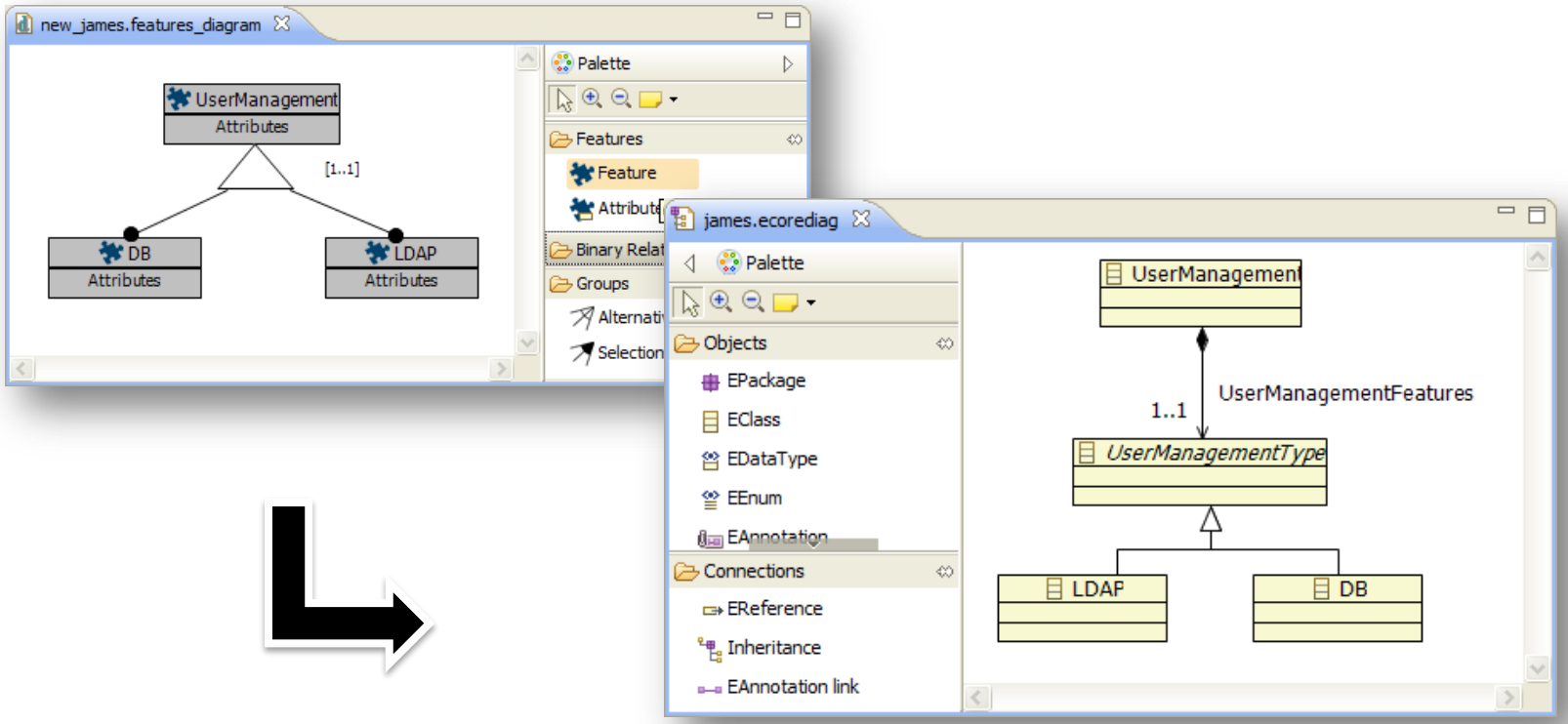


Relation SimpleRelationship2Reference

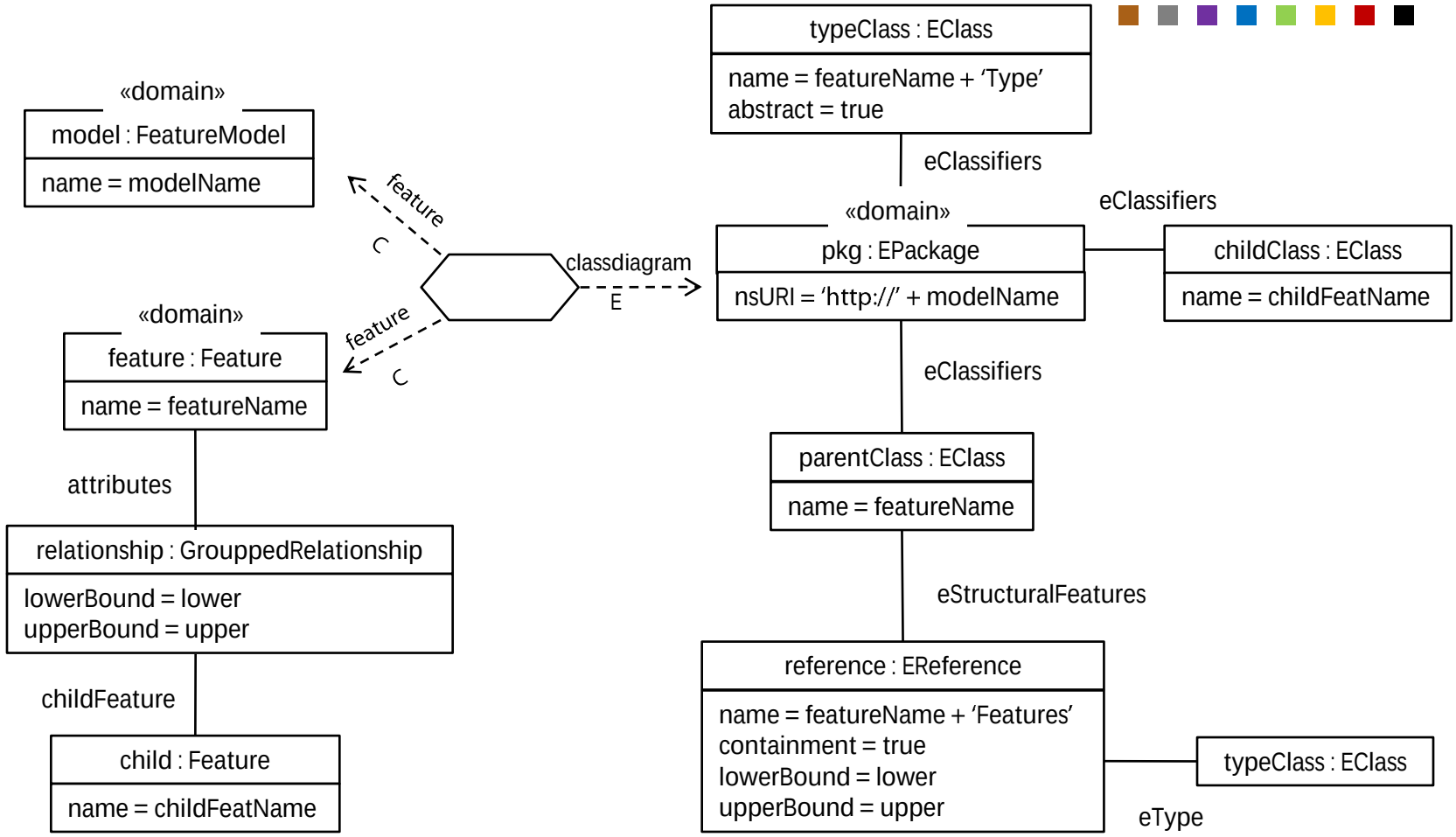


Relation MultipleRelationship2Reference

- Each *GrouppedRelationship* is transformed to a *containment EReference*.
- Each *child feature* inherits from a new class whose name derives from the parent feature. The inheritance relationship is set in the relation «*MultipleRelationshipChilds2Classes*»



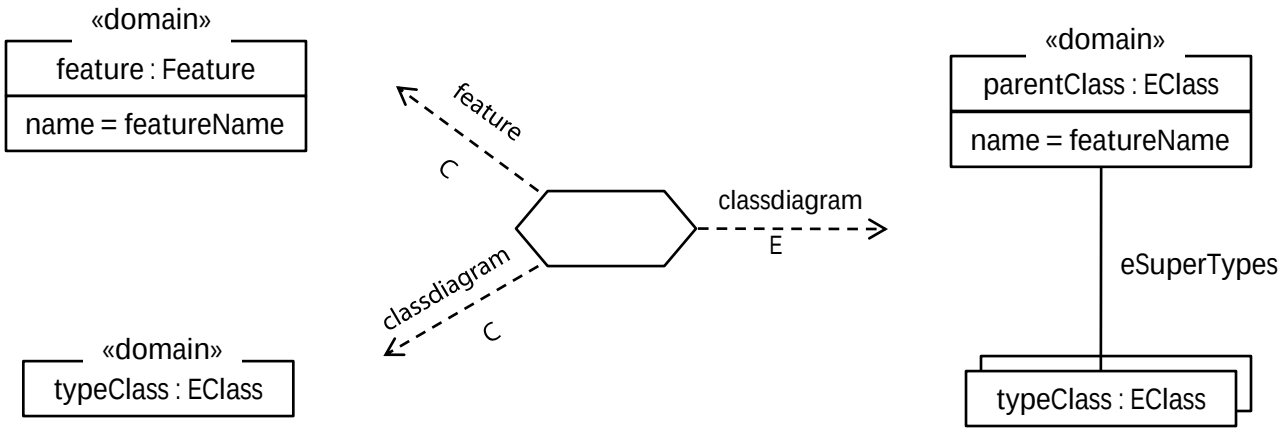
Relation MultipleRelationship2Reference



where

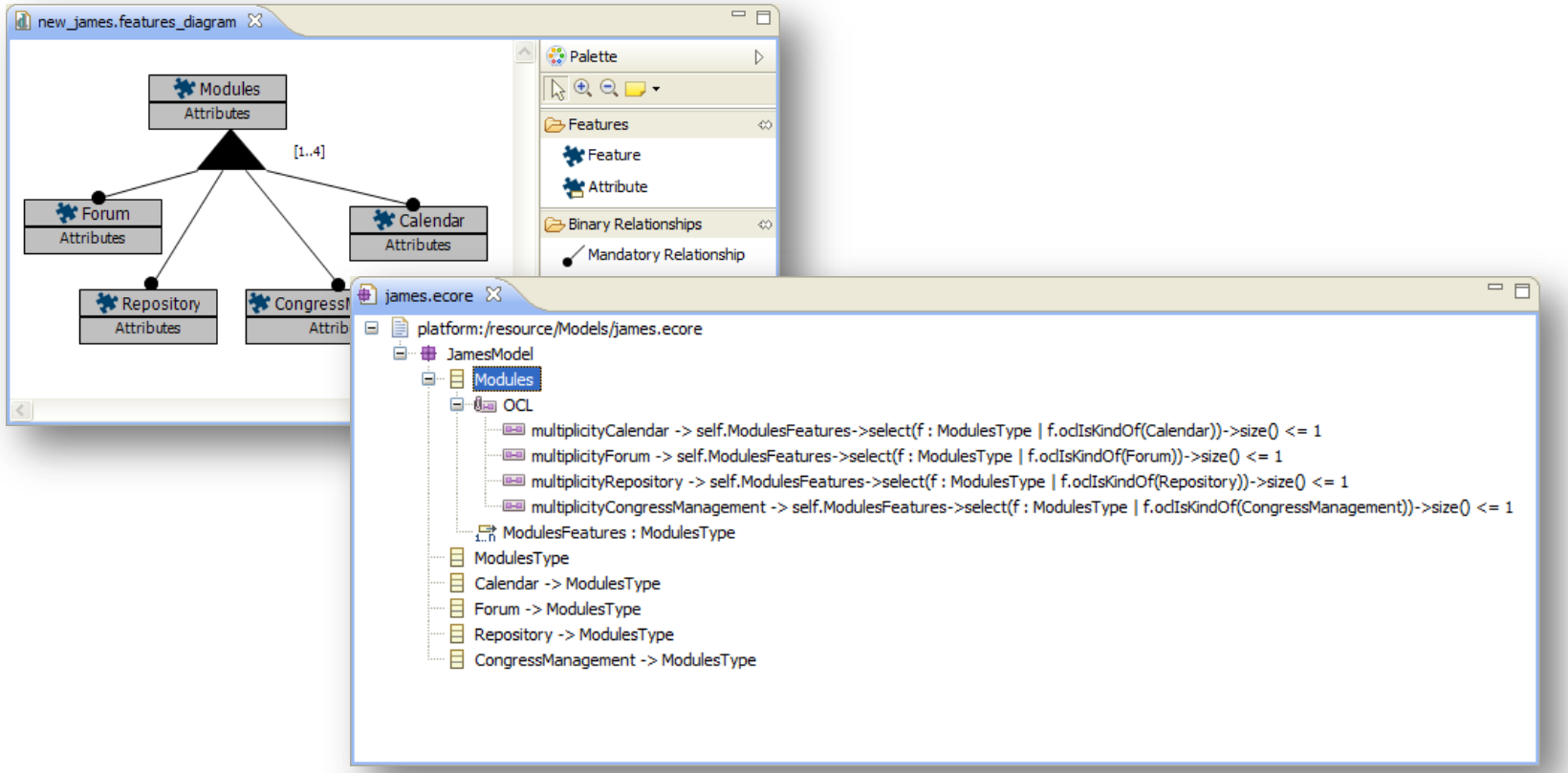
MultipleRelationshipChilds2Classes(childFeature, typeClass, childClass);
MultipleRelationshipChilds2Annot(childFeature, featureName, parentClass);

Relation MultipleRelationshipChilds2Classes

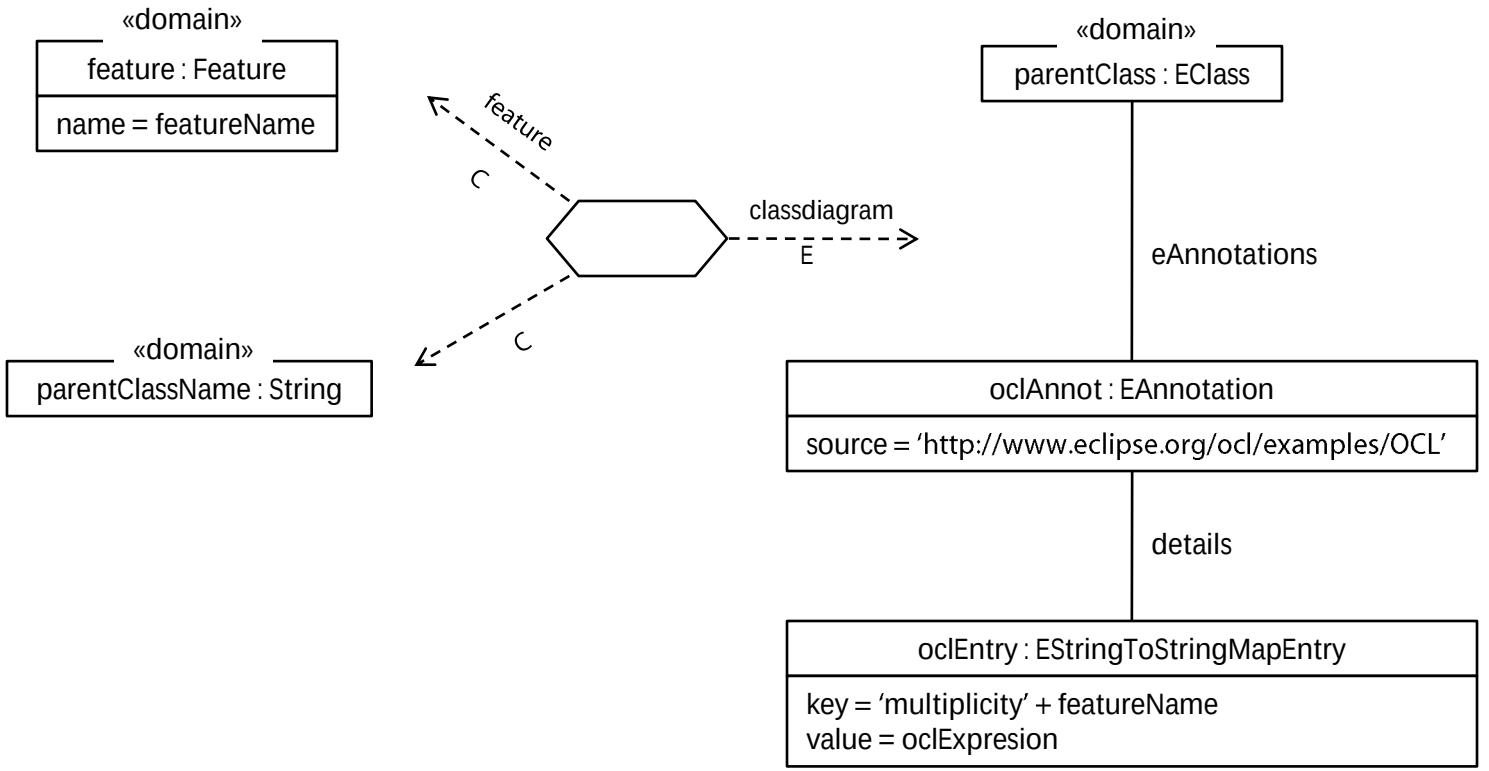


Relation MultipleRelationshipChilds2Annot

- When the *upperBound* of the *GrouppedRelationship* is greater than 1, we can automatically generate OCL invariants to avoid duplicated elements in the feature model instance.



Relation MultipleRelationshipChilds2Annot

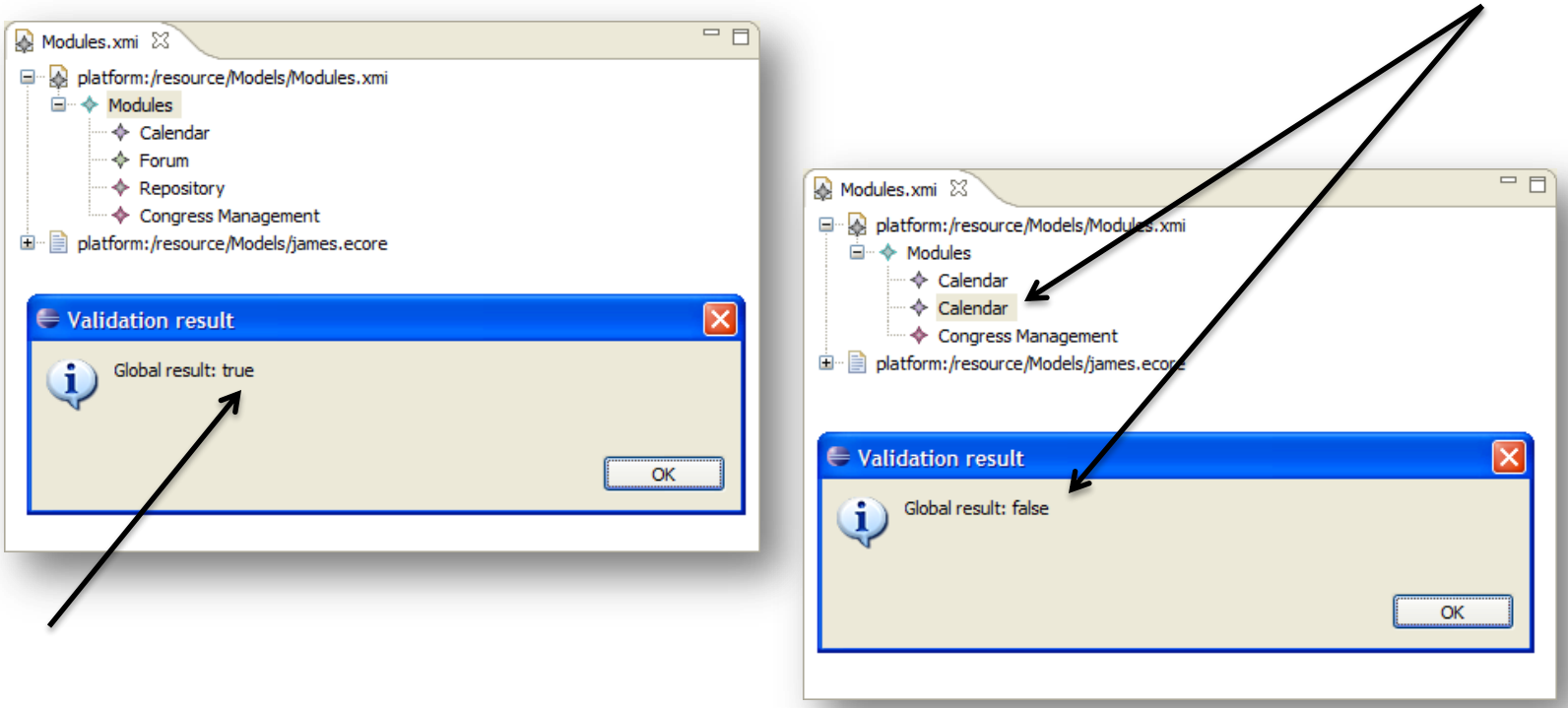


where

oclExpression =
'self.' + parentClassName + 'Features->select(f : ' + parentClassName + 'Type | f.oclIsKindOf(' + featureName + '))->size() <= 1'

Relation MultipleRelationshipChilds2Annot

- These OCL invariants can be checked automatically. In the example, the screenshot on the right has a duplicated module (Calendar).



- Moreover, any OCL invariant can be defined in order to check the consistency of the feature model instance.

Representing feature models as class diagrams

Abel Gómez, Isidro Ramos
Technical University of Valencia.
Departamento de Sistemas Informáticos y Computación.