Métamodelisation en environnement Eclipse : EMF

Julien Deantoni

Principes (caricature)

- **Modéliser :**
    - Représenter un système (communiquer, analyser, générer du code, documenter, …)

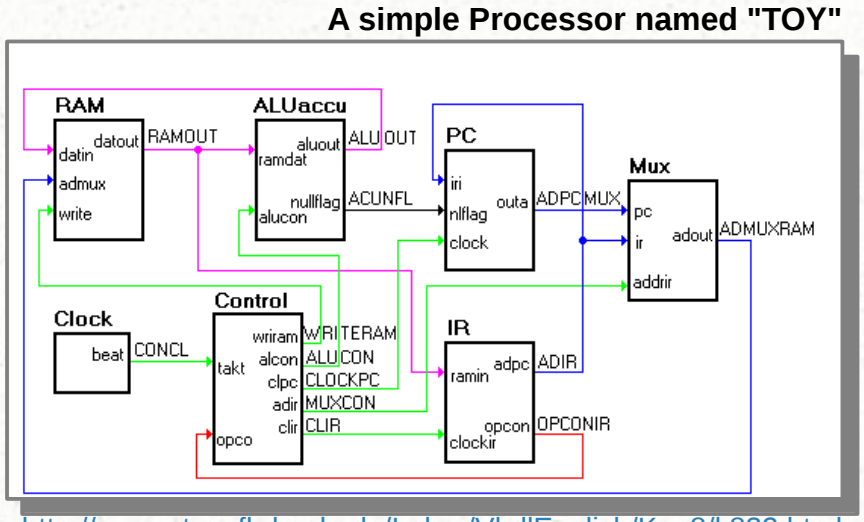
  Pourquoi : Pour abstraire un système (selon le but de notre modèle l'abstraction sera différente)

- **Métamodéliser :**
    - Représenter un domaine (architecture, télécommunication, robotique, …)

  Pourquoi : Pour faciliter la manipulation de modèles (éditeurs, transformations, ...)

Principes (caricature)

- Modéliser :



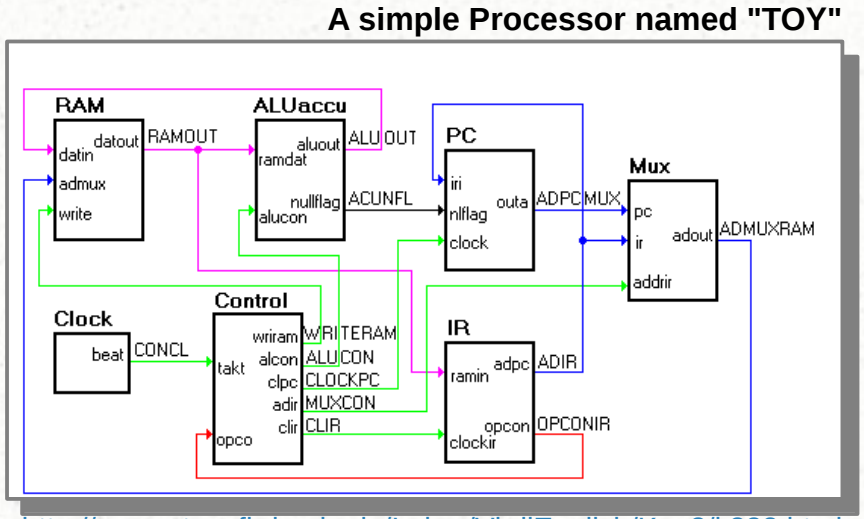
http://www.nt-nv.fh-koeln.de/Labor/VhdlEnglish/Kap8/k832.html

- Métamodéliser :
  - Représenter un domaine (architecture, télécommunication, robotique, …)

Pourquoi : Pour faciliter la manipulation de modèles (éditeurs, transformations, ...)

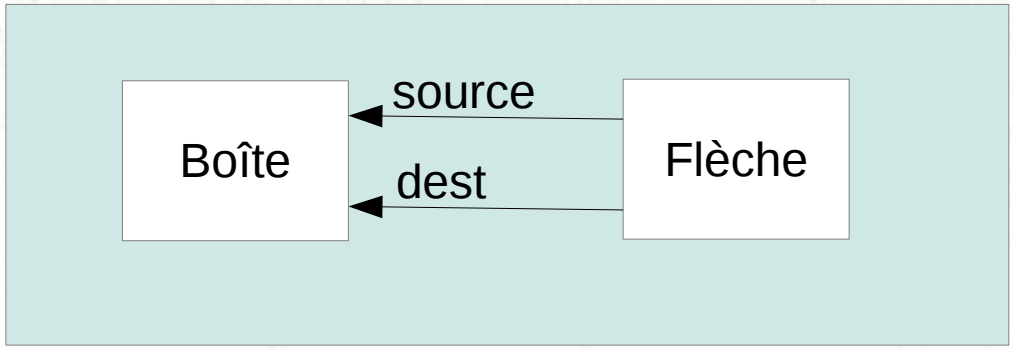
Principes (caricature)

- Modéliser :



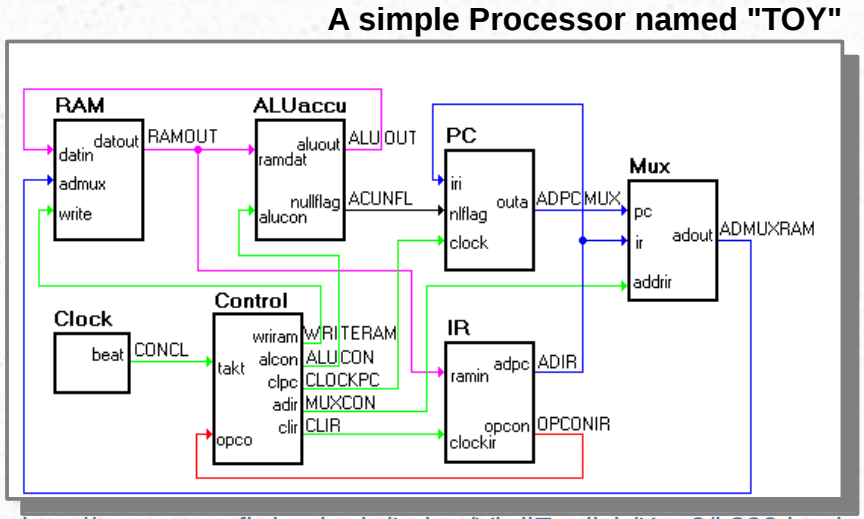
http://www.nt-nv.fh-koeln.de/Labor/VhdlEnglish/Kap8/k832.html

- Métamodéliser :



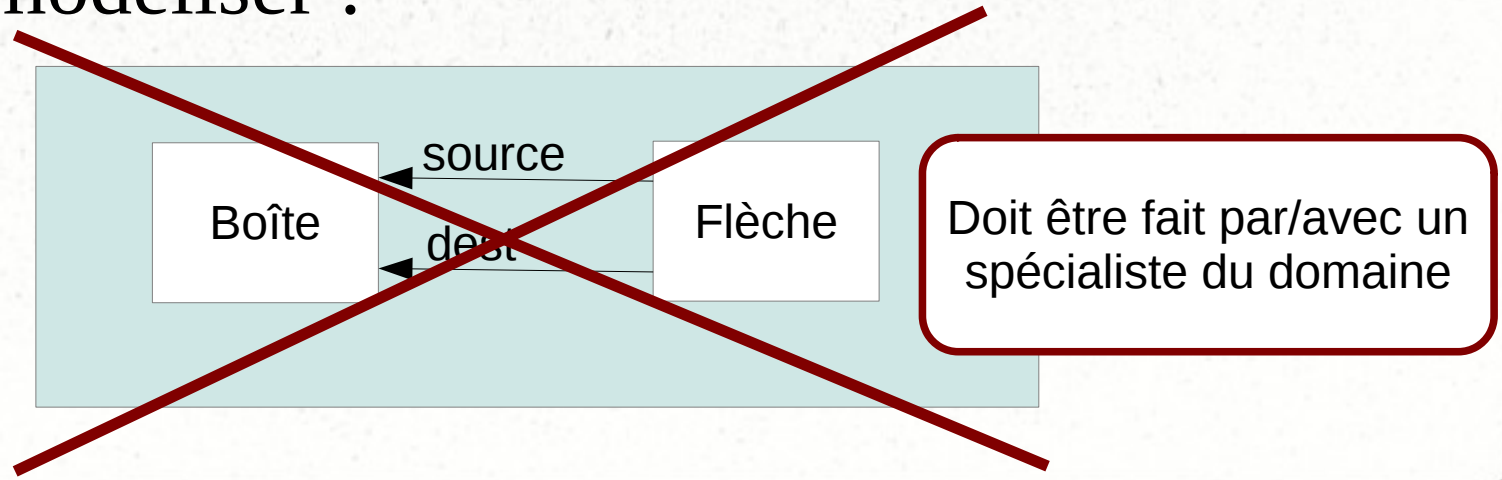
Principes (caricature)

- Modéliser :



http://www.nt-nv.fh-koeln.de/Labor/VhdlEnglish/Kap8/k832.html

- Métamodéliser :



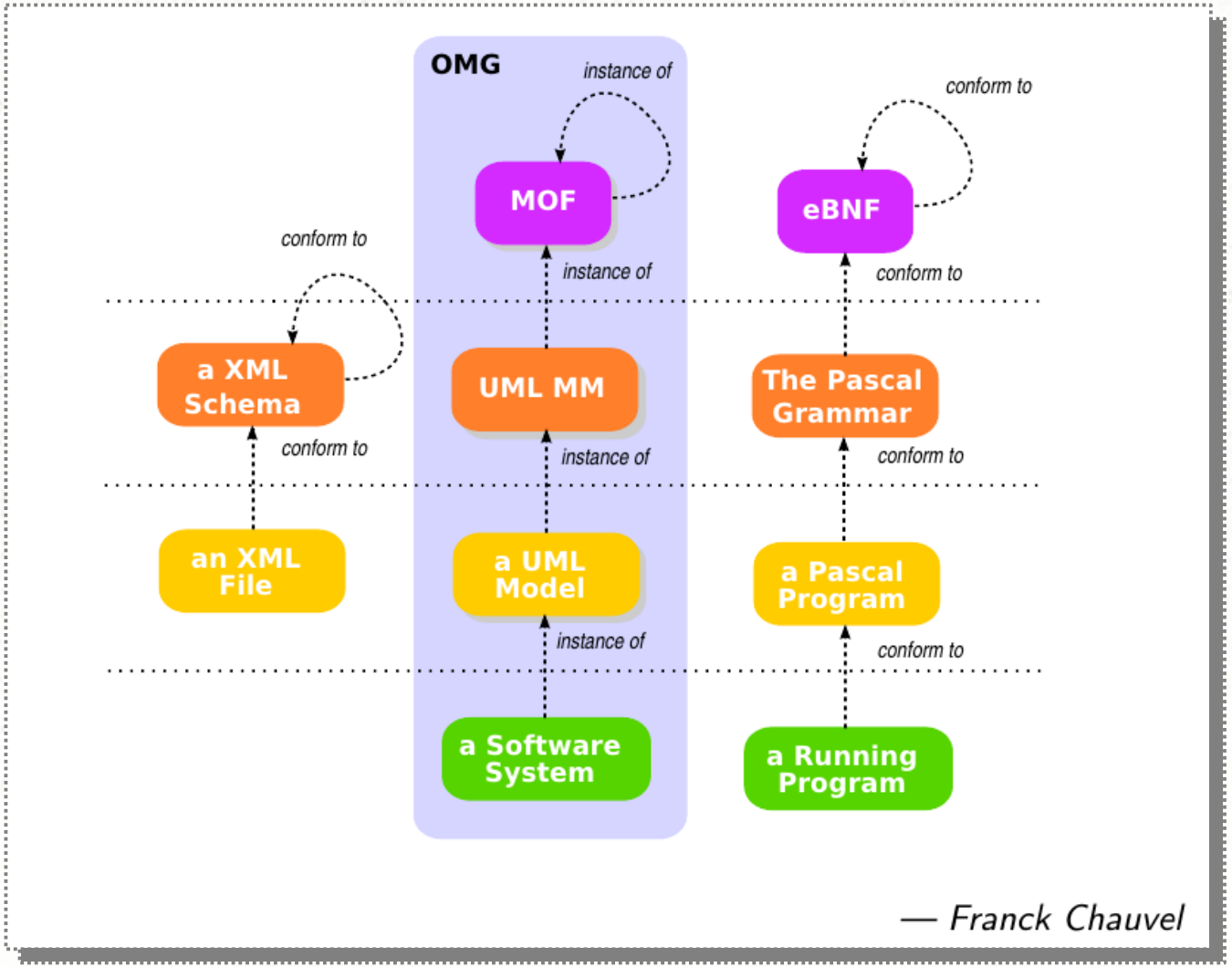
Modèle et métamodèle

- Un modèle est
  - Une instance d'un métamodèle
  - Conforme à son métamodèle
  - Une représentation syntaxique de l'entité modélisée
    (proche de la notion d'AST (Abstract Syntax Tree))

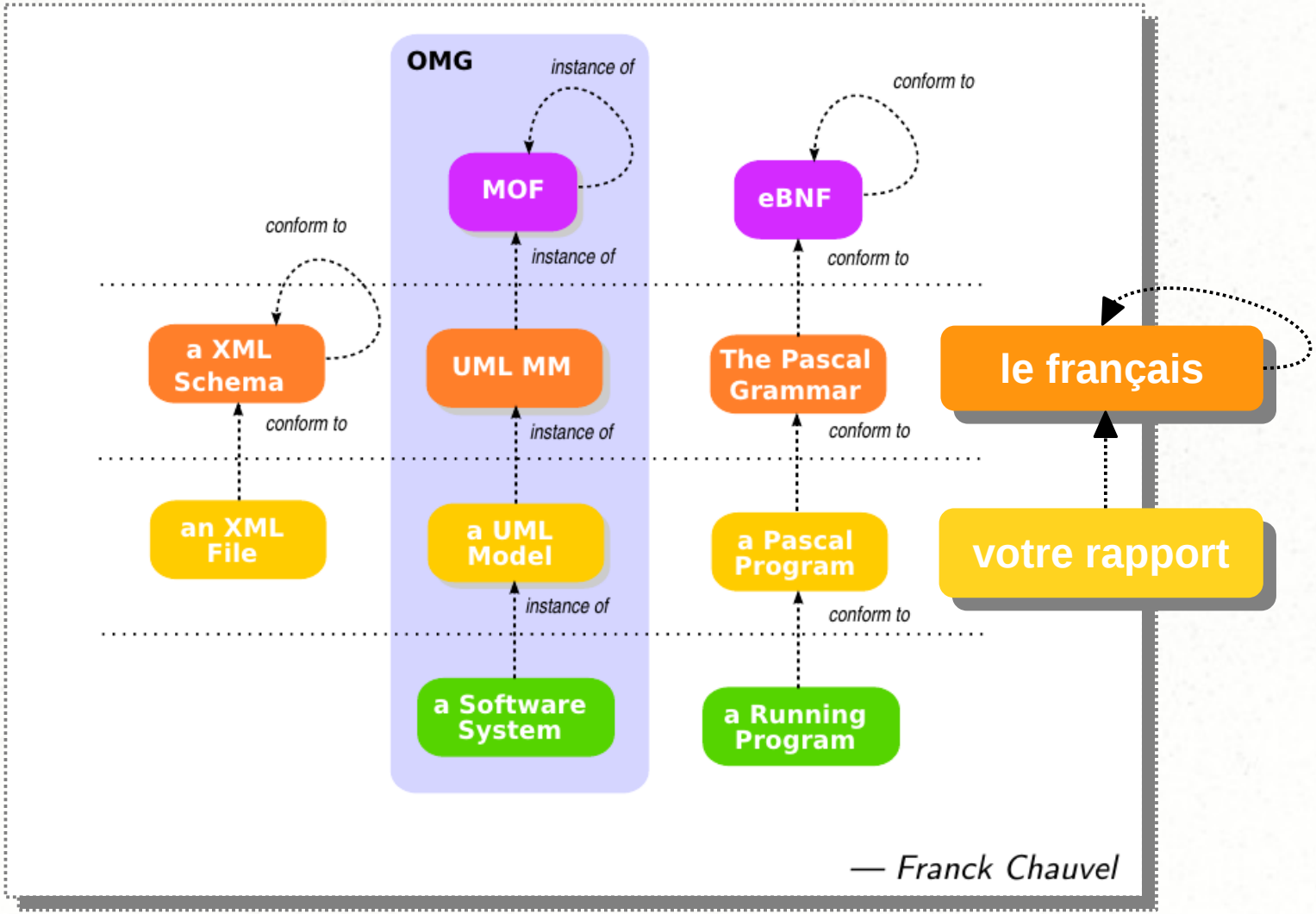
Modèle et métamodèle

- Un modèle est
  - Une instance d'un métamodèle
  - Conforme à son métamodèle
  - Une représentation syntaxique de l'entité modélisée (proche de la notion d'AST (Abstract Syntax Tree))
- Un métamodèle est
  - Un modèle
  - La définition des concepts et des relations des instances qui lui sont conformes (proche de la définition d'une grammaire)

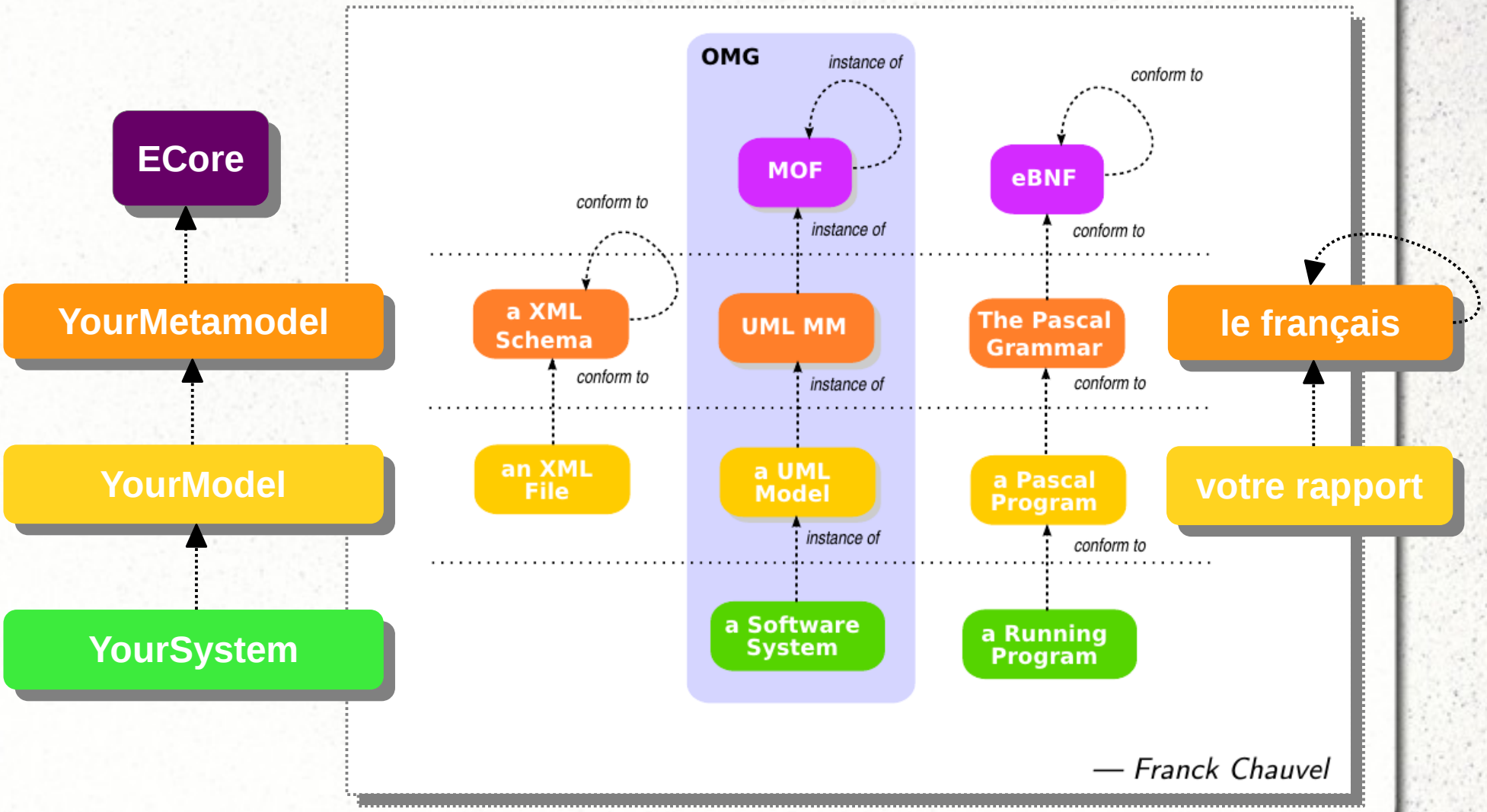
Modèle et métamodèle



Modèle et métamodèle

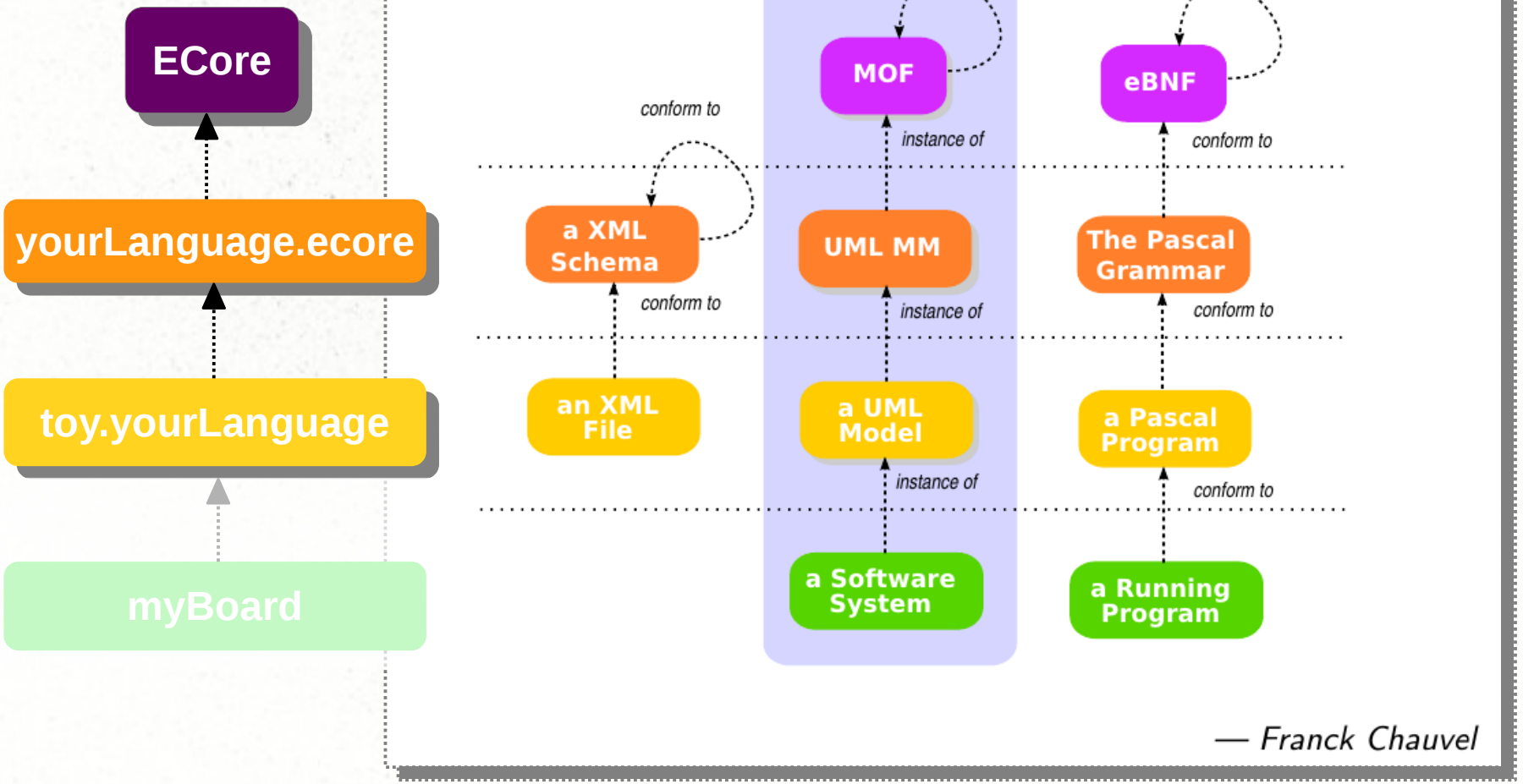


Eclipse Modeling Framework

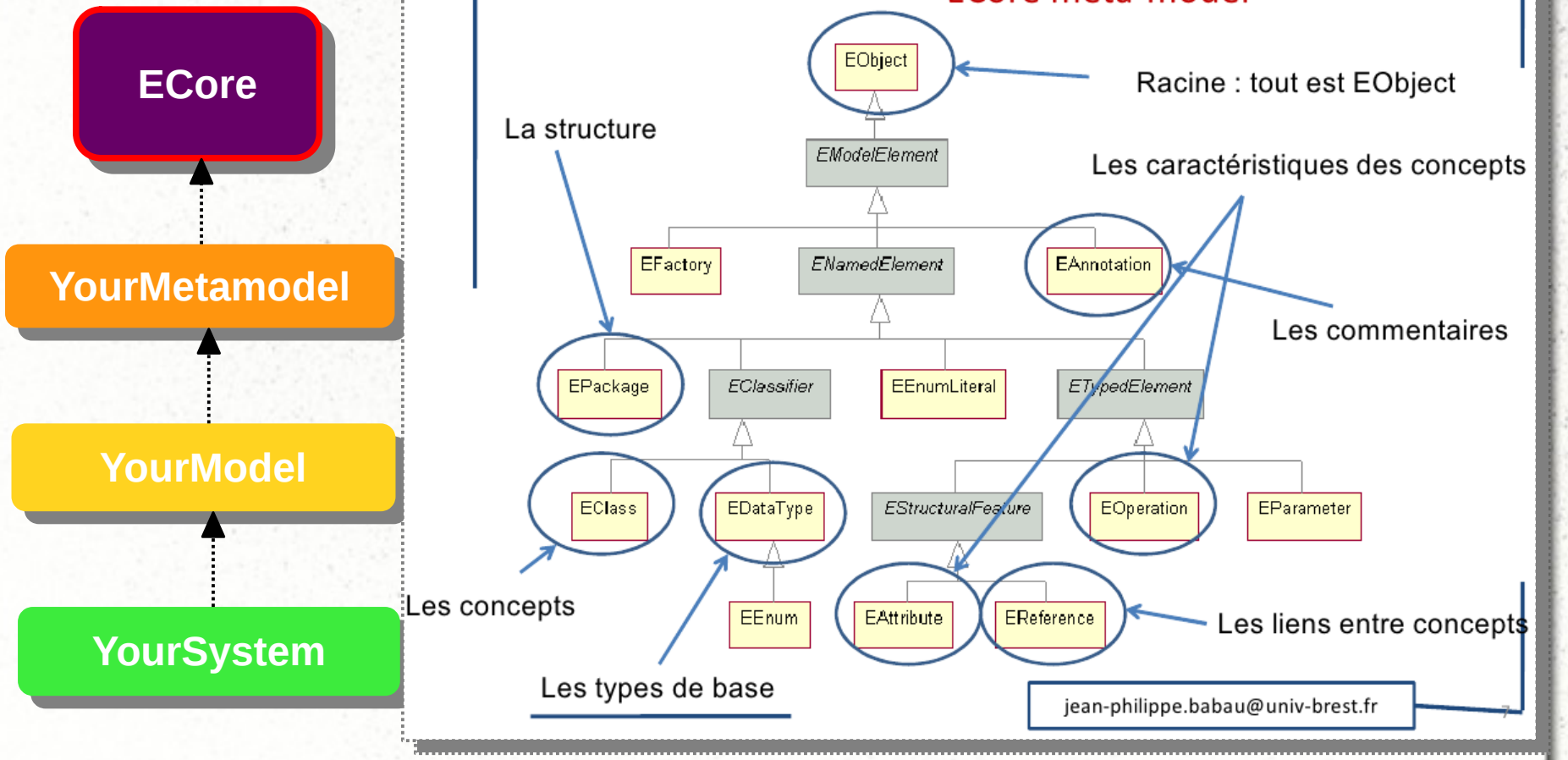


Eclipse Modeling Framework

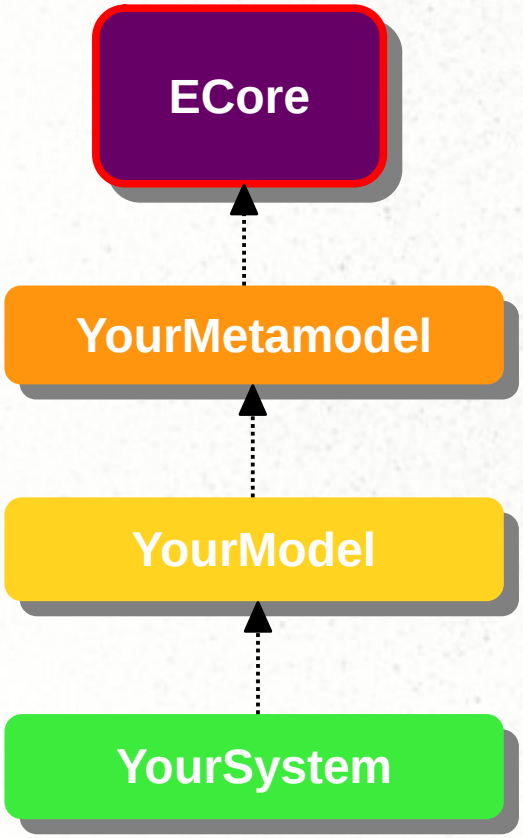
Dans quelques heures ?



Eclipse Modeling Framework



Eclipse Modeling Framework



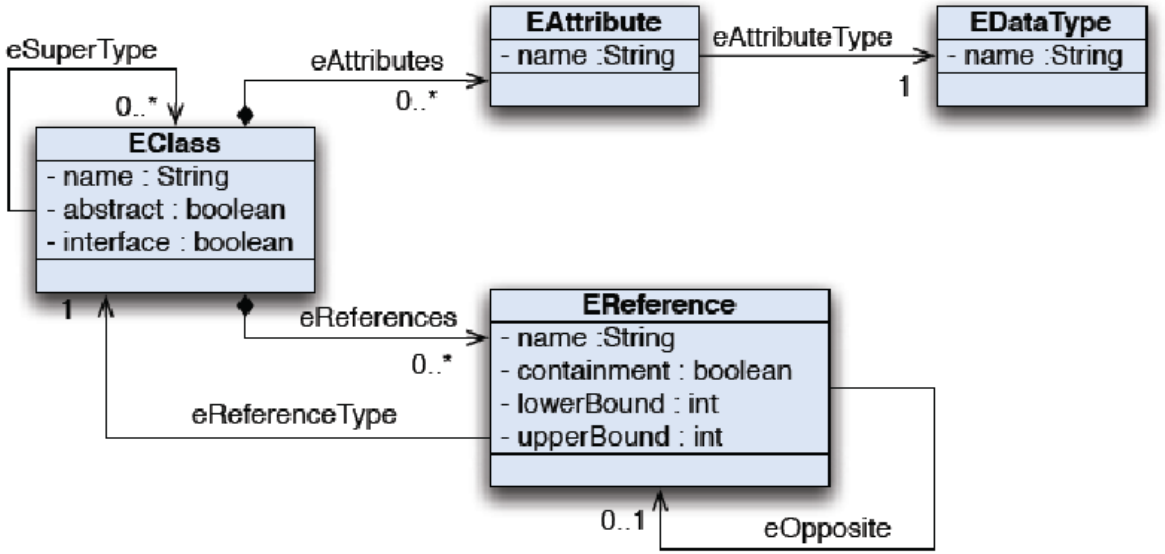
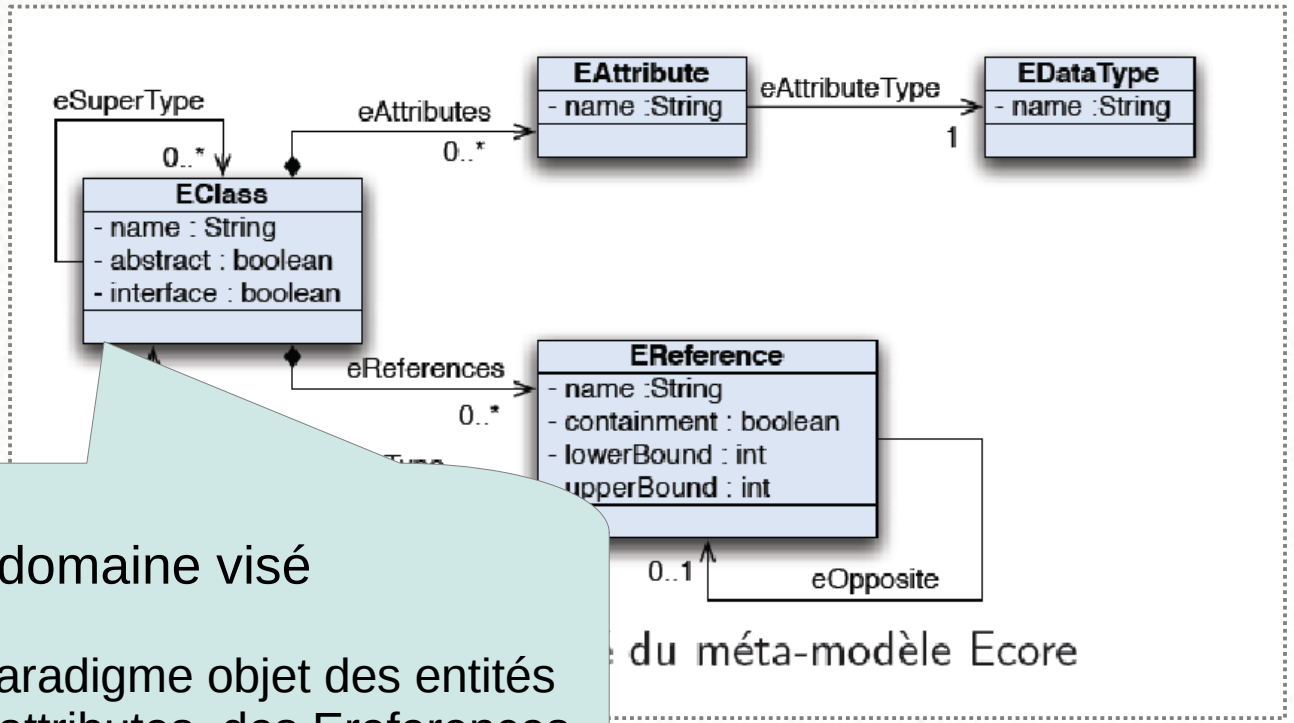


FIG.: Extrait simplifié du méta-modèle Ecore – *Sébastien Mosser*

Eclipse Modeling Framework

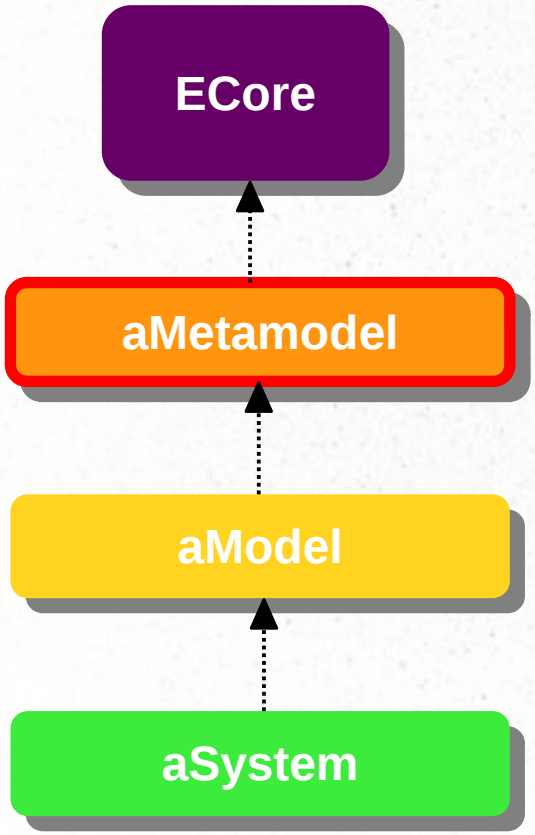
ECore

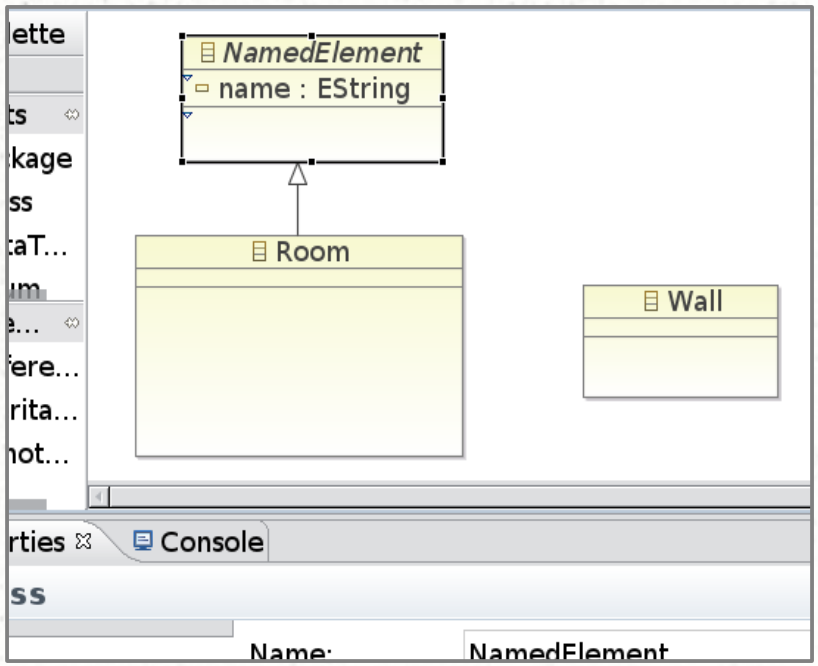


Modélise un concept du domaine visé

- Modélisation selon le paradigme objet des entités
- Caractérisée par des Eattributes, des Ereferences
  (et des EOperations)

Eclipse Modeling Framework





Eclipse Modeling Fr

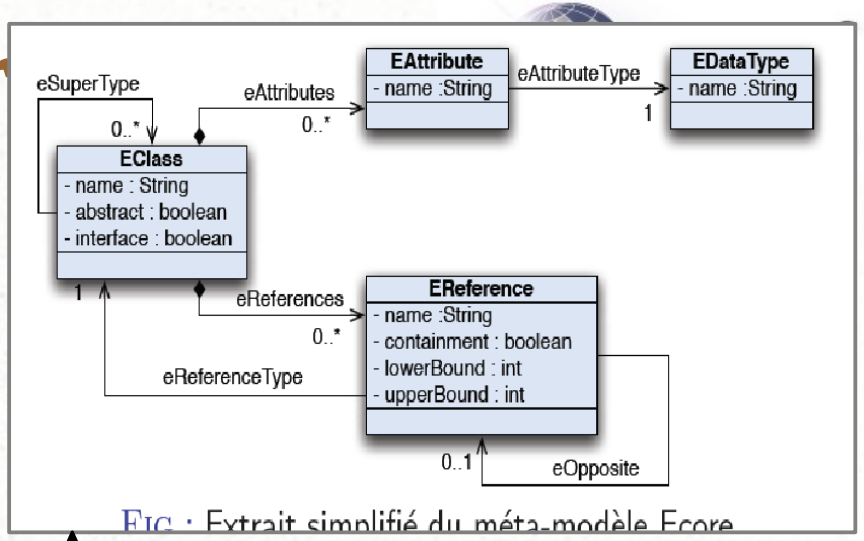


Fig. : Extrait simplifié du méta-modèle Ecore

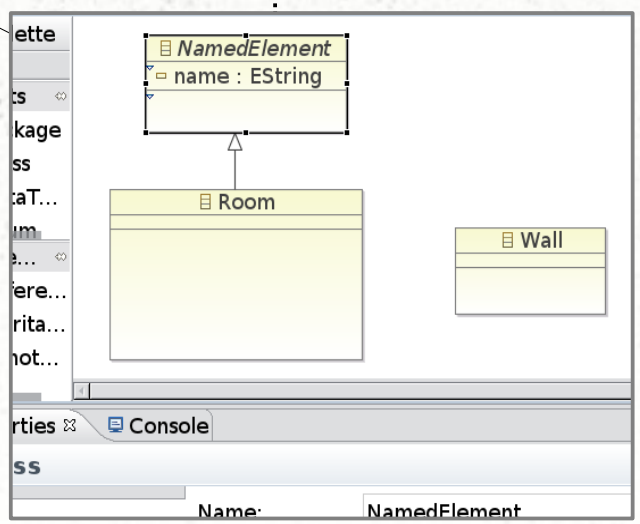
ECore

aMetamodel

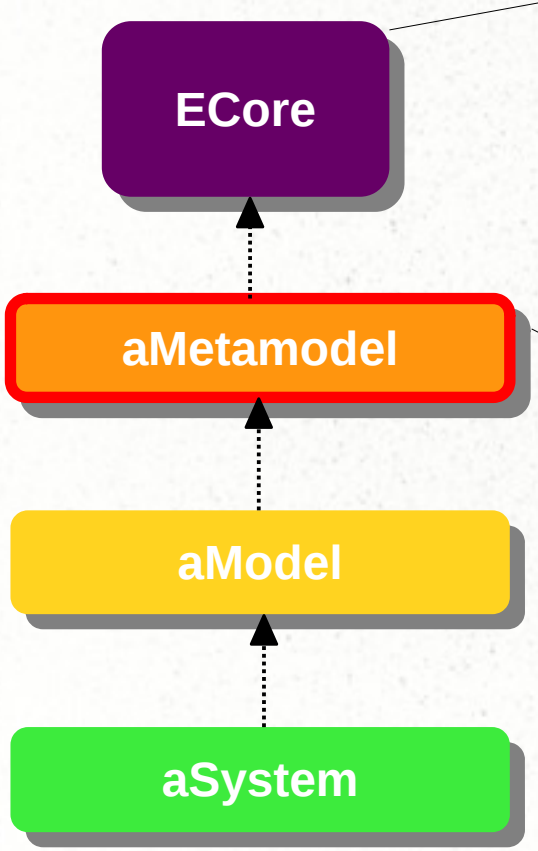
aModel

aSystem

conformsTo



Eclipse Modeling Fr



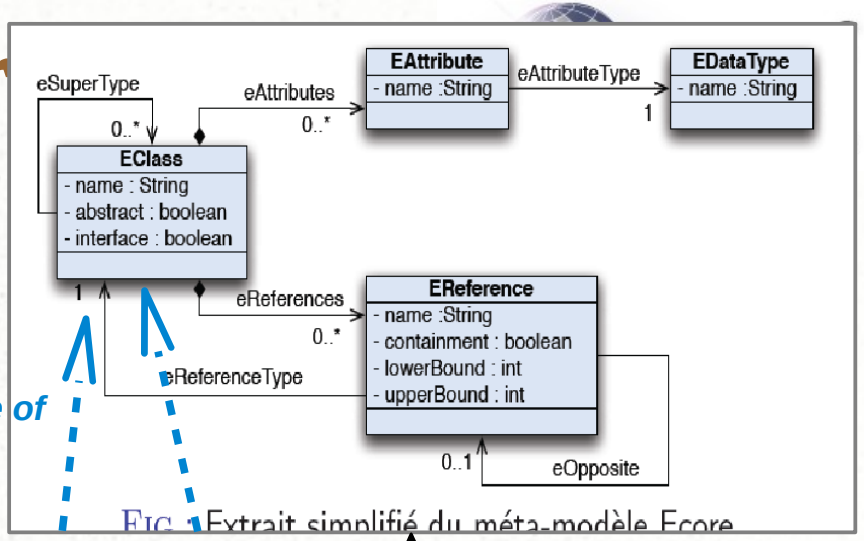
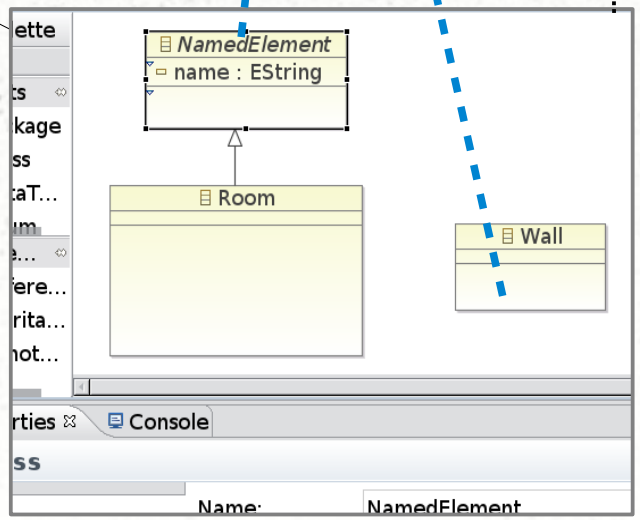


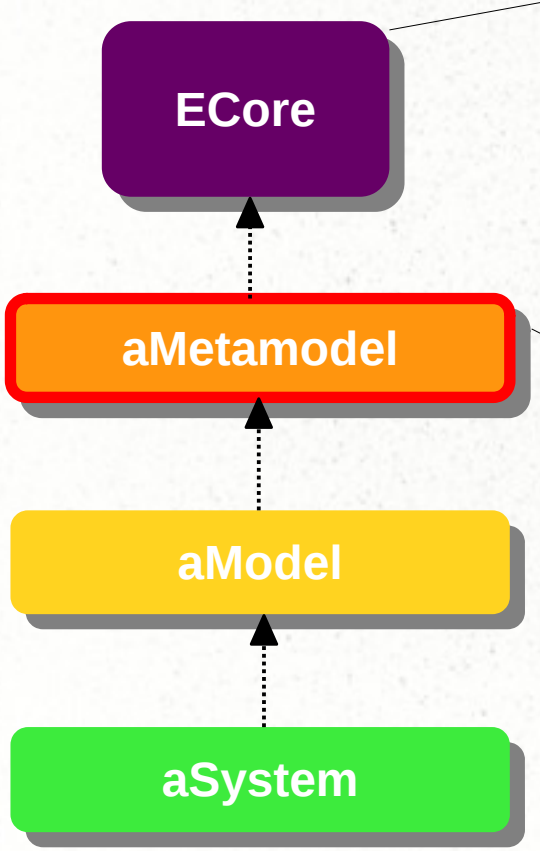
Fig. : Extrait simplifié du méta-modèle Ecore

Instance of

conformsTo



Eclipse Modeling Fr



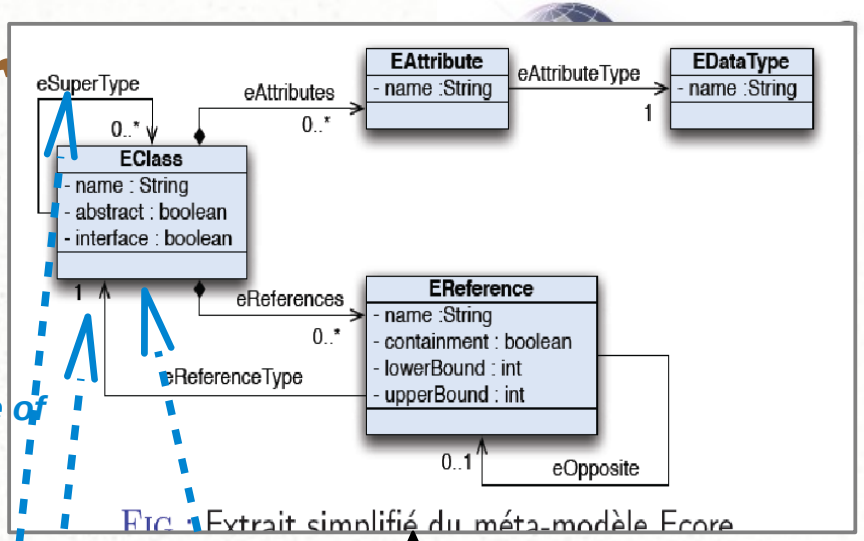
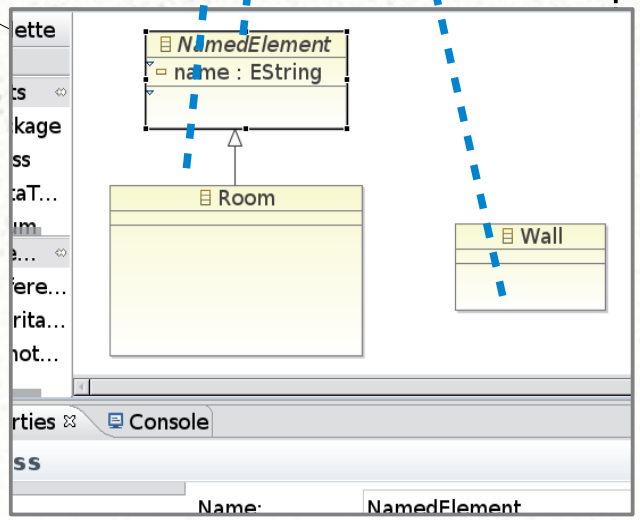


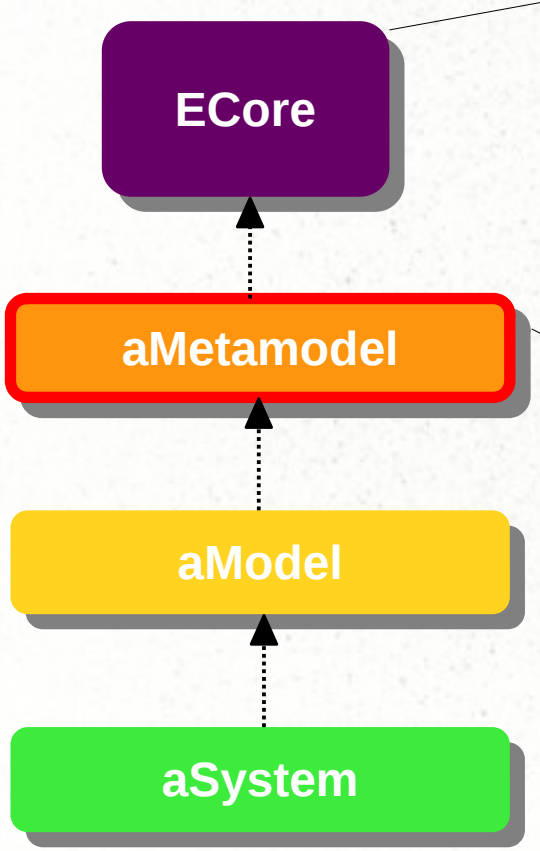
Fig. : Extrait simplifié du méta-modèle Ecore

Instance of

conformsTo



Eclipse Modeling Fr



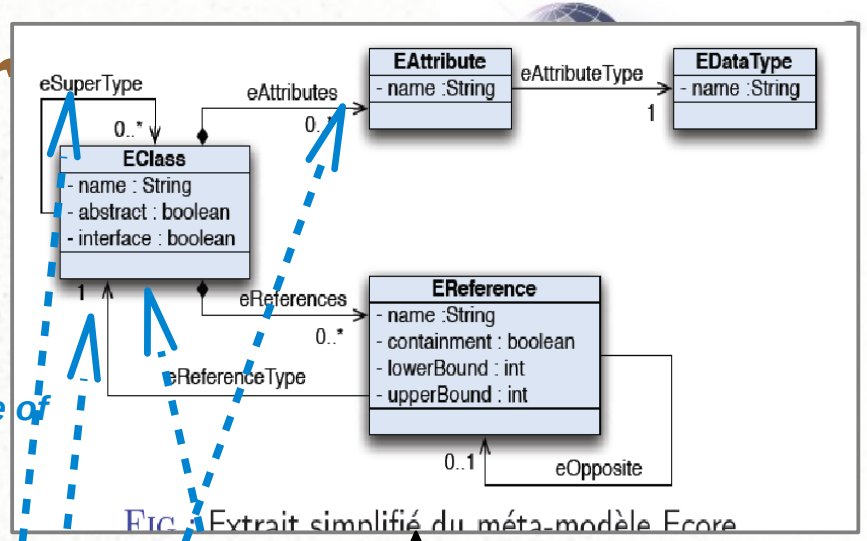
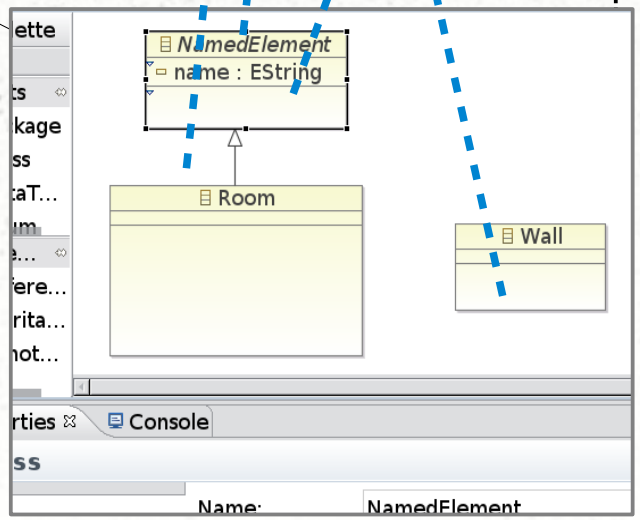


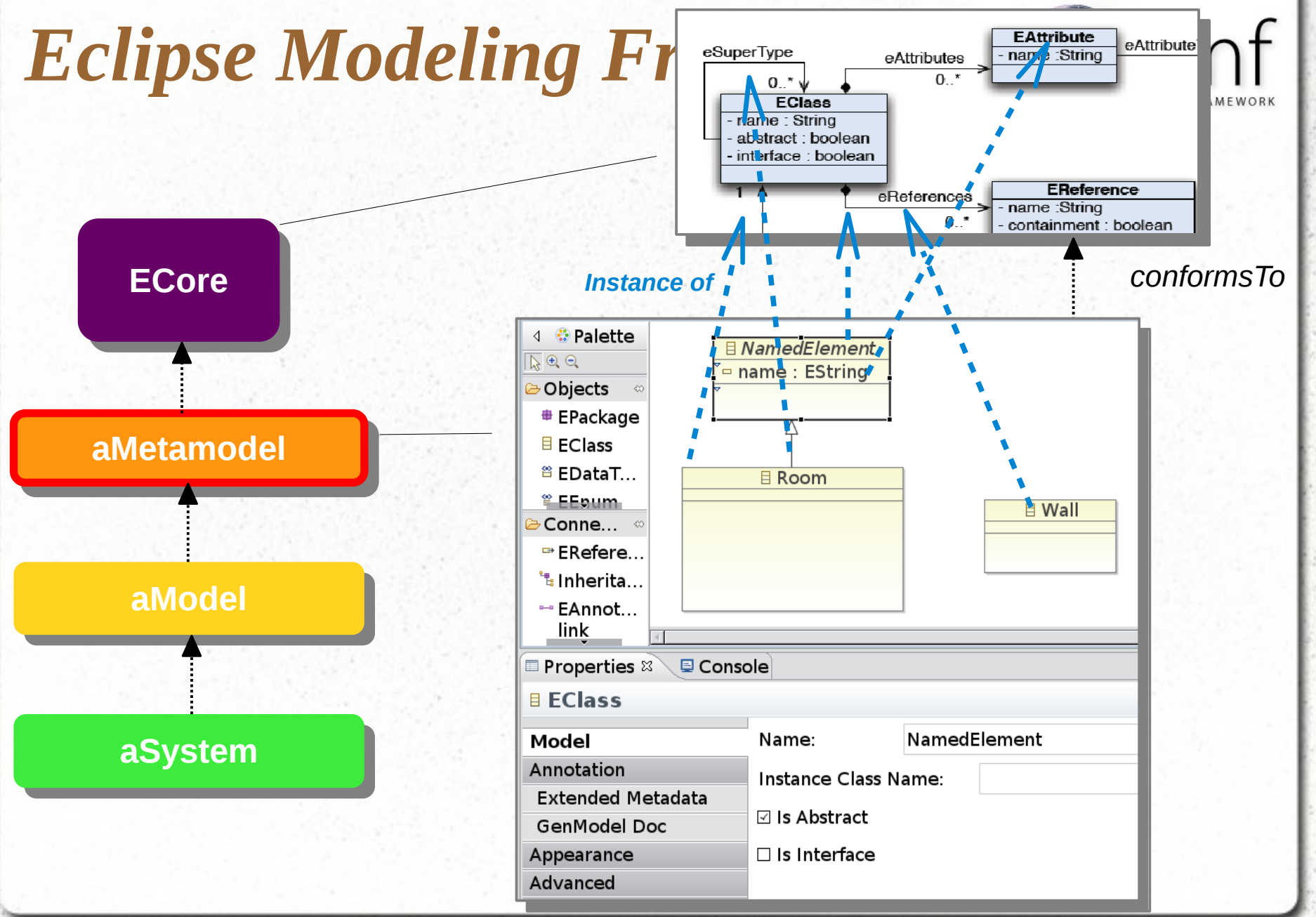
Fig. : Extrait simplifié du méta-modèle Ecore

Instance of

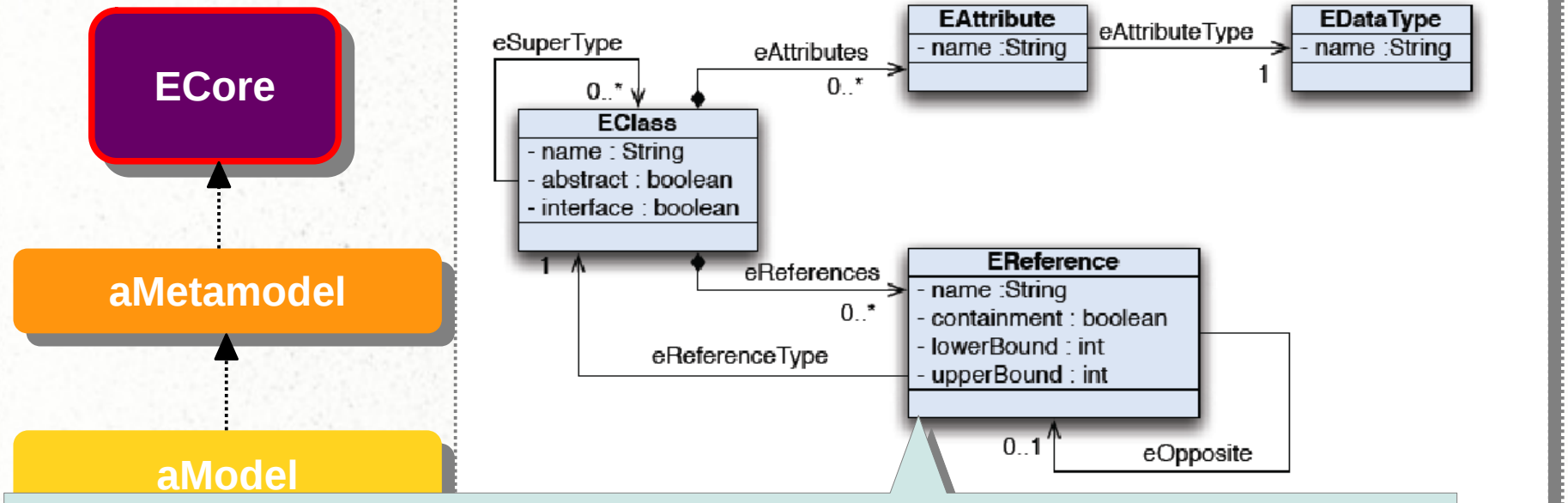
conformsTo



Eclipse Modeling Fr



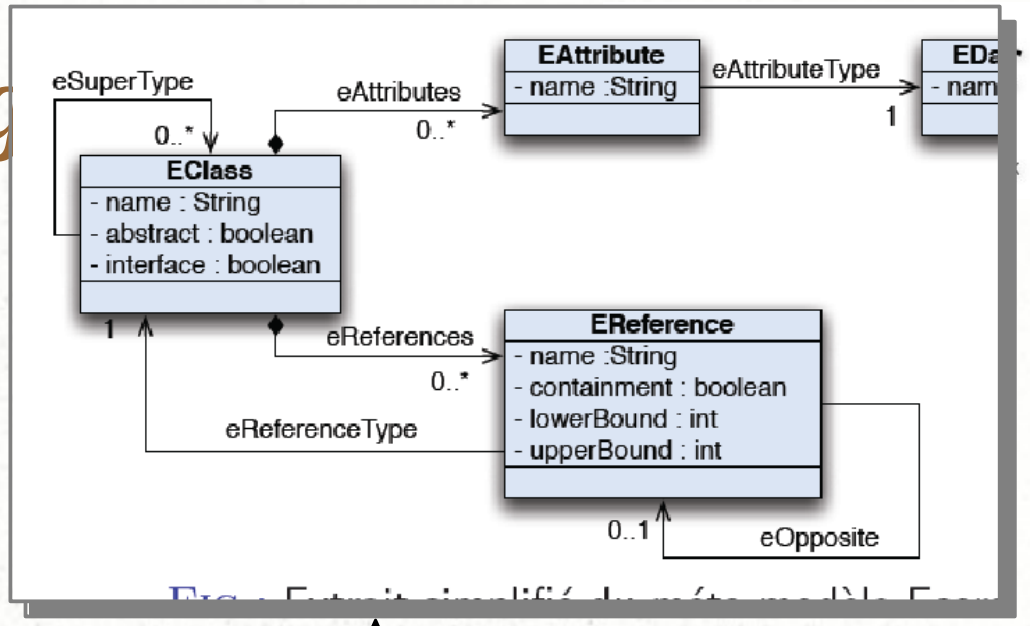
Eclipse Modeling Framework

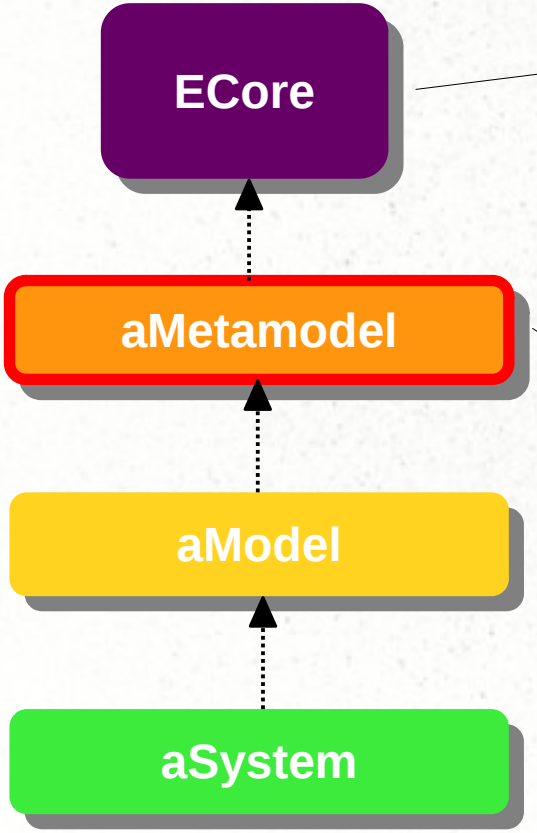


Lien entre concept

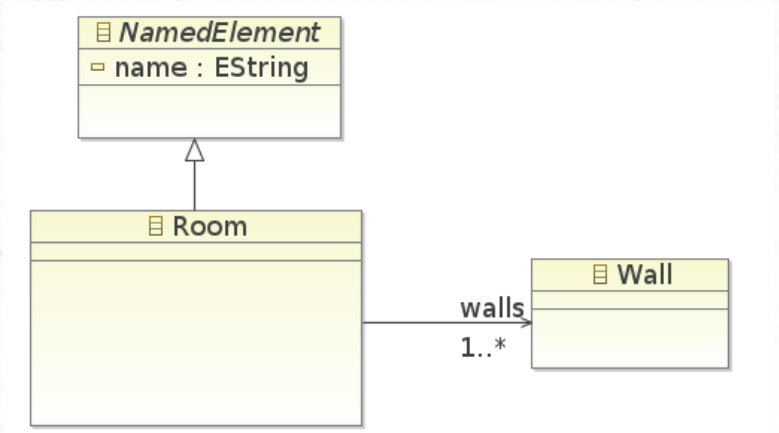
- Si “containment” est vrai alors equivalent à EAttribute pour les types non primitifs, sinon association simple.
- Caractérisé par un nom et une arité
- Eopposite assure que les instance d'un côté et de l'autre “correspondent”

Eclipse Modeling

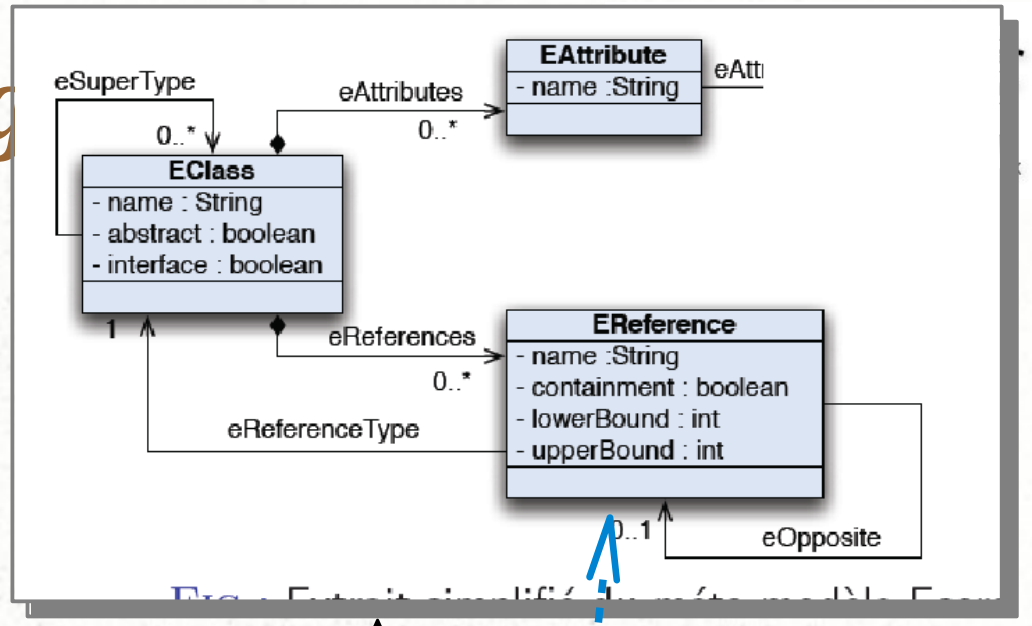


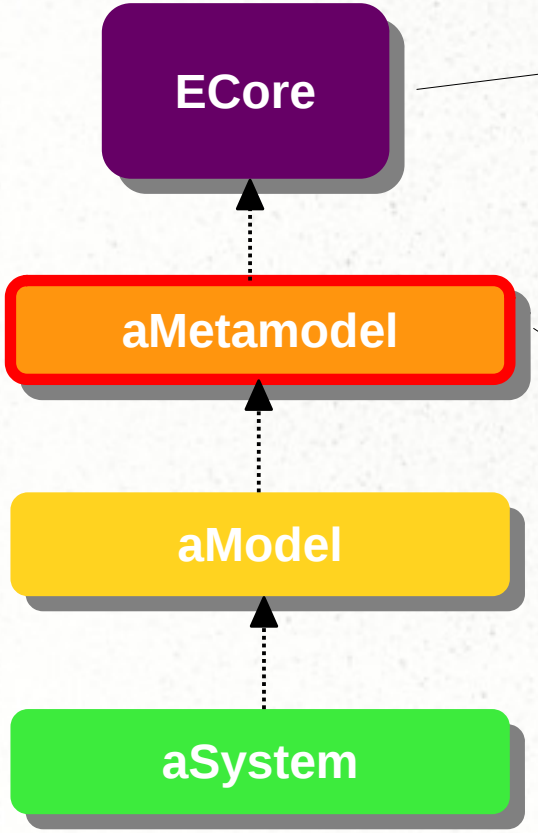


conformsTo

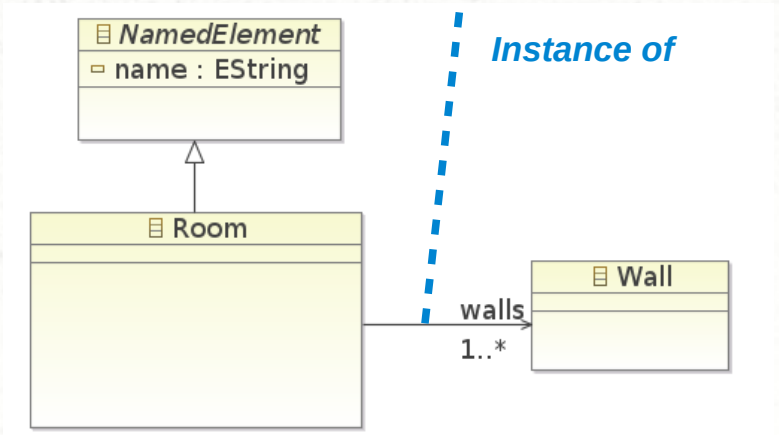


Eclipse Modeling

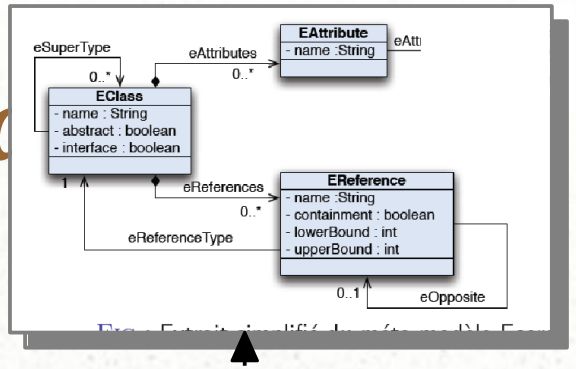




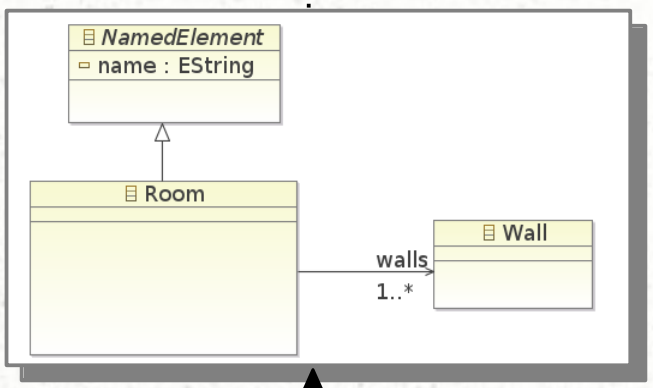
conformsTo



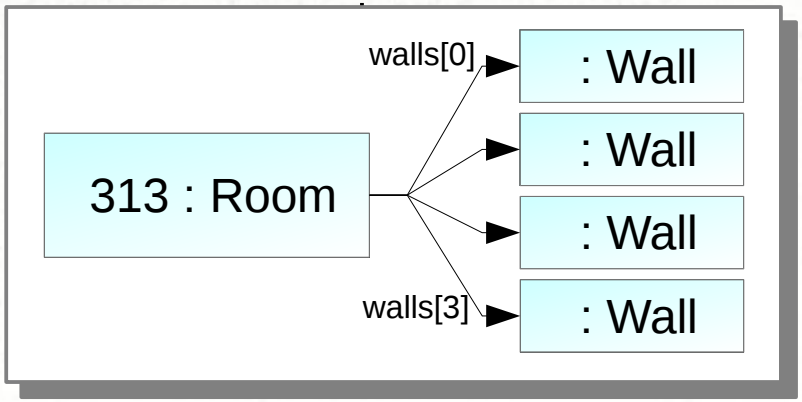
Eclipse Modeling Fr



conformsTo



conformsTo



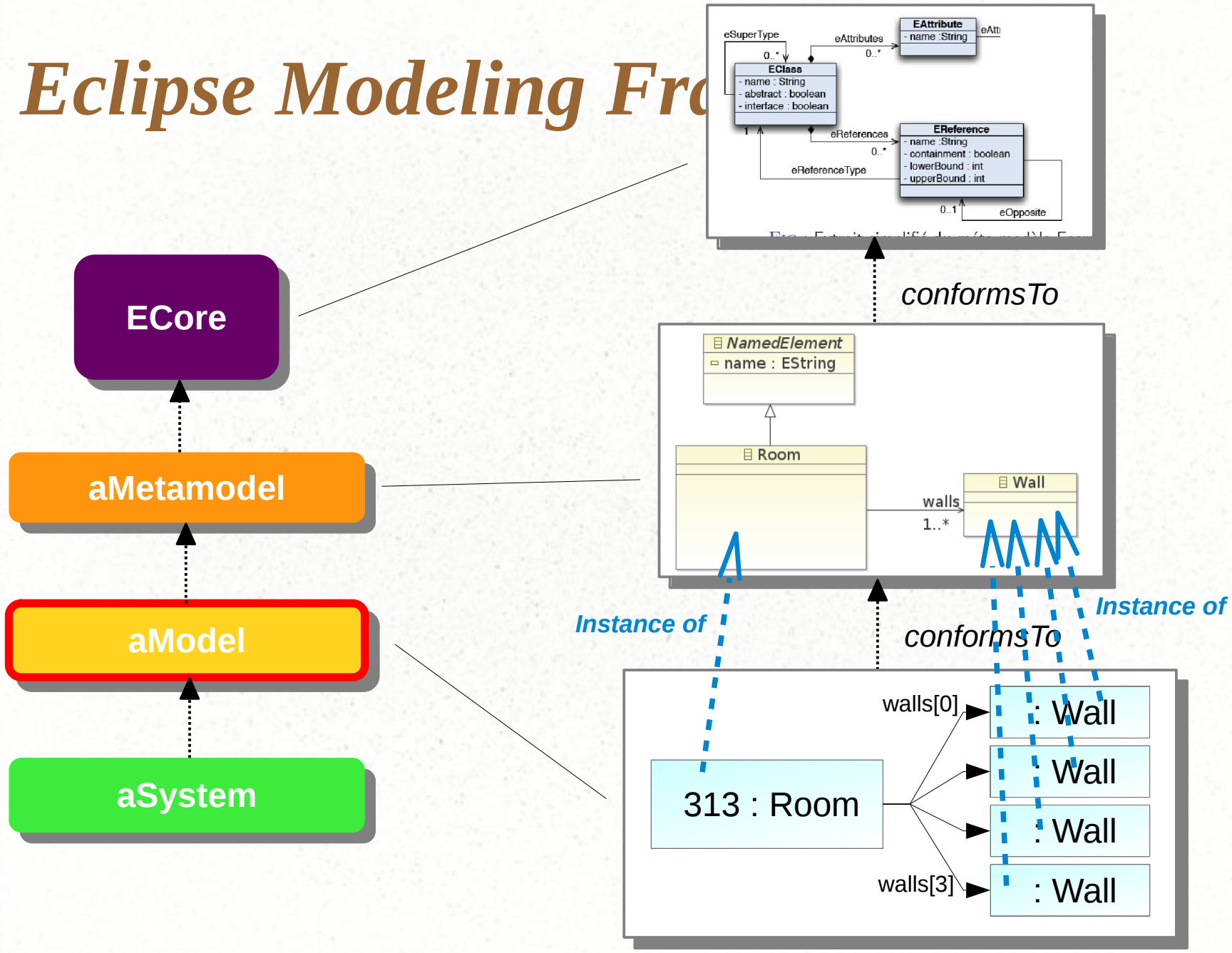
ECore

aMetamodel

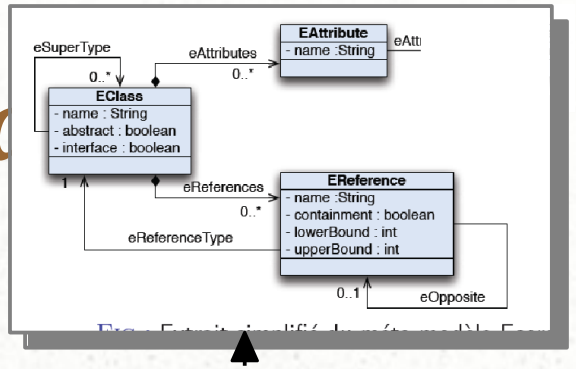
aModel

aSystem

Eclipse Modeling Fr



Eclipse Modeling Fr



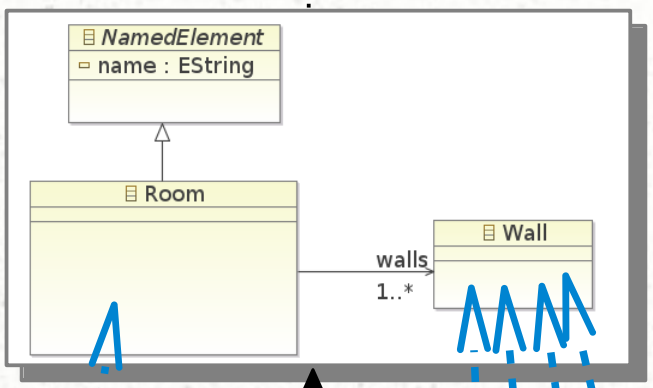
ECore

aMetamodel

aModel

aSystem

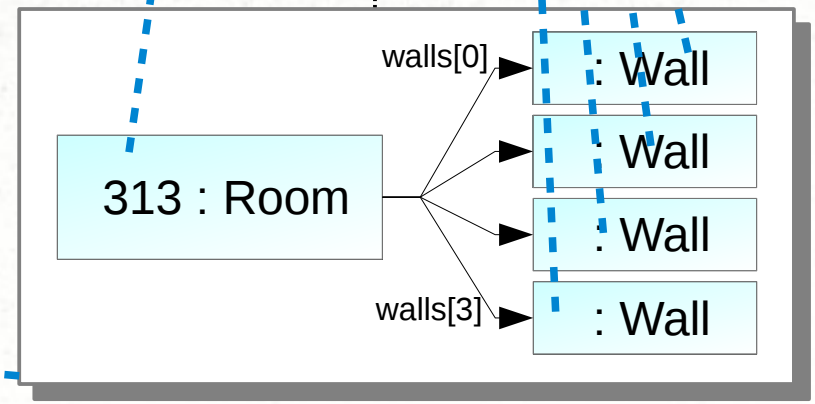
conformsTo



Instance of

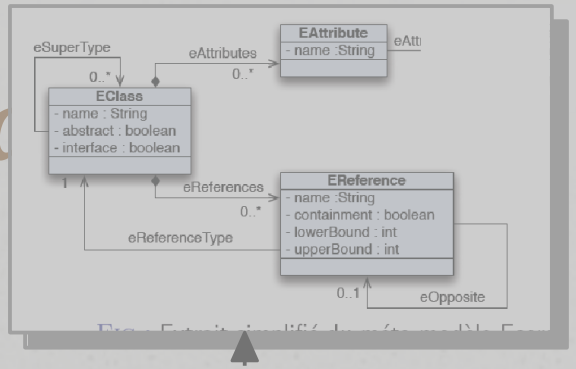
conformsTo

Instance of



Instance of ??

Eclipse Modeling Fr



ECore

aMetamo

aMod

aSystem

Contraintes d'un métamodèle:

- Il doit toujours y avoir (au moins) une EClass racine !
- À partir de cette racine, il doit exister un chemin de “containment” vers toutes les classes concrètes.

ance of

313 : Room

walls[3]

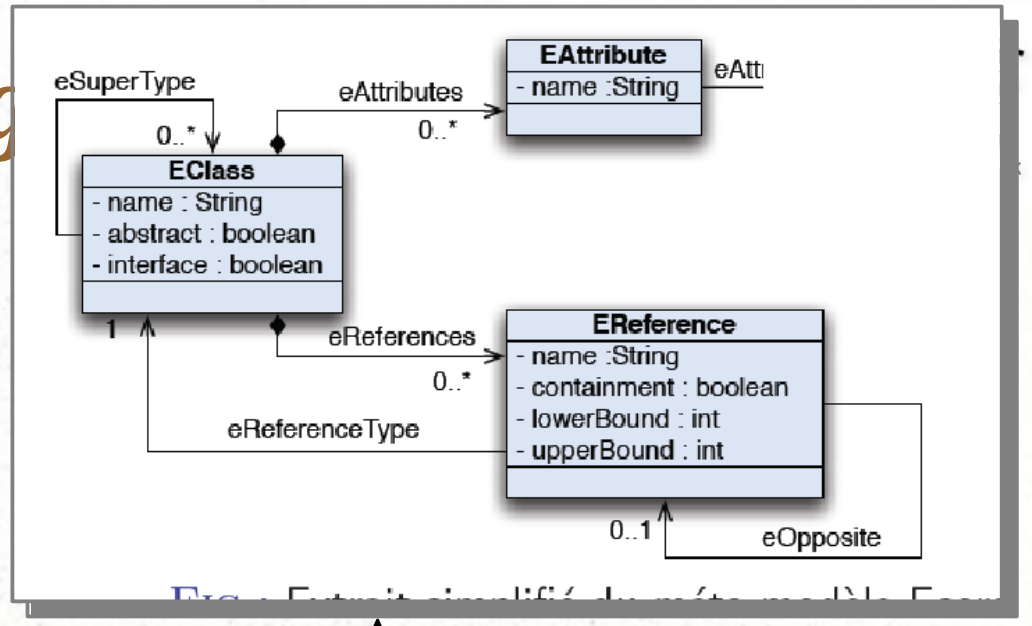
: Wall

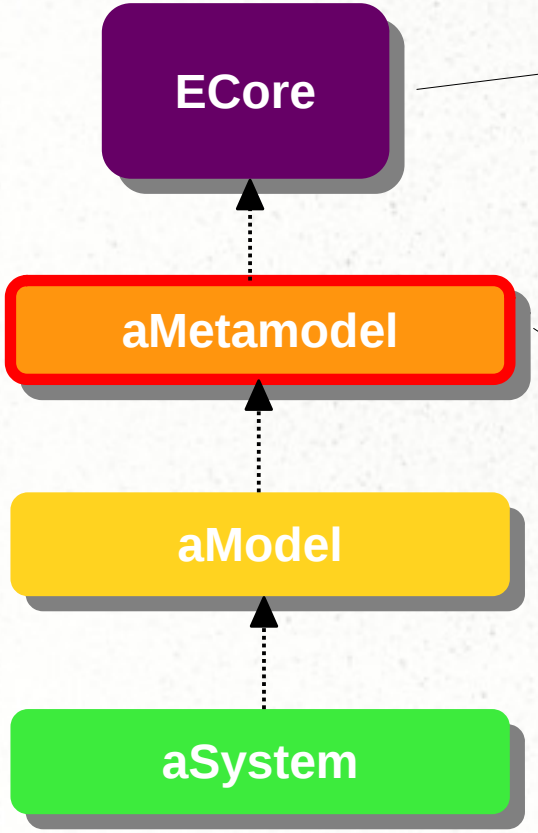
: Wall

: Wall

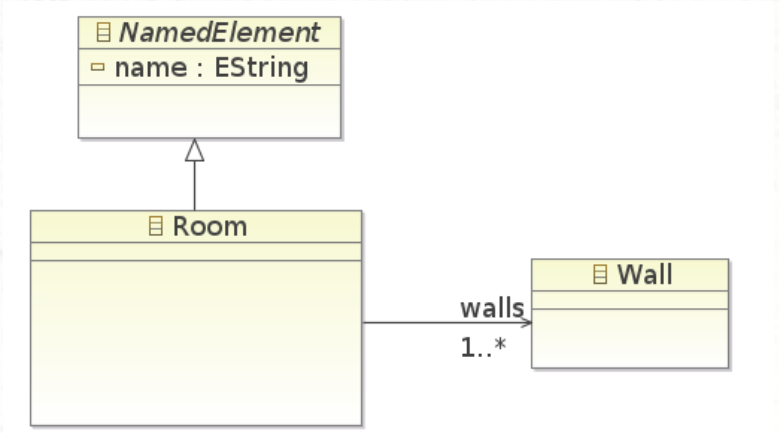
Instance of ??

Eclipse Modeling

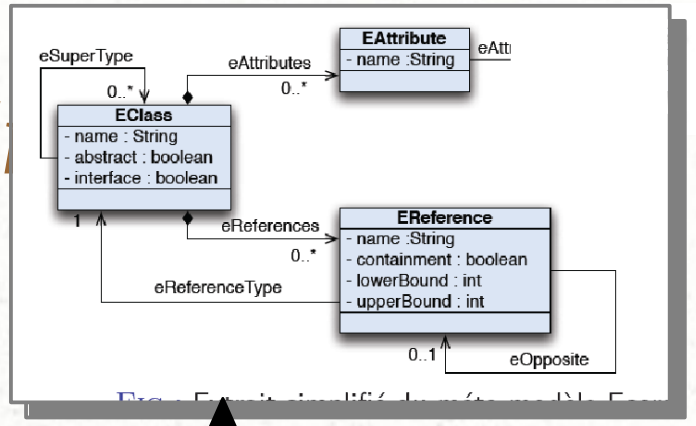




conformsTo



Eclipse Modeling F



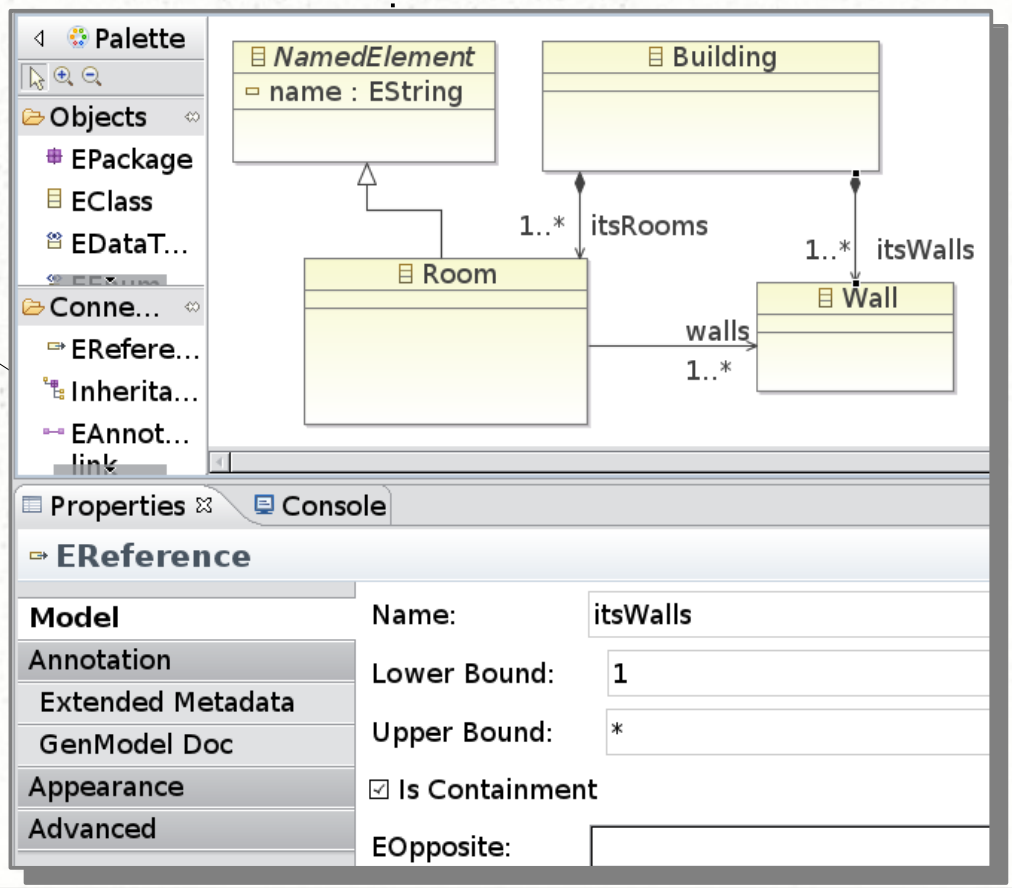
conformsTo

ECore

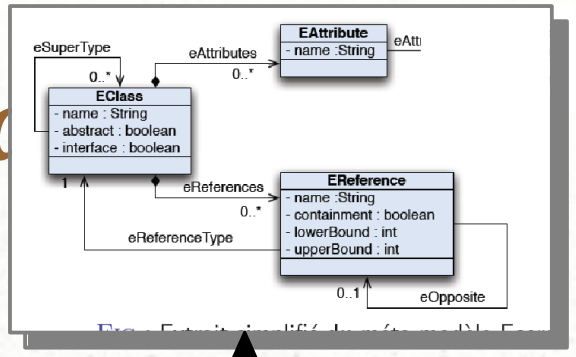
aMetamodel

aModel

aSystem



Eclipse Modeling Fr



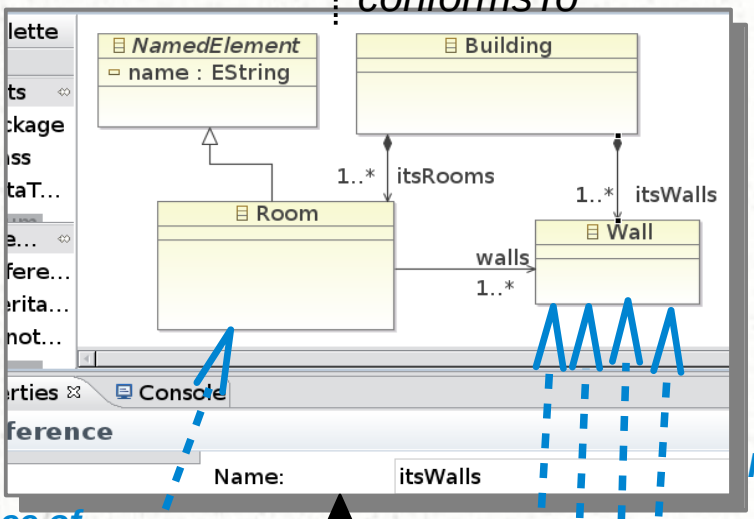
conformsTo

ECore

aMetamodel

aModel

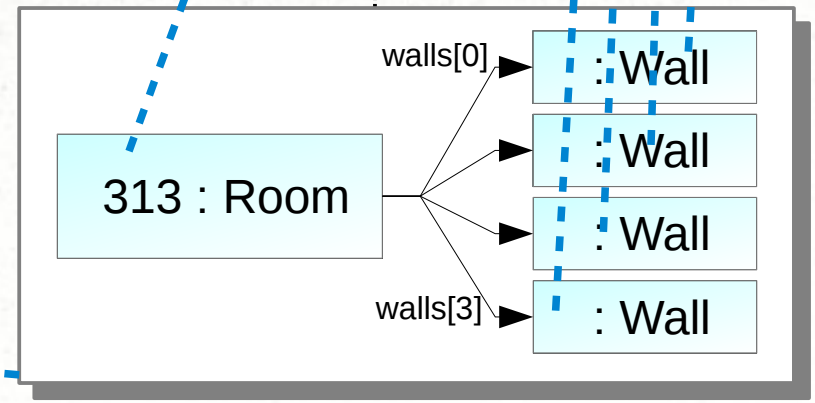
aSystem



Instance of

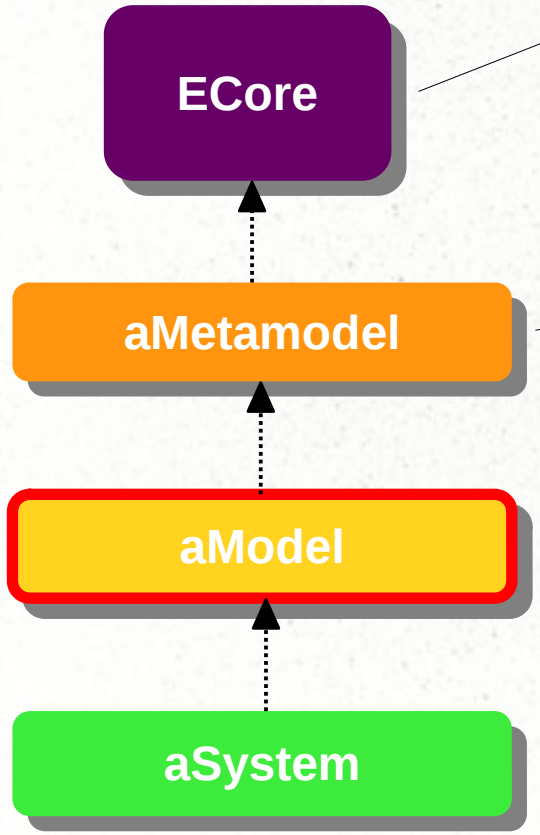
Instance of

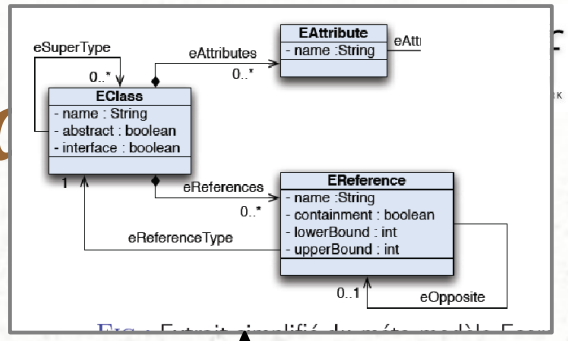
conformsTo



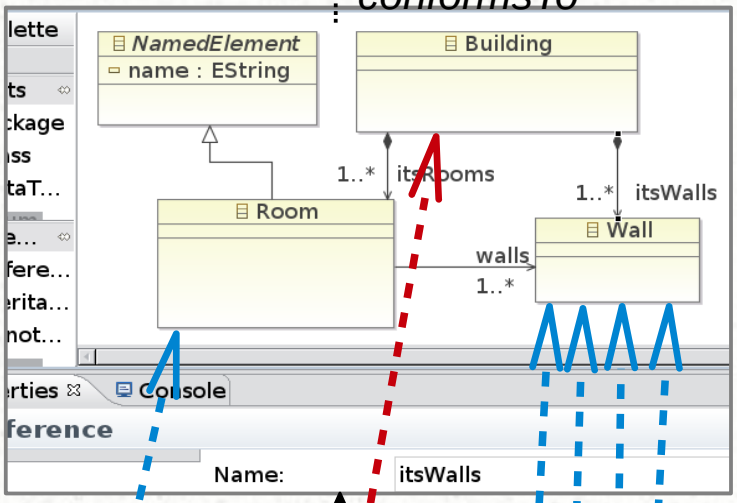
Instance of ??

Eclipse Modeling Fr





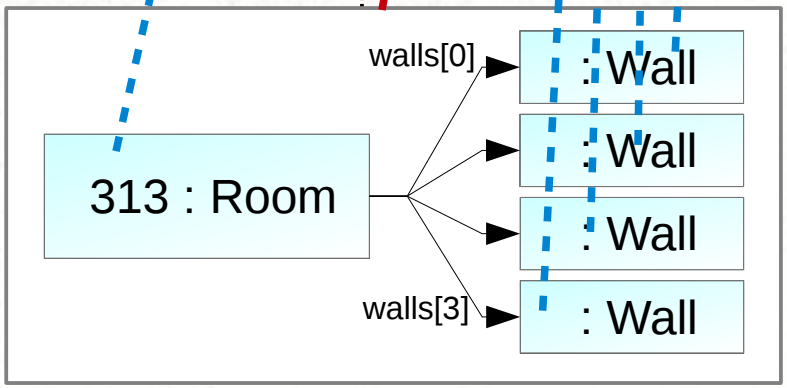
conformsTo



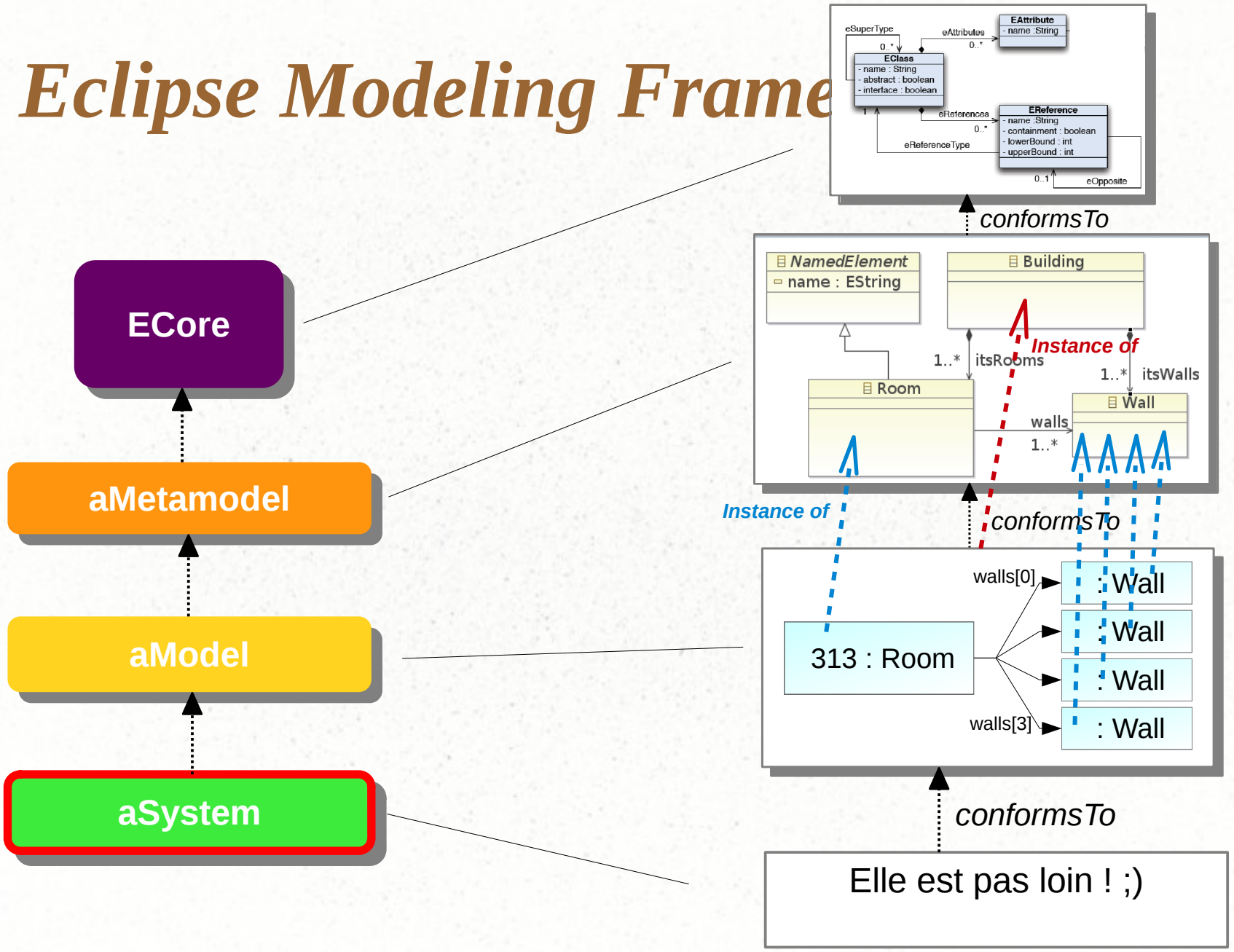
Instance of

Instance of

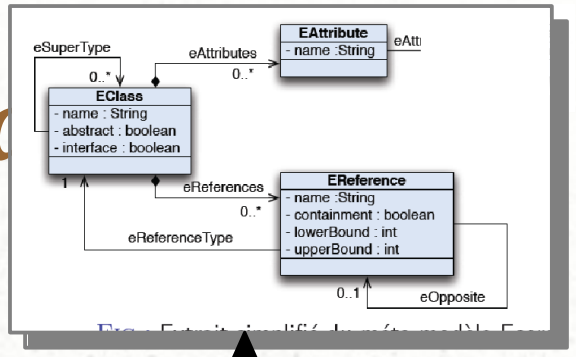
conformsTo



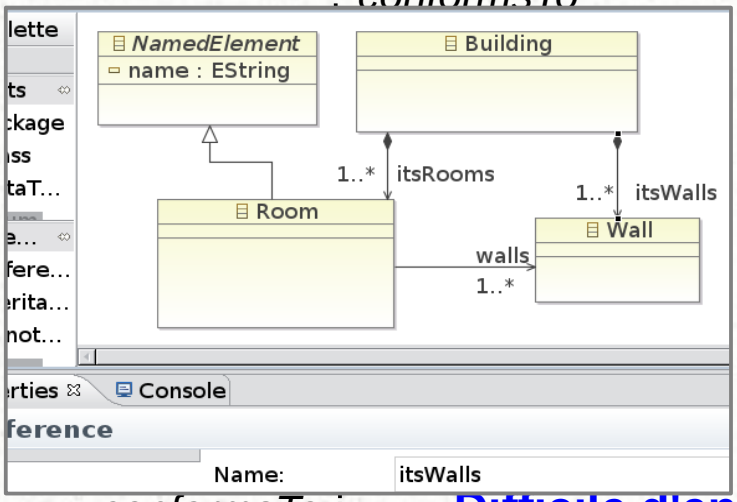
Eclipse Modeling Frame



Eclipse Modeling Fr



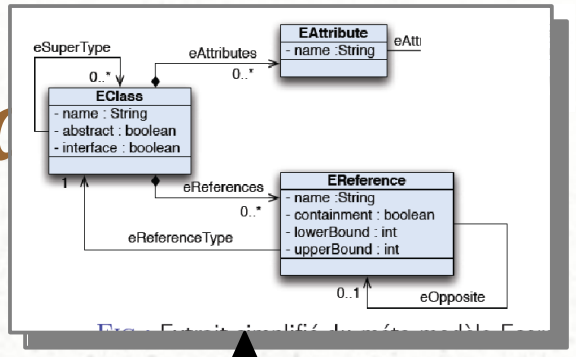
conformsTo



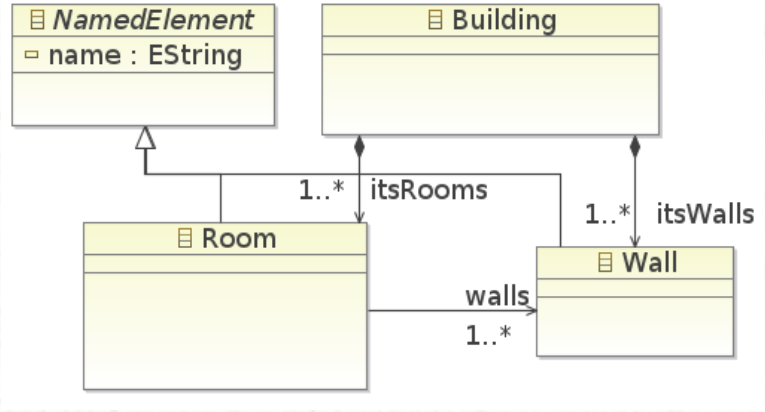
conformsTo

Difficile d'en parler....

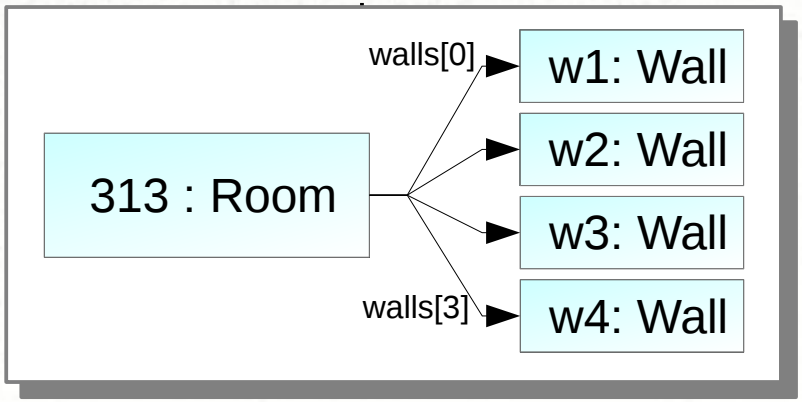
Eclipse Modeling Fr



conformsTo



conformsTo



ECore

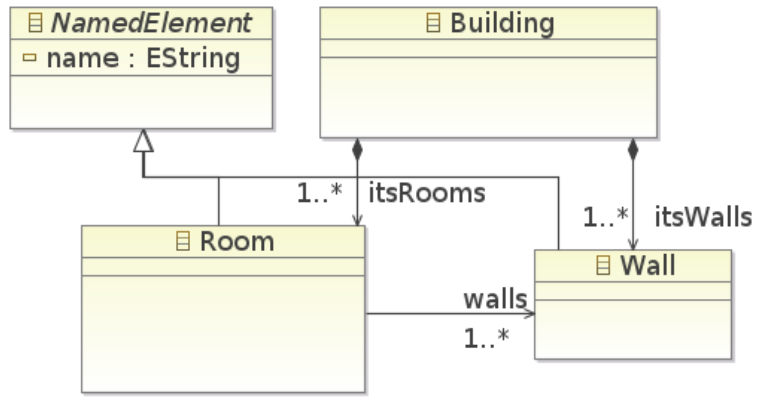
aMetamodel

aModel

aSystem

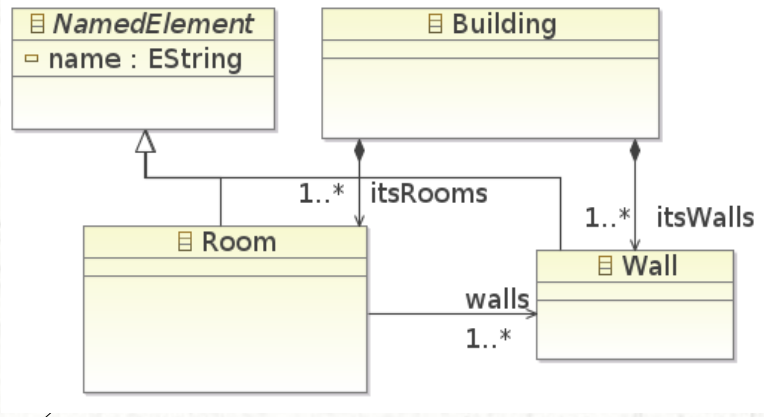
Avantages de la méta-modélisation

- Pourquoi : Pour faciliter la manipulation de modèles (éditeurs, transformations, ...)



Avantages de la méta-modélisation

- Pourquoi : Pour faciliter la manipulation de modèles (éditeurs, transformations, ...)

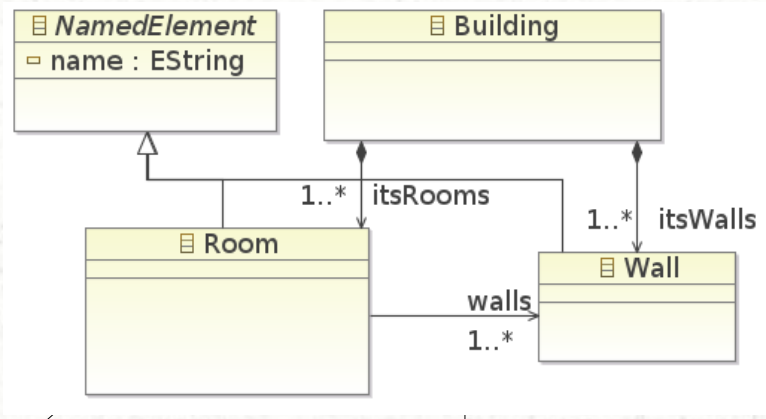


Génération d'une API de manipulation java
(sérialisation / déserialisation gratuites)

Manipulation aisée des modèles en java, gain de temps pour sauvegarder les modèles

Avantages de la méta-modélisation

- Pourquoi : Pour faciliter la manipulation de modèles (éditeurs, transformations, ...)

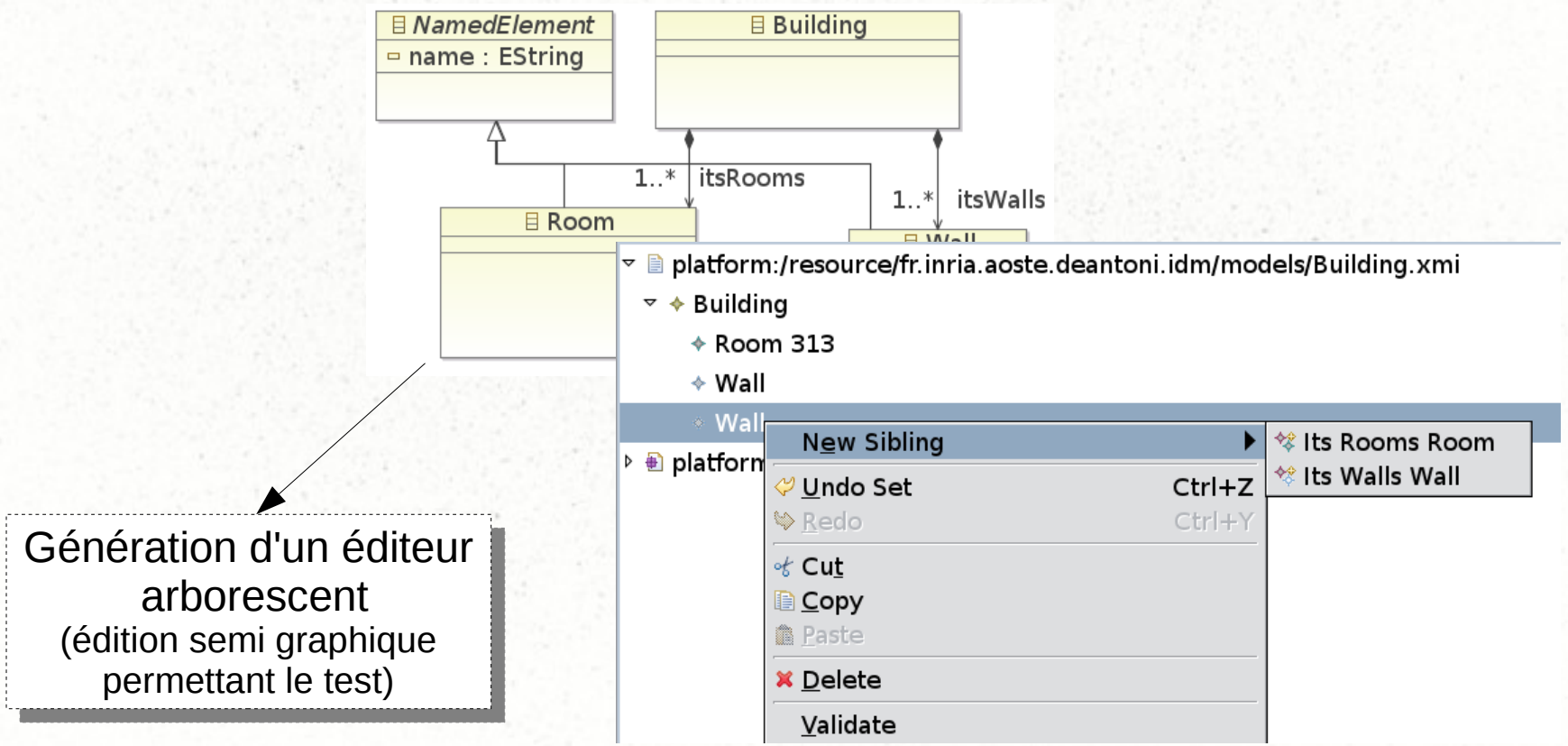


Génération d'une API de manipulation java
(sérialisation / déserialisation gratuites)

Génération d'un éditeur arborescent
(édition semi graphique permettant le test)

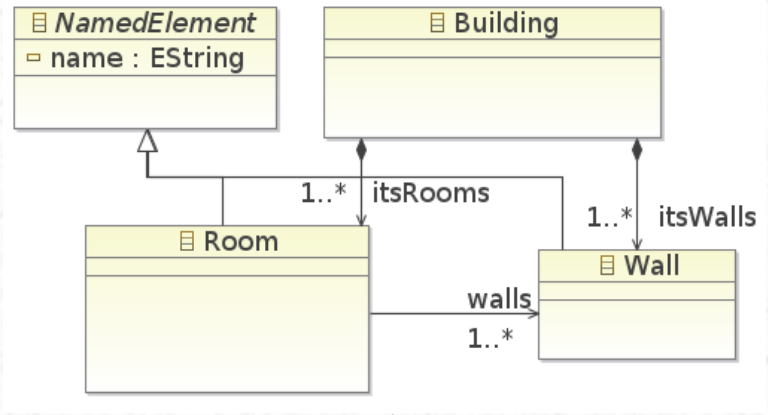
Avantages de la méta-modélisation

- Pourquoi : Pour faciliter la manipulation de modèles (éditeurs, transformations, ...)



Avantages de la méta-modélisation

- Pourquoi : Pour faciliter la manipulation de modèles (éditeurs, transformations, ...)



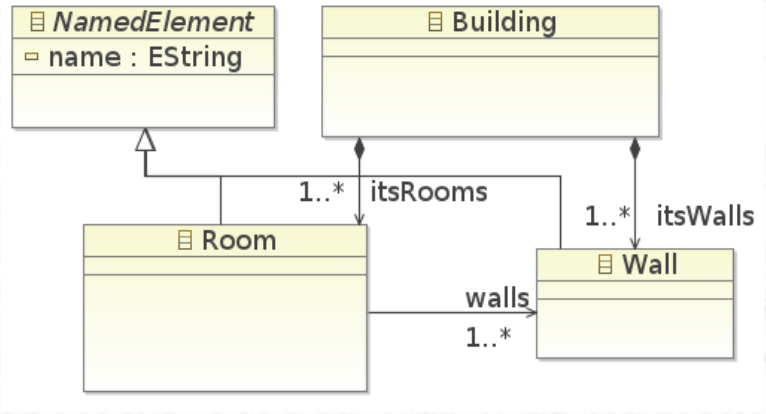
Génération d'une API de manipulation java
(sérialisation / déserialisation gratuites)

Génération d'un éditeur arborescent
(édition semi graphique permettant le test)

Accès aux outils basés sur les métamodèles:
- Xtext
- Sirius
- Transformation modèle à modèle
- Transformations modèle à texte
- ...

Avantages de la méta-modélisation

- Pourquoi : Pour faciliter la manipulation de modèles (éditeurs, transformations, ...)



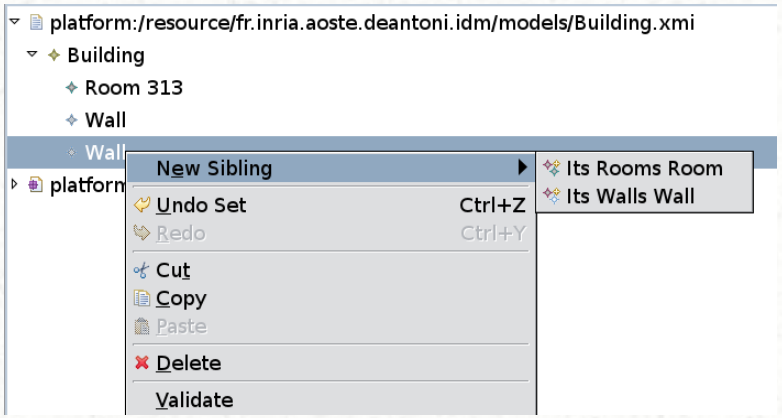
Outil de création d'une syntaxe concrète...

Accès aux outils basés sur les métamodèles:

- Xtext
- Sirius
- Transformation modèle à modèle
- Transformations modèle à texte
- ...

Syntaxes concrètes et abstraites
(caricature)

- Syntaxe abstraite

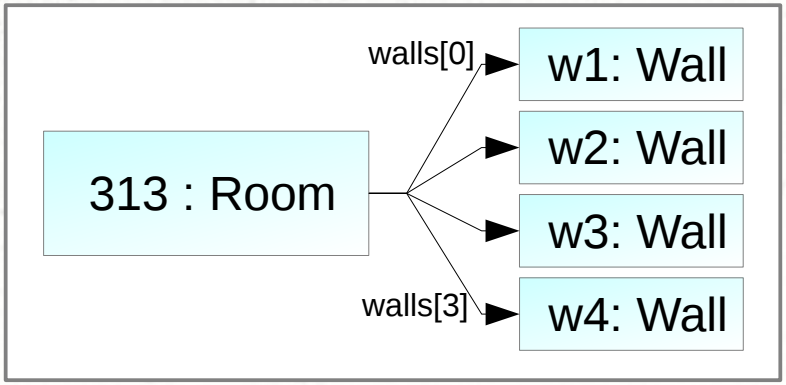


- Syntaxe concrète

textuelle

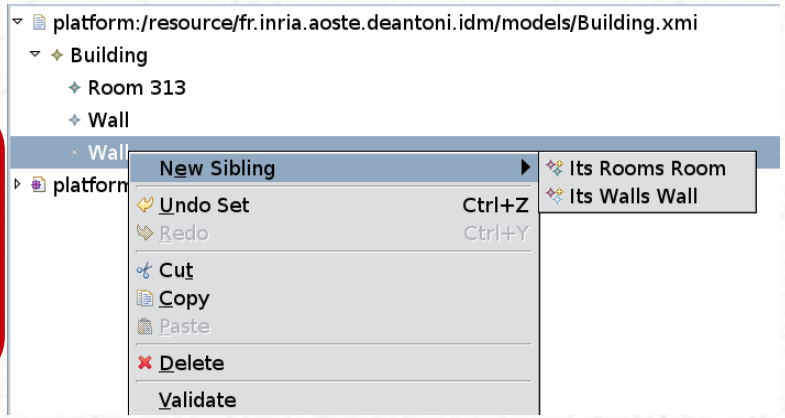
```
Building{
    Wall w1;
    Wall w2;
    Wall w3;
    Wall w4;
    Room 313 (w1,w2,w3,w4);
}
```

graphique



Syntaxes concrètes et abstraites
(caricature)

- Syntaxe abstraite



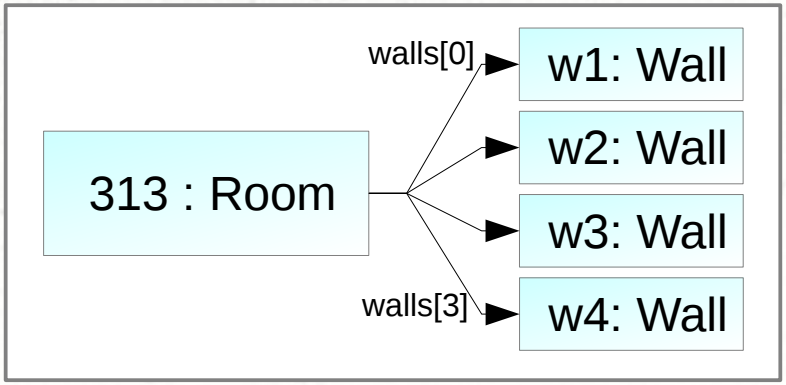
On peut avoir plusieurs syntaxes concrètes pour une même syntaxe abstraite

- Syntaxe concrète

textuelle

```
Building{
    Wall w1;
    Wall w2;
    Wall w3;
    Wall w4;
    Room 313 (w1,w2,w3,w4);
}
```

graphique



Bon ok mais...

Comment on fait...

Mise en oeuvre de EMF

1. Télécharger le dernier gemoc studio

https://ci.eclipse.org/gemoc/job/gemoc-studio-integration/job/master/lastSuccessfulBuild/artifact/gemoc-studio/gemoc_studio/releng/org.eclipse.gemoc.gemoc_studio.updatesite/target/products/

2. Créer un “ecore modeling project”
3. Donner un nom du genre
   `fr.unice.polytech.idm.lenomduprojet.model`
4. Remplissez correctement les différents champs du wizard pour éviter des problèmes futurs.

Faire le métamodèle EMF

1. Commencez votre métamodèle...
2. Pour le tester:
    1. ouvrir le fichier .ecore associé à la vue graphique (oui, c'est la vue de la syntaxe abstraite)
    2. Clic droit sur la classe root puis “create dynamic instance”
    3. Essayer de créer des modèles en utilisant la syntaxe abstraite
3. Continuer votre métamodèle et rebouclez en 2 ou finir...

finaliser le métamodèle EMF

1. Dans la vue syntaxe abstraite, regardez les propriétés de chaque package
2. Renseigner NSPrefix tel que le package root ait un nom de la forme
   `fr.unice.polytech.idm.lenomduprojet`
3. Et une NSUri de la forme
   `http://fr.unice.polytech.idm.lenomduprojet`

Générer l'API java et le code de l'éditeur

1. Clic droit sur le .ecore
2. New → EMF generator model
3. Ouvrir et modifier si souhaité les paramètres du fichier généré (.genmodel). C'est un modèle de la génération...
4. Click droit sur la racine du modèle et vous pouvez générer ce que vous voulez (soyez fous, générez tout)

Utiliser l'éditeur

1. Vous avez généré le code de l'éditeur (c'est un plug-in eclipse)
2. Vous devez lancer un nouvel eclipse dans lequel ce code sera “déployé”

    click droit et run as “Eclipse application” (vous pouvez pour plus de contrôle aller dans le run configuration !)

Bon...

À vous maintenant !