Ingénierie Dirigée par les modèles

Organisation du module

- Contenu :
  - Modèles, métamodèles, méta-métamodèles
  - Transformations de modèles
  - DSML et syntaxe graphique
  - Syntaxe textuelle : xtext (?)
- Intervenantes
  - Clémentine Nebut
  - Marianne Huchard

Organisation du module

- Organisation hebdomadaire nominale :
  - 13h15-16h30 : CM / TD
  - 16h45-18h15 : TP
- Mais surtout des exceptions !
  - Regardez l’emploi du temps
  - Écoutez les consignes en cours !
- MCC :
  - CCI
  - Plus d’informations à venir ...

D'où vient l'ingénierie dirigée par les modèles ?

Un peu d’histoire

- Naissance du génie logiciel : fin des années 60
  - Crise du logiciel (années 60), artisanat du logiciel
- Décomposition fonctionnelle, modularité
- Naissance des objets face à la variation des exigences
- Composants, lignes de produits
- Omni-présence logicielle
- Obsolescence technologique accélérée

Comment maîtriser la complexité du logiciel ?

Unification des concepts : l'objet ?

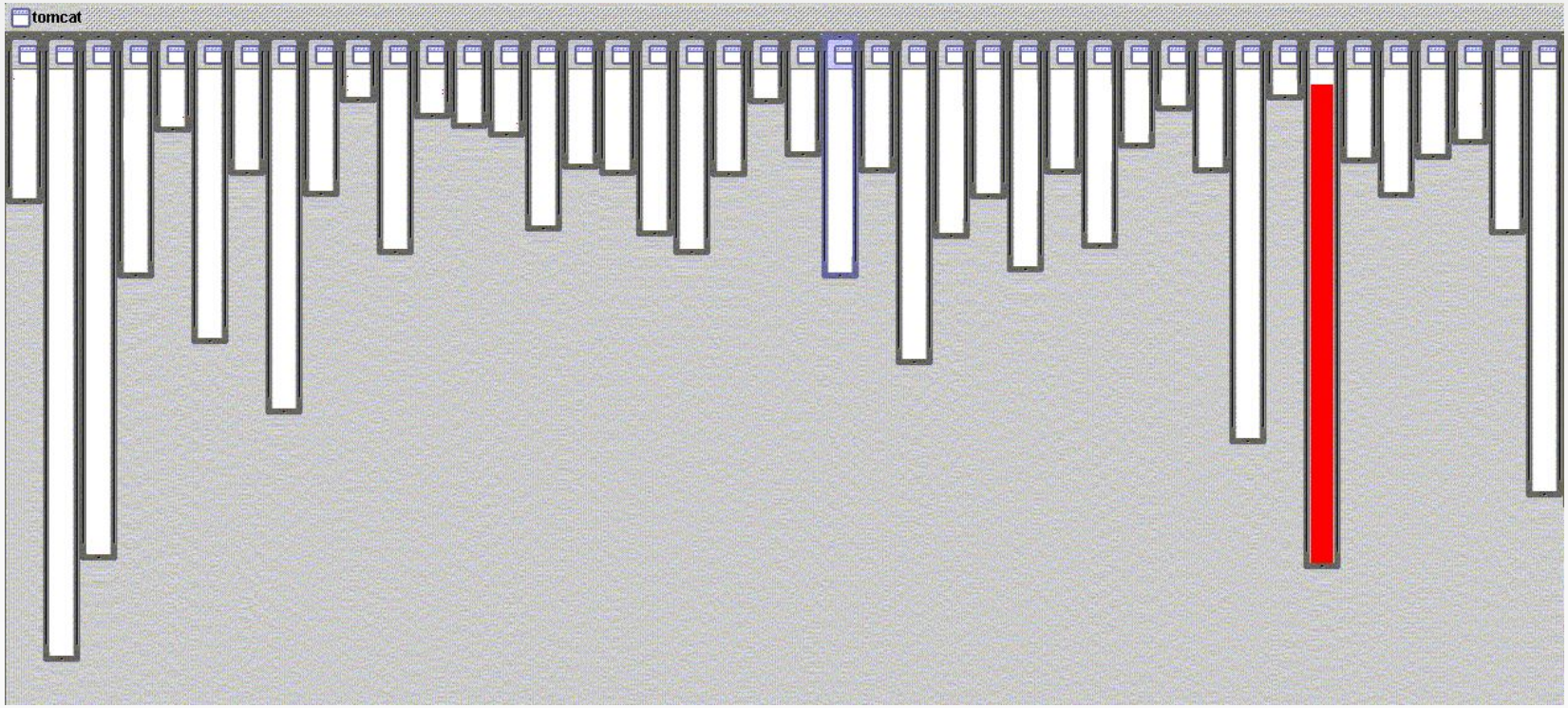
- Un des objectifs de l’objet : réduire le nombre de concepts
- Mais …

Les patrons de conception

- La complexité est entre les objets, dans les collaborations
- Naissance des patrons de conception
  - Encapsulent des savoir-faire d'experts
  - Traitent autant du problème que des solutions
  - Traitent d'aspects non-fonctionnels
  - Des collaborations paramétrées

- Les patrons de conception ne sont pas des objets

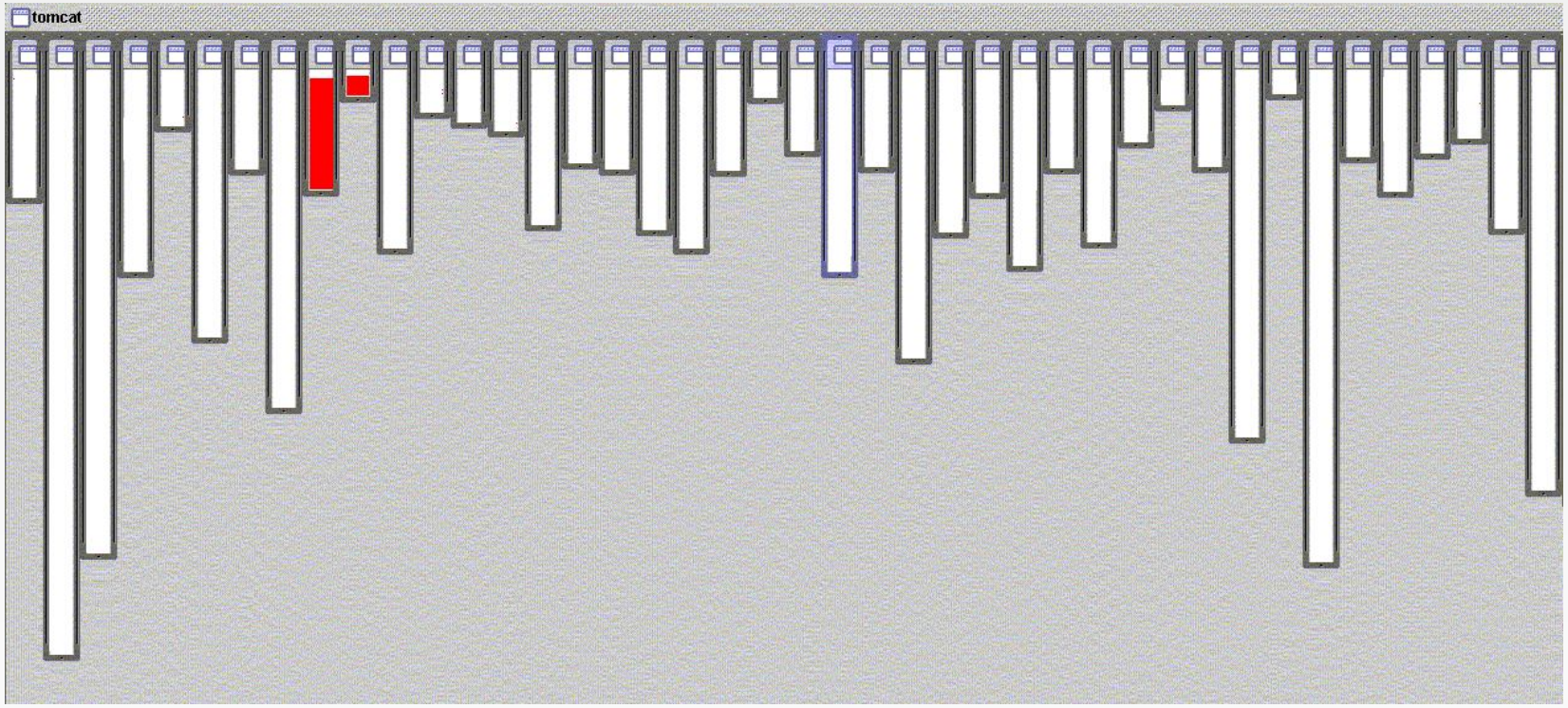
Les aspects



- XML parsing in org.apache.tomcat
  - red shows relevant lines of code
  - nicely fits in one box

Transparent emprunté à Mik Kersten, AOSD-OOPSLA'02

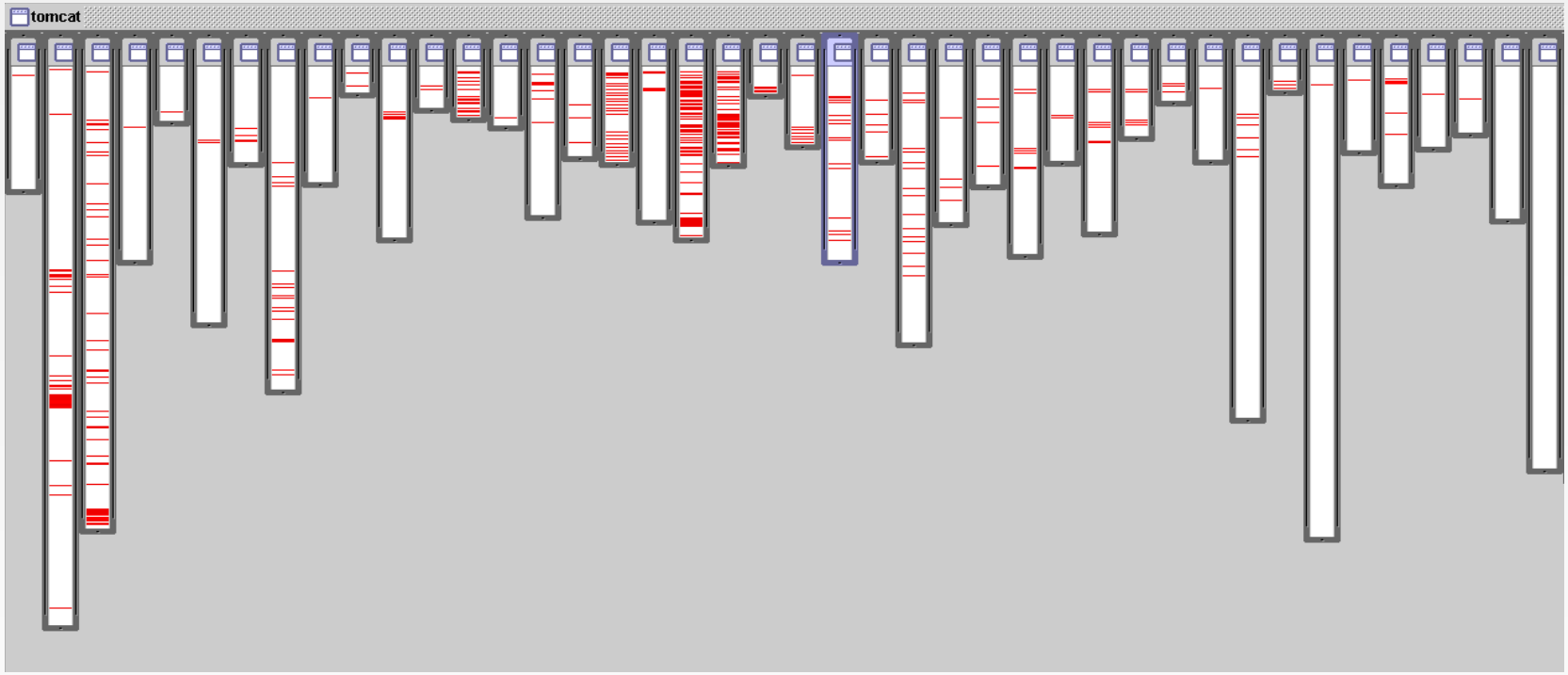
Les aspects



- URL pattern matching in org.apache.tomcat
  - red shows relevant lines of code
  - nicely fits in two boxes (using inheritance)

Transparent emprunté à Mik Kersten, AOSD-OOPSLA'02

Les aspects



- logging in org.apache.tomcat
  - red shows lines of code that handle logging
  - not in just one place
  - not even in a small number of places

Transparent emprunté à Mik Kersten, AOSD-OOPSLA'02

Les aspects

- Aspect-oriented Programming
  - Kiczales et al, ECOOP97
- Encapsuler les préoccupations orthogonales

Des objets aux composants, les services webs

- Composants
  - Notion de déploiement
  - Assemblage similaire aux composants électroniques
  - Configuration
  - Persistance
  - ...
- Engouement pour les services web comme solution à l’interopérabiblité

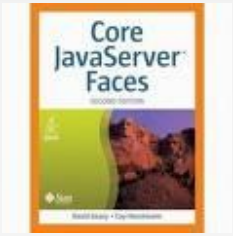
Middleware ou Middle war ?

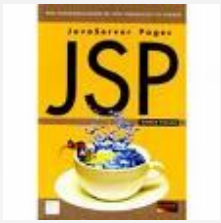
« It is difficult – in fact next to impossible – for a large enterprise to standardize on a single middleware platform »

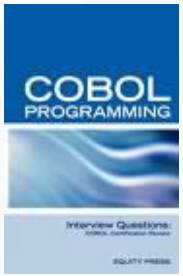
R. Soley

- COM/DCOM
- Sun : java et EJB
- Microsoft .NET
- CORBA, IIOP
- Services web, XML, SOAP, REST
- …

L’évolution des technologies : le nombre







L'évolution des technologies : la perennité

Retirement of J# language and Java Language Conversion Assistant from future versions of Visual Studio

Since customers have told us that the existing J# feature set largely meets their needs and usage of J# is declining, Microsoft is retiring the Visual J# product and Java Language Conversion Assistant tool to better allocate resources for other customer requirements. The J# language and JLCA tool will not be available in future versions of Visual Studio. To preserve existing customer investments in J#, Microsoft will continue to support the J# and JLCA technology that shipped with Visual Studio 2005 through to 2015 as per our product life-cycle strategy. For more information, see Expanded Microsoft Support Lifecycle Policy for Business & Development Products.

http://msdn.microsoft.com/en-us/vjsharp/default.aspx

L’évolution des technologies

- Nombre de technologies
- Perennité ?
- Interopérabilité
- Maintenance et migration
- Mélange métier/technique

Le mélange métier/technique

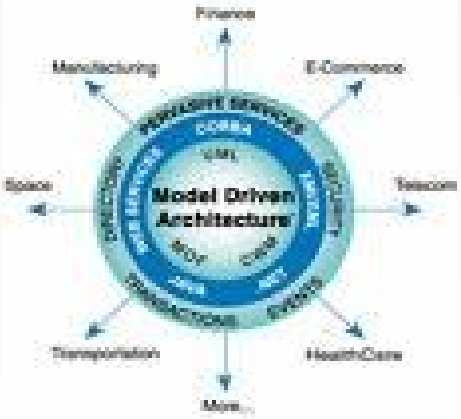
- La logique métier est mélangée avec le code technique
  - Vrai pour toutes les technologies
  - Même si les annotations (@webmethod et autres [webmethod]) allègent un peu le code
- L'évolution des applications n'est pas simple
  - Garder la logique métier
  - Jeter le code technique
- *Nécessite de s'abstraire des technologies*
  - *Modéliser de manière abstraite le métier*
  - *Projeter ensuite sur des technologies (kleenex)*

Les solutions de modélisation

- UML – 1995

- Le profil UML
  - Si ça ne rentre pas par la porte, ça rentrera par la fenêtre

- MDA



- MDE

Métamodèles

Modèle : une définition

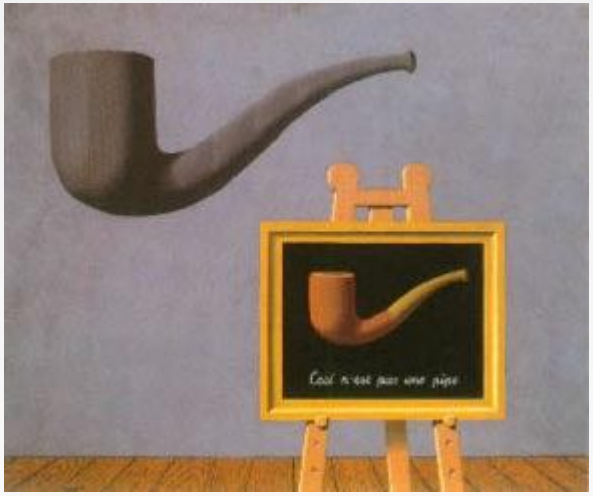
- Modeling, in the broadest sense, is the cost-effective use of something in place of something else for some cognitive purpose. It allows us to use something that is simpler, safer or cheaper than reality instead of reality for some purpose.

- A model represents reality for the given purpose; the model is an abstraction of reality in the sense that it cannot represent all aspects of reality. This allows us to deal with the world in a simplified manner, avoiding the complexity, danger and irreversibility of reality.

"The Nature of Modeling"
Jeff Rothenberg, 1989

Le modèle et la réalité

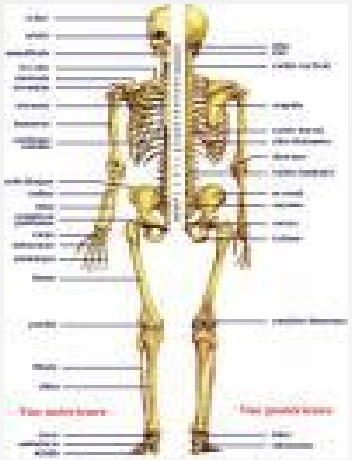
René Magritte,
« La trahison des images »
1928-29



René Magritte,
« Les deux mystères »
1966

Le modèle et la réalité

- Modèle = représentation simplifiée d'une partie du monde : le système
- Pour un objectif de modélisation donné, le modèle répond de la même façon que le système modélisé



Est décrit par

Espace de modélisation

Le "vrai" monde

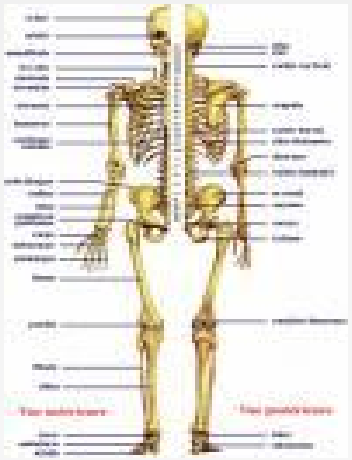
Homme

©R. Doisneau

Le modèle et la réalité

Nombre d'os ?

204



Est décrit par

Espace de modélisation

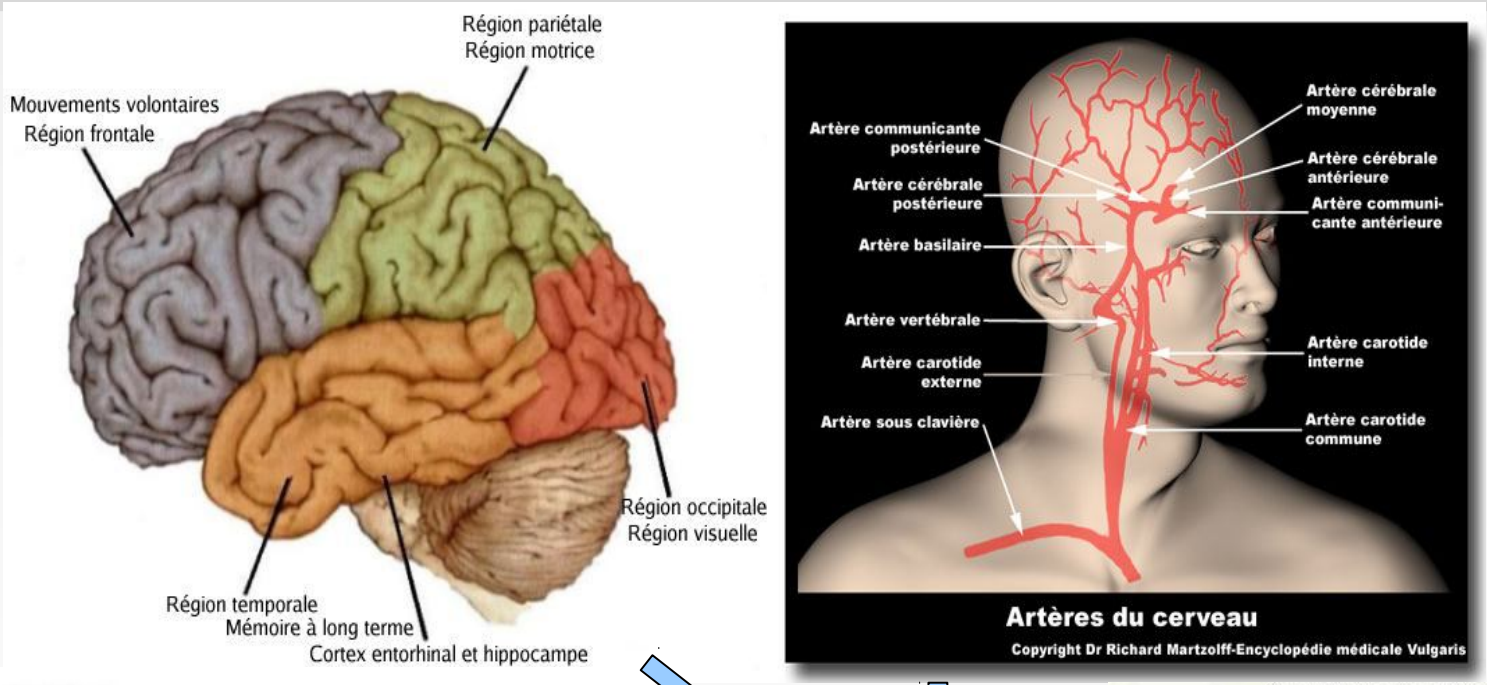
Le “vrai” monde

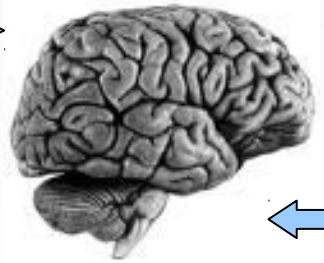
Homme

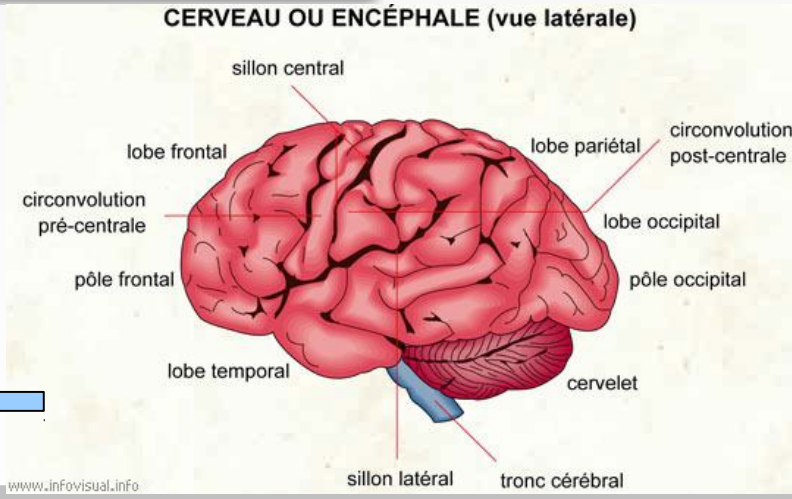
Nombre d'os ?

204

Un modèle, un point de vue







Précision d'un modèle

"That's another thing we've learned from *your* Nation," said MeinHerr, "map-making. But we've carried it much further than you. What do you consider the *largest* map that would be really useful?"

"About six inches to the mile."

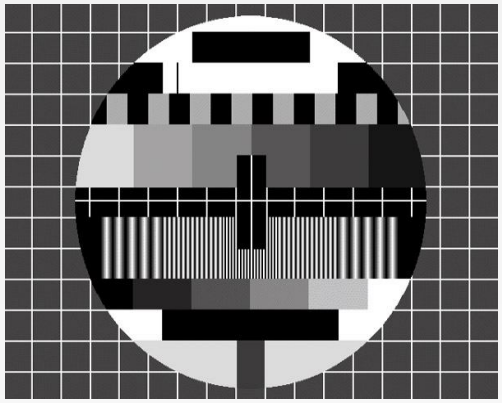
"Only *six inches!"* exclaimed MeinHerr. "We very soon got to six *yards* to the mile. Then we tried a *hundred* yards to the mile. And then came the grandest idea of all! We actually made a map of the country, on the scale of *a mile to the mile!"*

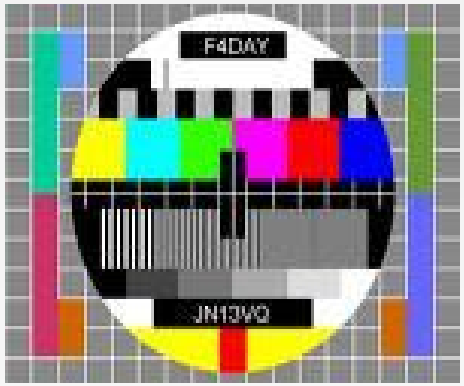
"Have you used it much?" I enquired.

"It has never been spread out, yet," said MeinHerr: "the farmers objected: they said it would cover the whole country, and shut out the sunlight! So we now use the country itself, as its own map, and I assure you it does nearly as well. »

Lewis Carroll, *Sylvie and Bruno concluded* (Londres, 1893)

Que puis-je utiliser pour concevoir mes modèles ?

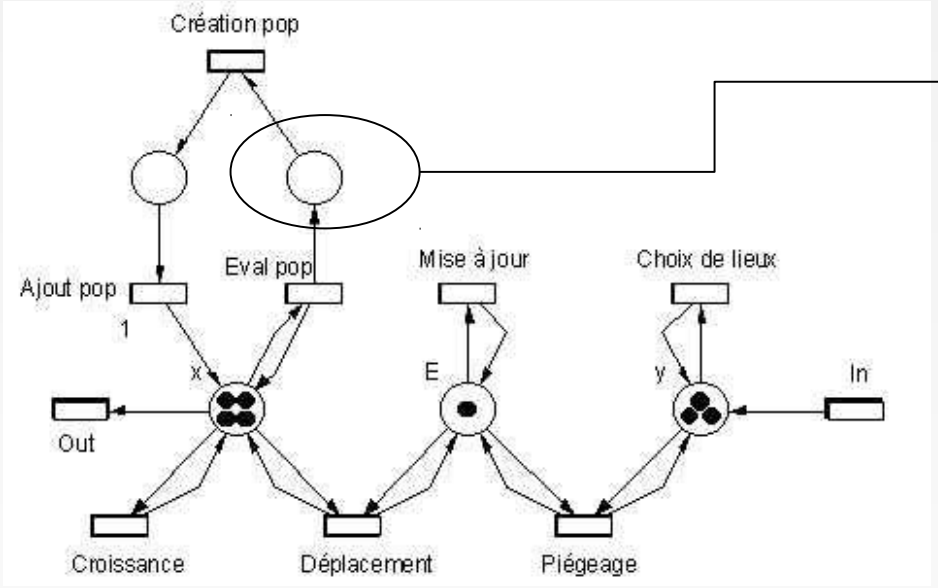




Couleur ou noir et blanc ?

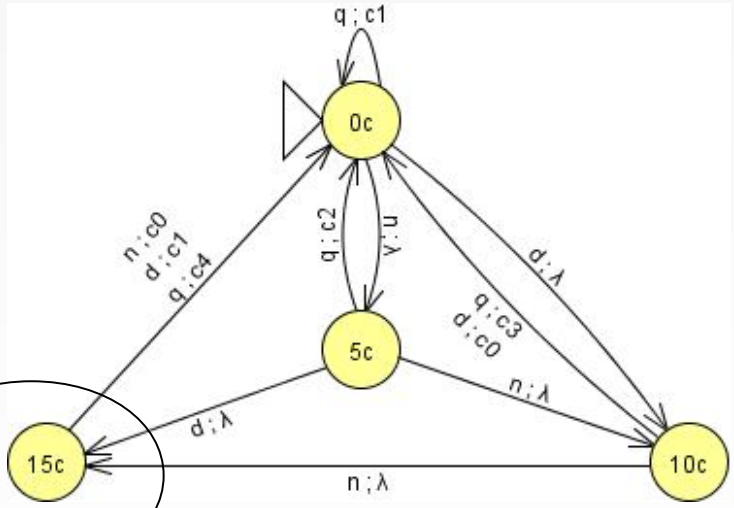
2D ou 3D ?

Que puis-je utiliser pour concevoir mes modèles ?



place

Réseau de Pétri



état

Machine de Mealy

Modèles et métamodèles

- « Un méta-modèle définit une abstraction, ses concepts »
  - Les règles de définition des descriptions
- Le méta-modèle est la légende de la carte.

Métamodèles

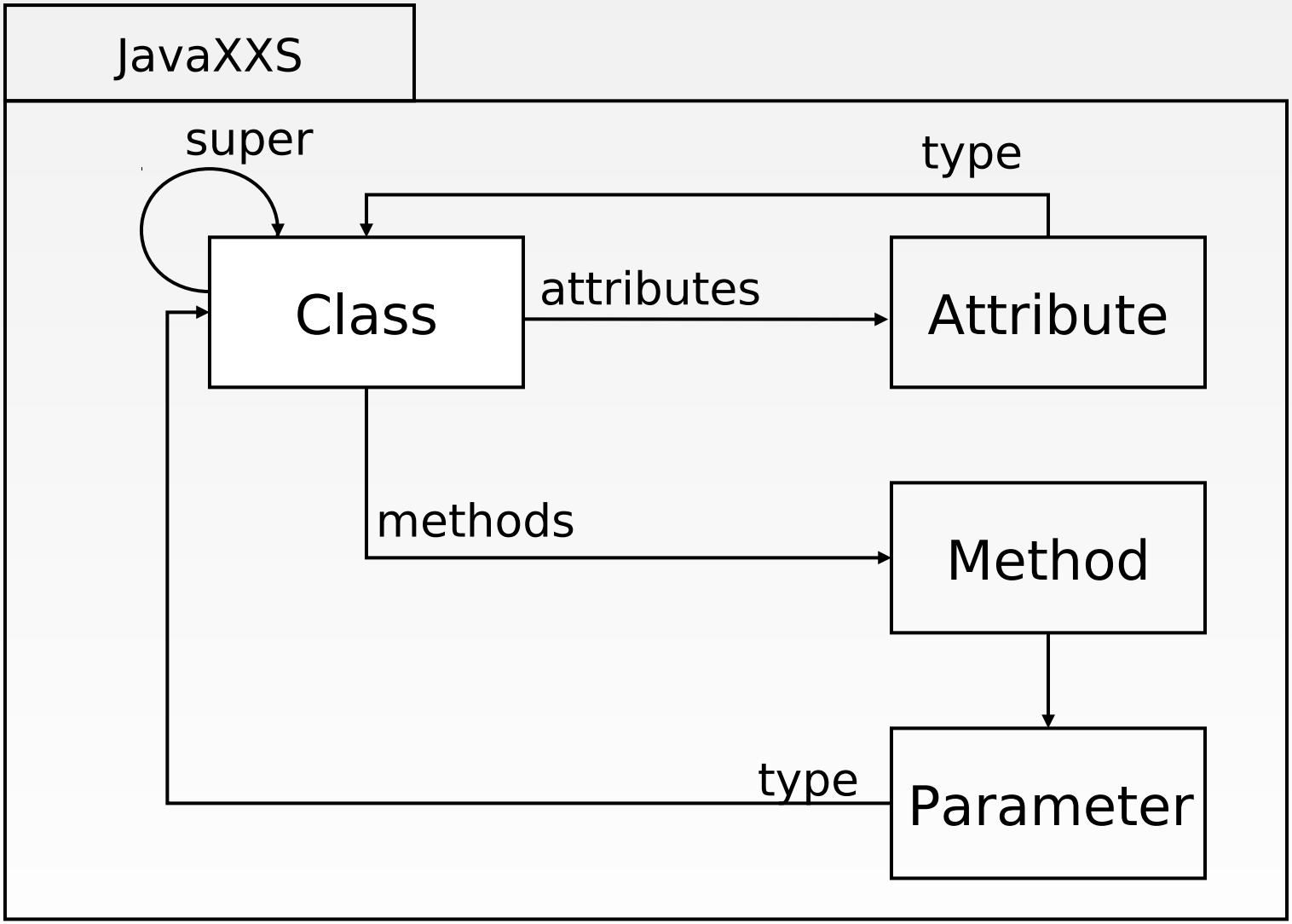
- *« A model is a description of (part of) a system written in a well-defined language »*

  A. Kleppe, S. Warmer, W. Bast, « MDA explained . The model driven architecture: Practice and Promise »

*« A **meta-model** is a model that defines the language for expressing a **model** »*

OMG, « Meta-Object Facilities (MOF) Specification » Version 1.4 Avril 2002

Exemple : un métamodèle simple de Java



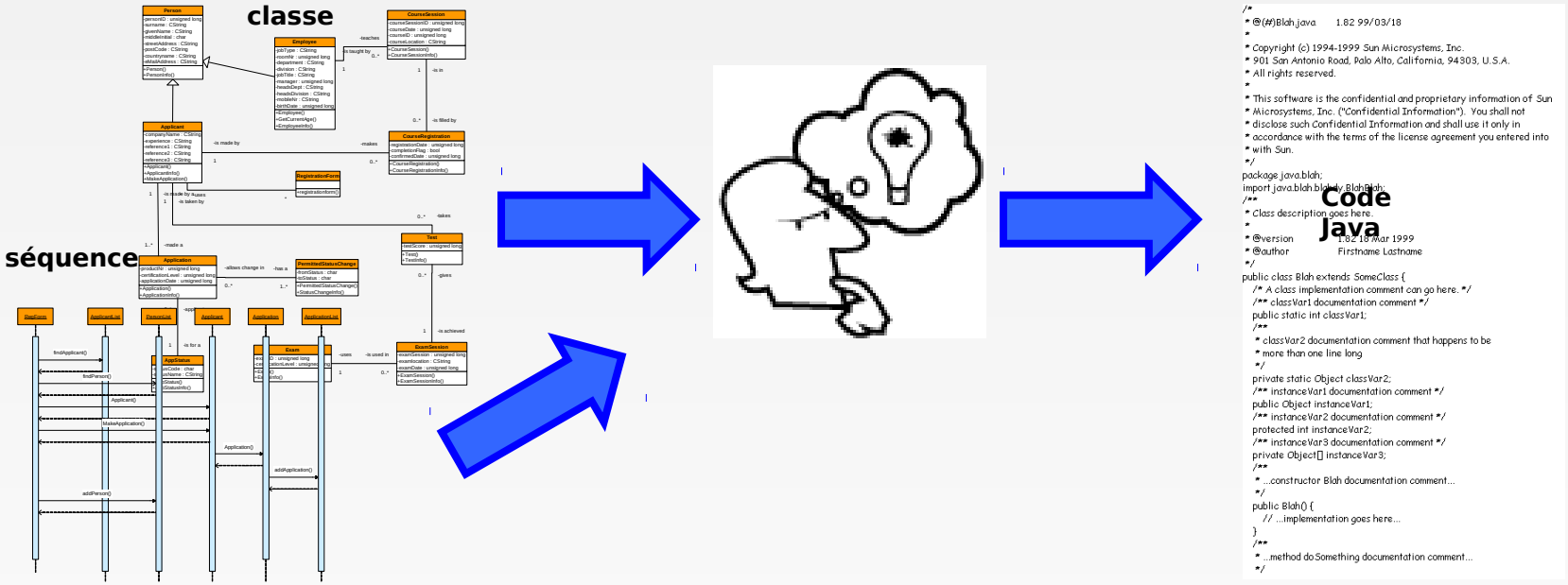
Exemple : le métamodèle d’UML

- Définit tous les éléments disponibles pour écrire un modèle UML :
  - classe
  - paquetage,
  - attribut
  - ...

Vertus des métamodèles

- Définissent le langage du modèle
  - Le modèle devient manipulable
  - Donnent du sens aux modèles
  - Le modèle n'est plus juste une image pour décorer un dossier de conception
- Homogénéité de concept
  - Le métamodèle est un modèle

Des modèles contemplatifs ...

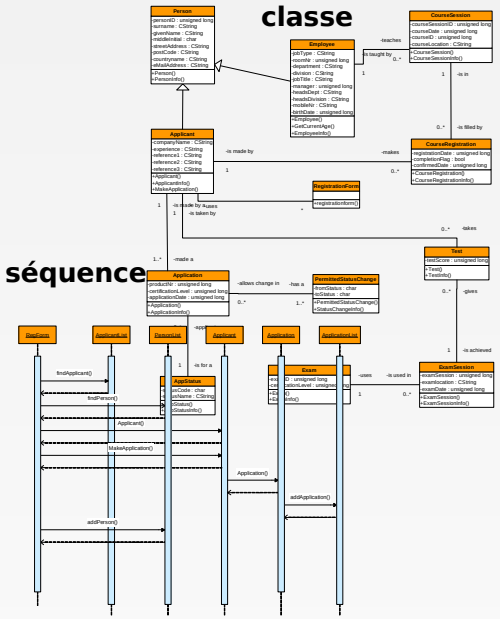


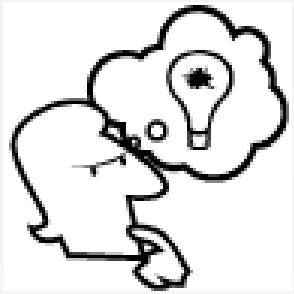
Des modèles

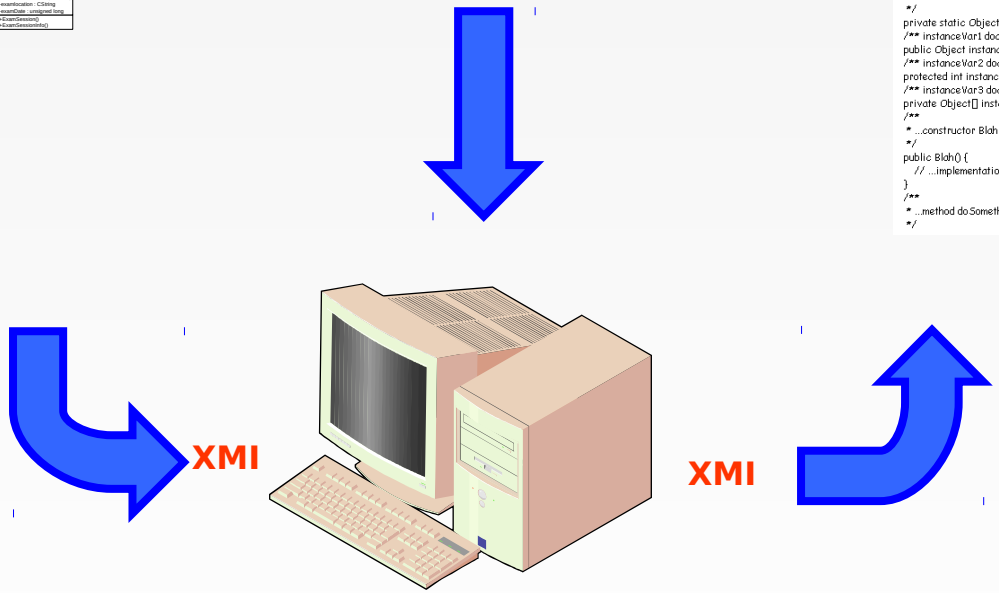
comme documentation

comme syntaxe graphique pour C#, Eiffel, Java, ...

… aux modèles productifs



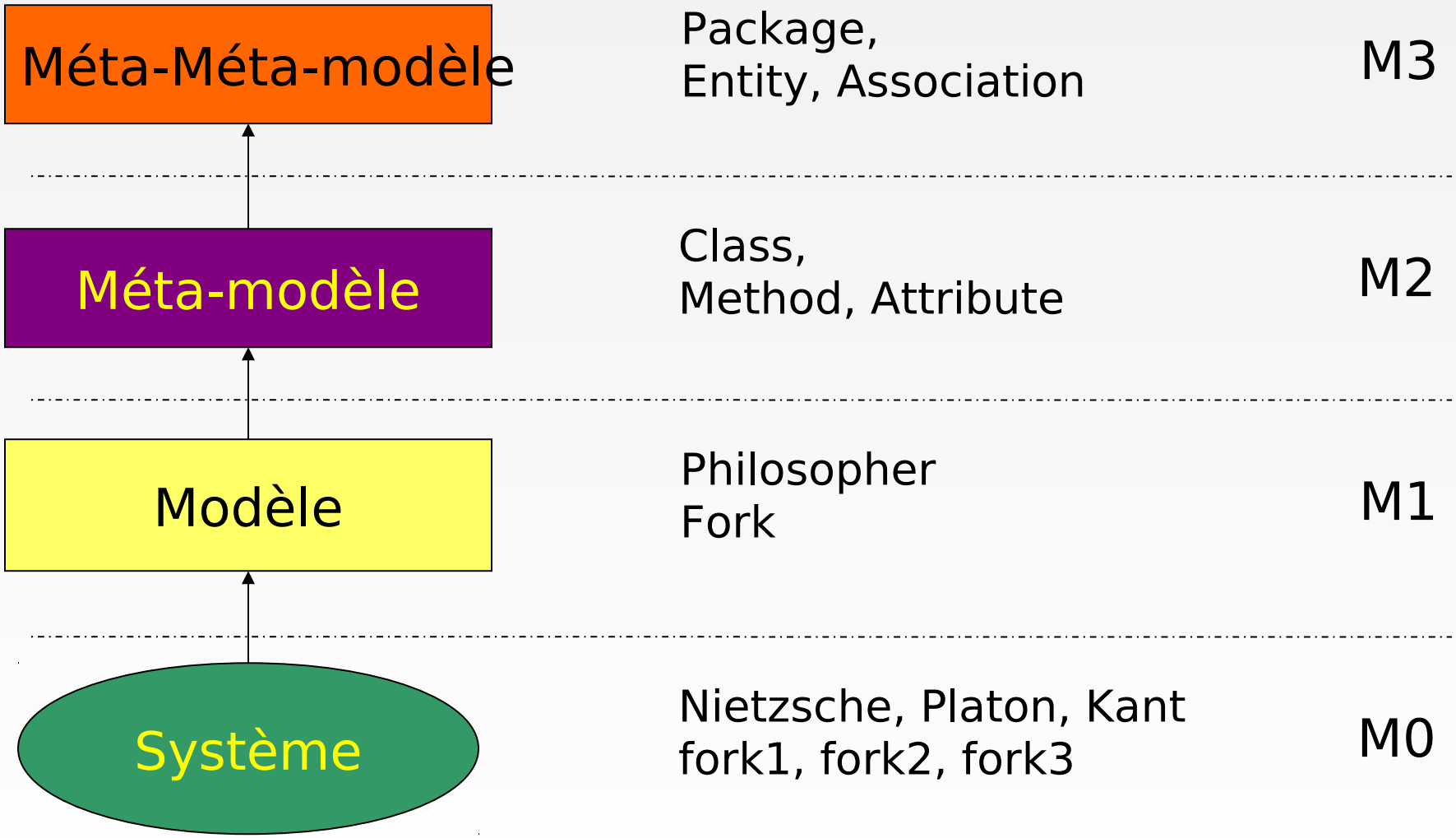




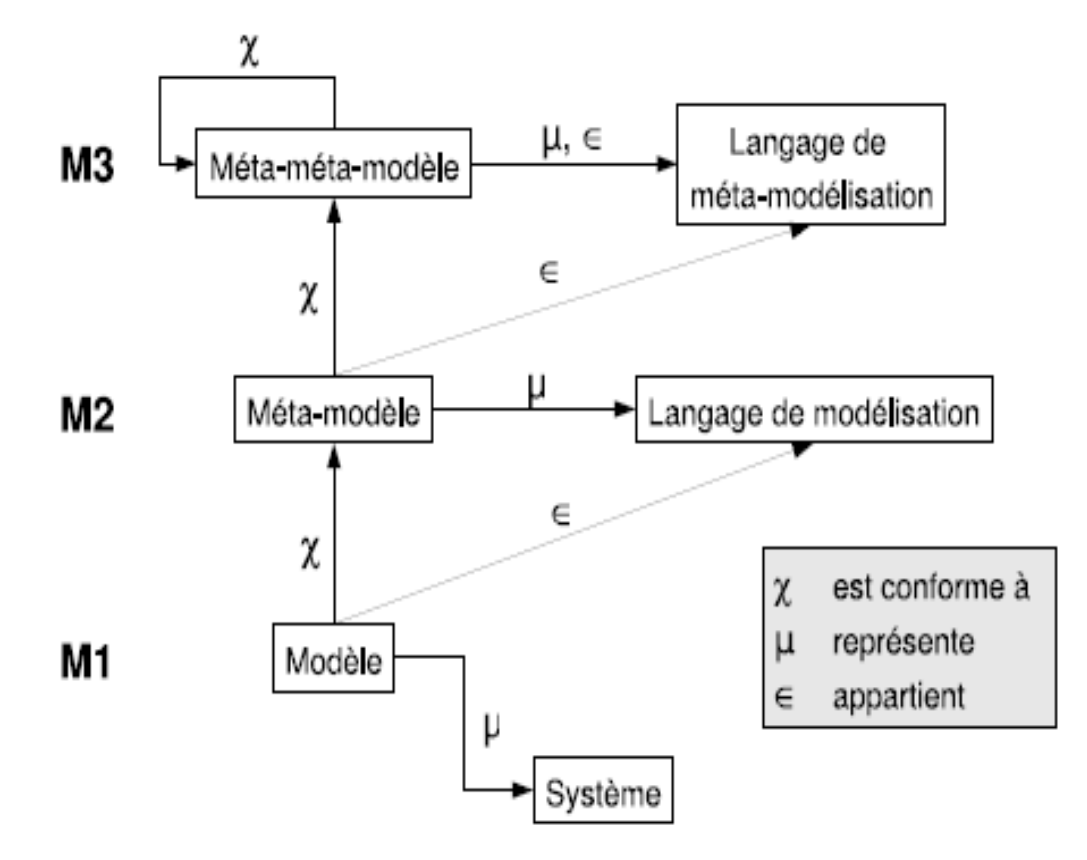
Mais au fait ...

- Comment est défini le métamodèle ?
- Par un méta-métamodèle (si si)

Des instances aux méta-modèles



Niveaux de modélisation



Extrait de la thèse de Franck Fleurey (IRISA, Rennes)

Pile de méta-modélisation (OMG)

- Un unique méta-modèle
  - Meta Object Facility (MOF)
  - Concepts de méta-modélisation
- Un ensemble de méta-modèles
  - Pour différents besoins / domaines
  - Compatibles puisque définis avec le MOF
- Un grand nombre de modèles
  - Décrivant un grand nombre de systèmes
  - Définis avec les concepts de leurs uniques méta-modèles

Le MOF

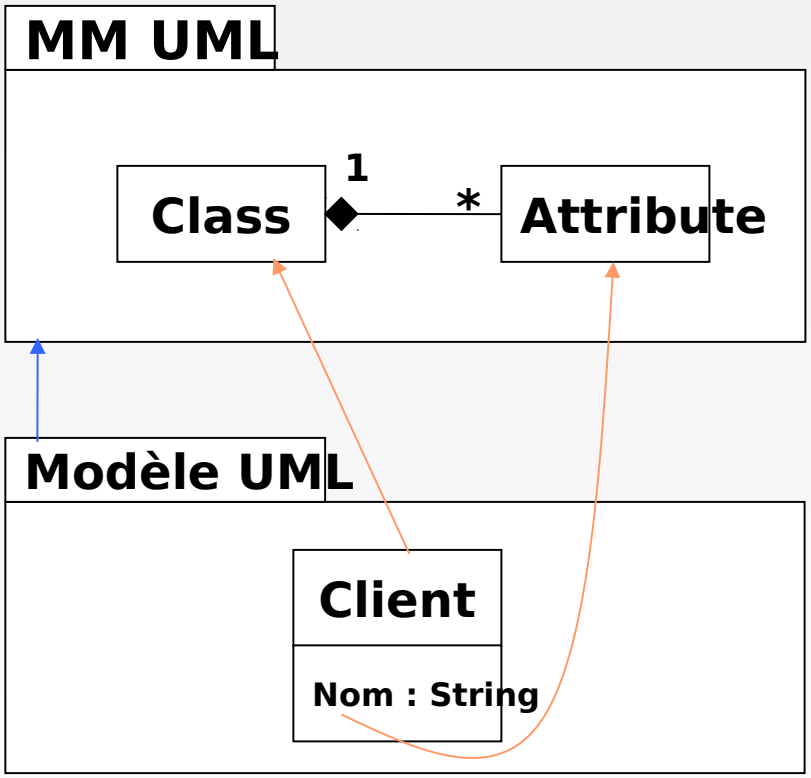
Meta Object Facility

- Méta-méta-modèle méta-circulaire
  - Suffit à sa propre définition
- Les concepts pour la définition de M2
  - Plus d'une vingtaine (MOF 1.4)
- Concepts essentiels
  - Classe (+méthodes et attributs)
  - Association
  - Références
  - Package

Modèle -> Méta-modèle

Relation
entité → méta-entité

Relation
modèle → méta-modèle

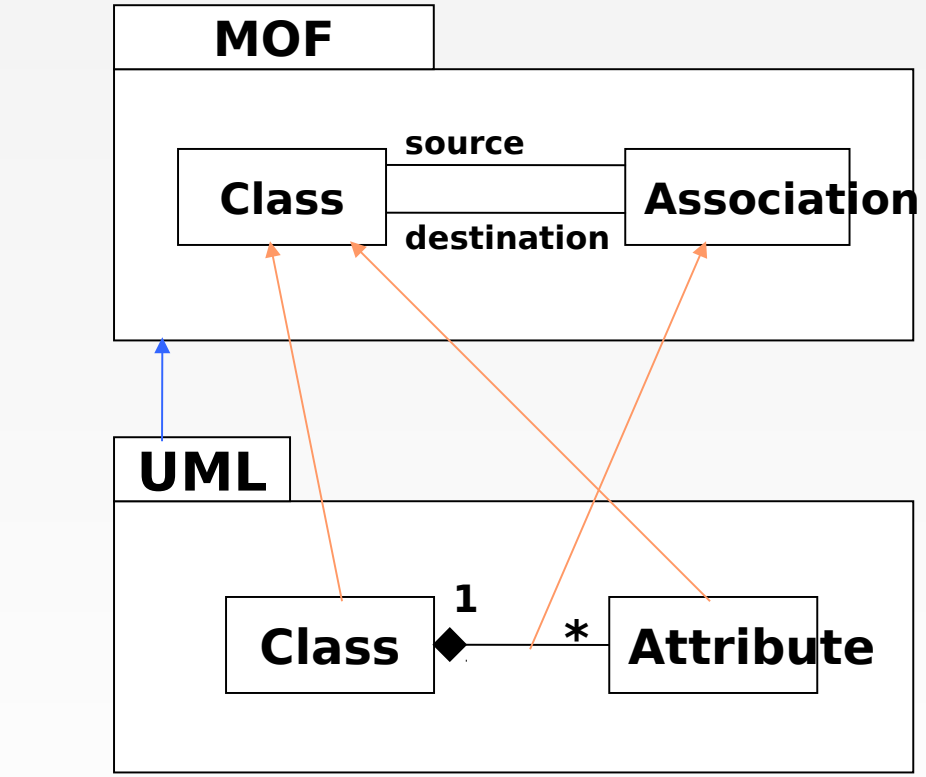


Transparent emprunté à Jean Bézivin

Méta-modèle -> Méta-méta-modèle

Relation
entité → méta-entité

Relation
modèle → méta-modèle

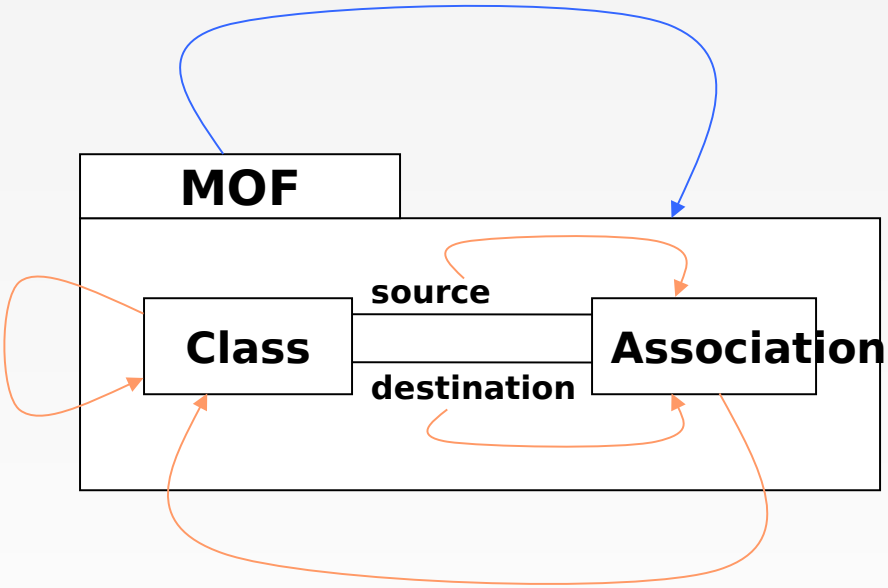


Transparent emprunté à Jean Bézivin

Méta-méta-modèle -> Méta-méta-modèle

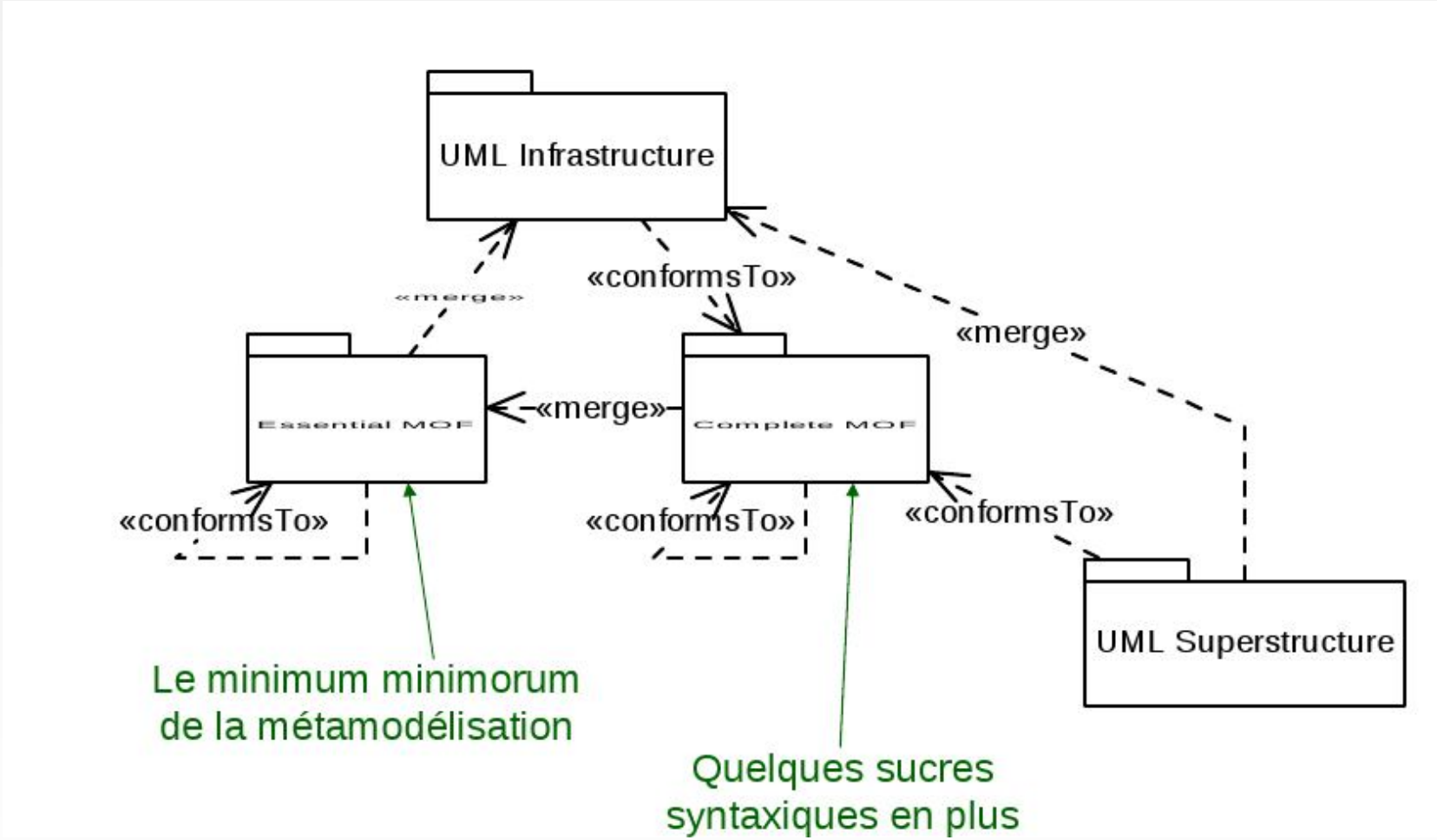
Relation entité → méta-entité

Relation modèle → méta-modèle

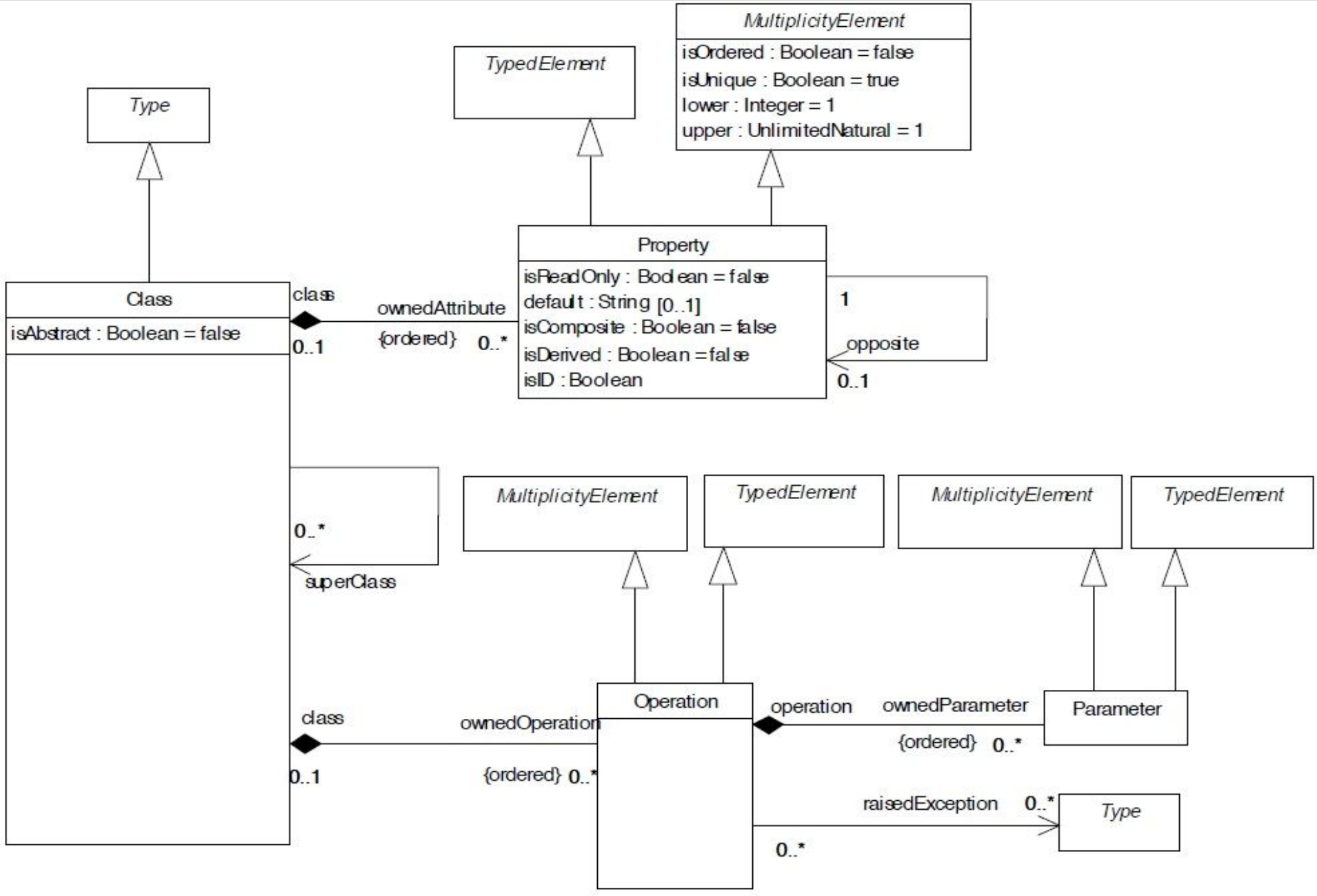


Transparent emprunté à Jean Bézivin

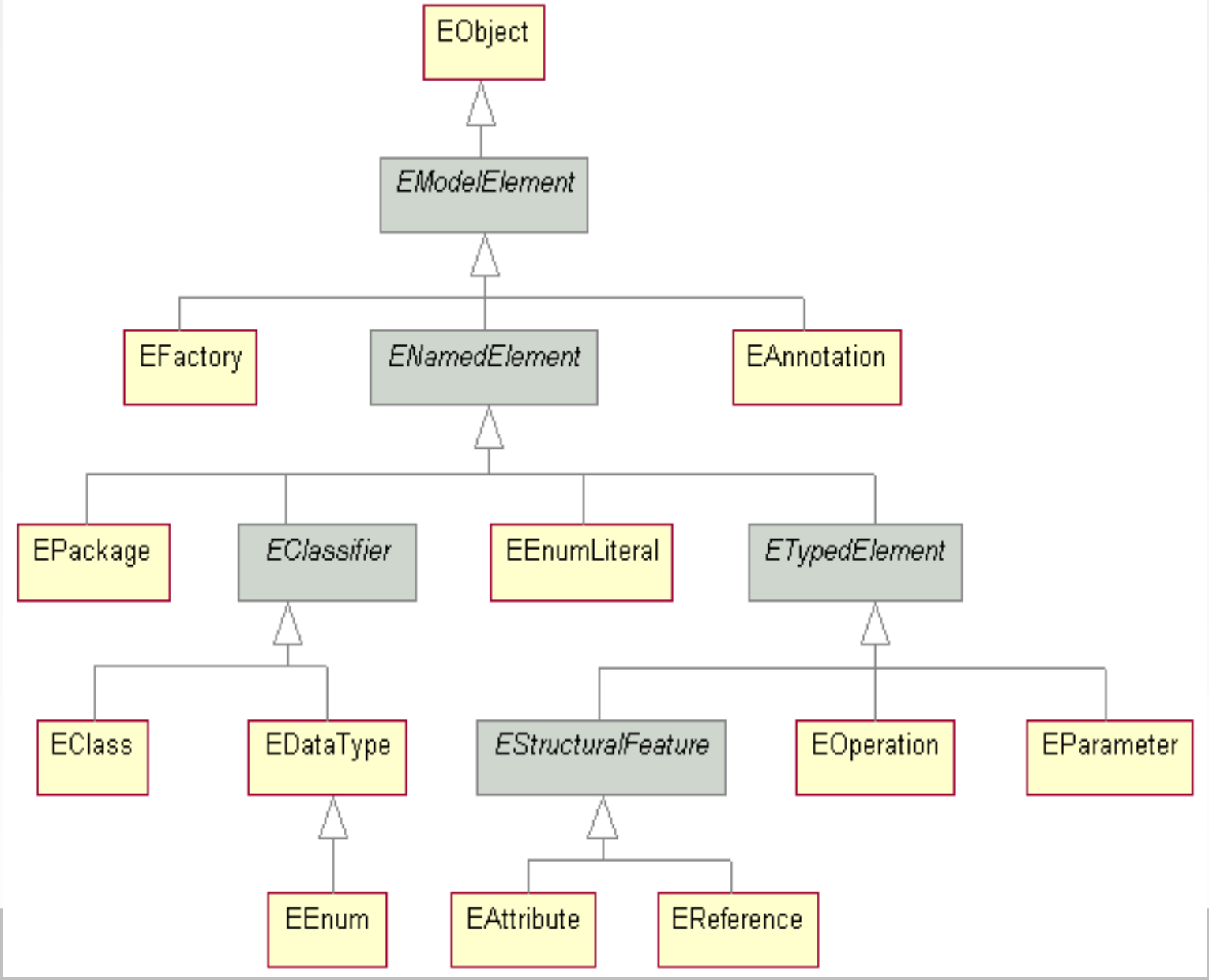
Le MOF

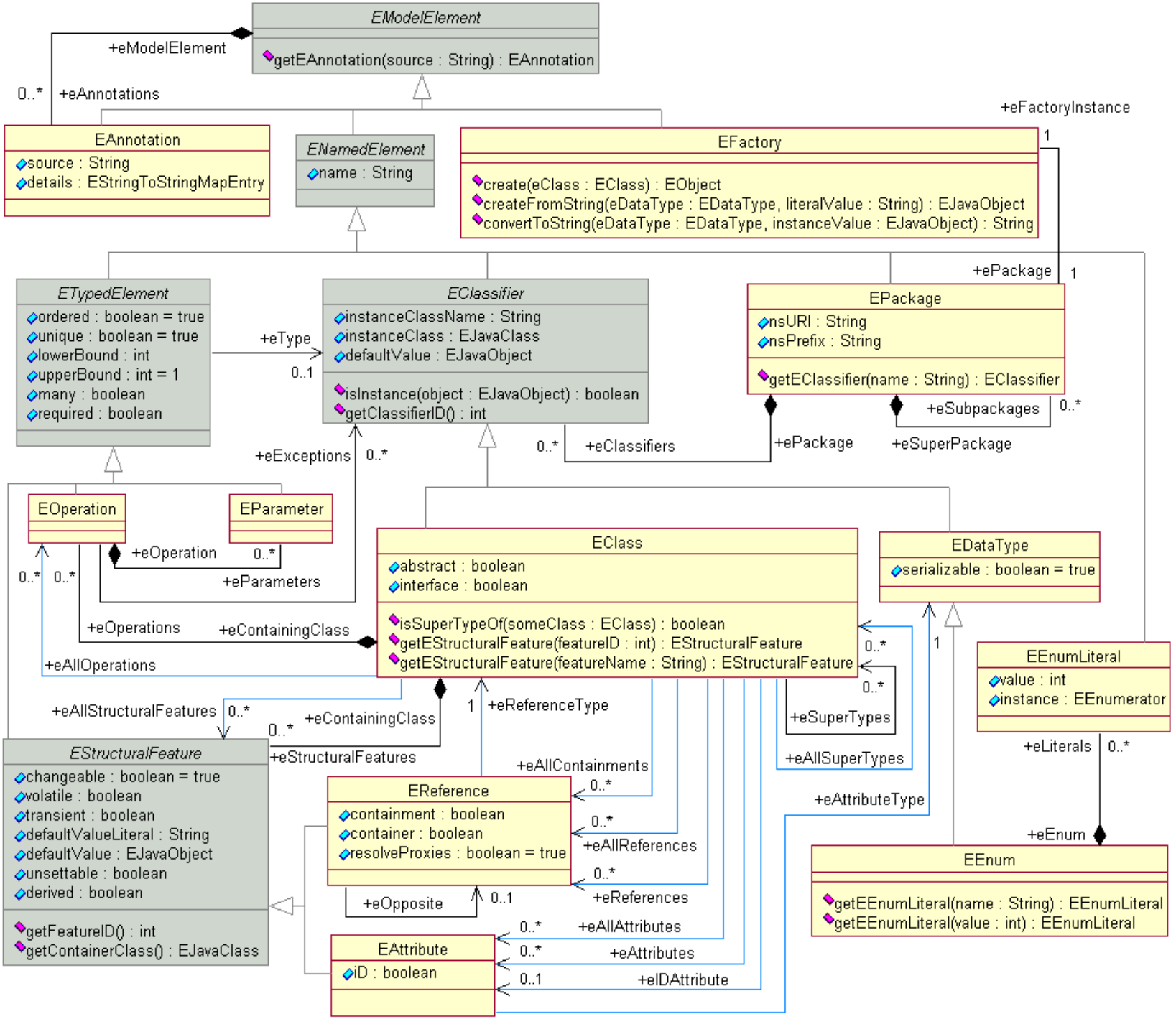


EMOF

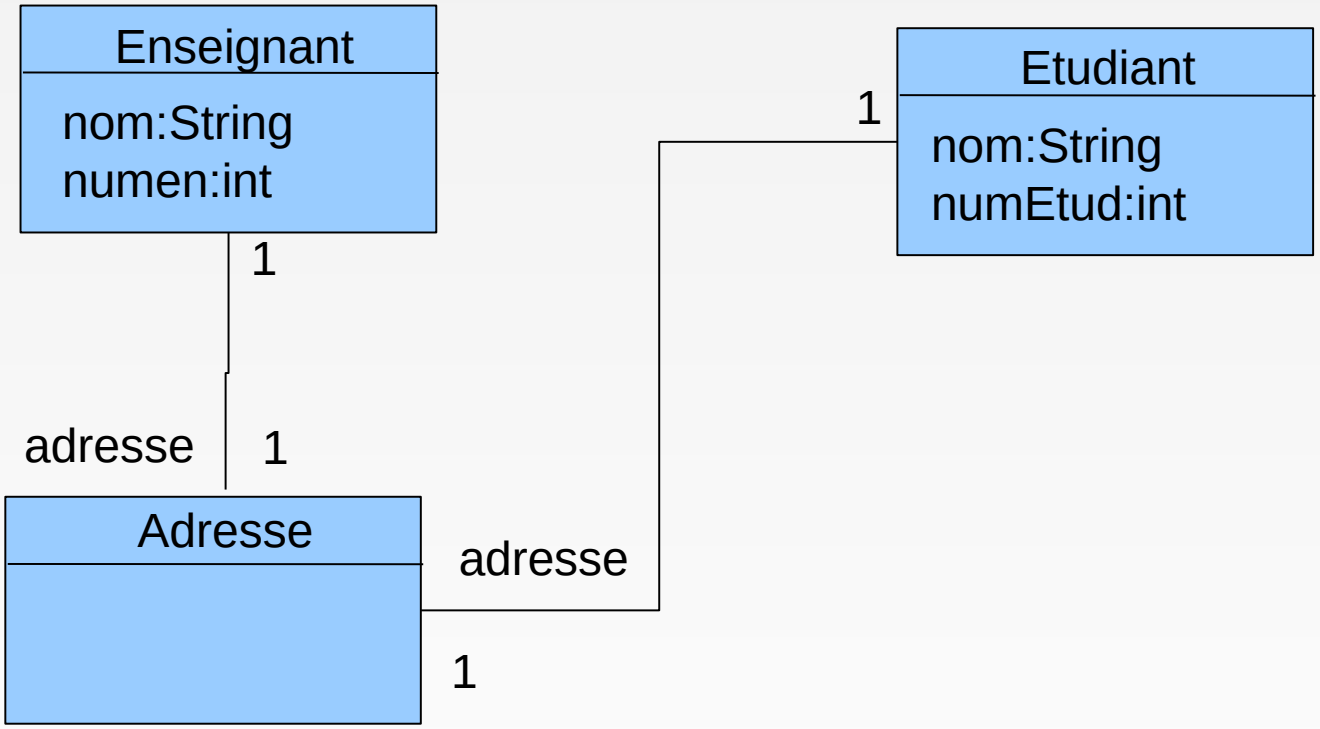

MultiplicityElement
isOrdered : Boolean = false
isUnique : Boolean = true
lower : Integer = 1
upper : UnlimitedNatural = 1
TypedElement
Type
Property
isReadOnly : Boolean = false
default : String [0..1]
isComposite : Boolean = false
isDerived : Boolean = false
isID : Boolean
Class
isAbstract : Boolean = false
class
ownedAttribute
{ordered}
0..1
0..*
1
opposite
0..1
0..*
superClass
MultiplicityElement
TypedElement
MultiplicityElement
TypedElement
Operation
class
ownedOperation
{ordered}
0..1
0..*
operation
ownedParameter
{ordered}
0..*
Parameter
raisedException
0..*
0..*
Type

Ecore


EObject
EModelElement
EFactory
ENamedElement
EAnnotation
EPackage
EClassifier
EEnumLiteral
ETypedElement
EClass
EDataType
EStructuralFeature
EOperation
EParameter
EEnum
EAttribute
EReference

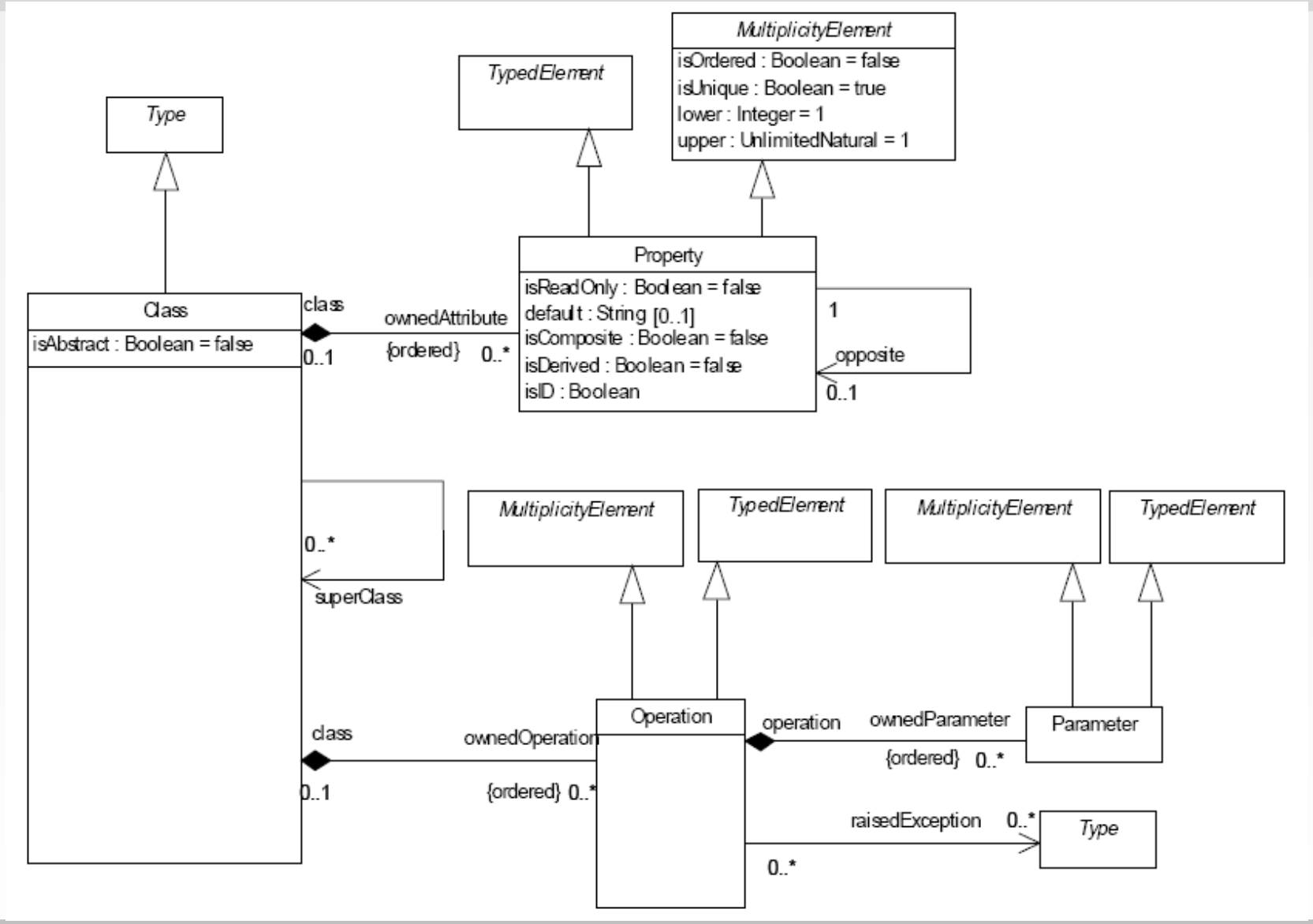

EModelElement
getEAnnotation(source : String) : EAnnotation
+eModelElement
0..*
+eAnnotations
EAnnotation
source : String
details : EStringToStringMapEntry
ENamedElement
name : String
EFactory
create(eClass : EClass) : EObject
createFromString(eDataType : EDataType, literalValue : String) : EJavaObject
convertToString(eDataType : EDataType, instanceValue : EJavaObject) : String
+eFactoryInstance
1
ETypedElement
ordered : boolean = true
unique : boolean = true
lowerBound : int
upperBound : int = 1
many : boolean
required : boolean
+eType
0..1
EClassifier
instanceClassName : String
instanceClass : EJavaClass
defaultValue : EJavaObject
isInstance(object : EJavaObject) : boolean
getClassifierID() : int
+ePackage
1
EPackage
nsURI : String
nsPrefix : String
getEClassifier(name : String) : EClassifier
+eSubpackages
0..*
+eSuperPackage
+eExceptions
0..*
0..*
+eClassifiers
+ePackage
EOperation
EParameter
+eOperation
0..*
+eParameters
0..*
0..*
+eOperations
+eContainingClass
+eAllOperations
EClass
abstract : boolean
interface : boolean
isSuperTypeOf(someClass : EClass) : boolean
getEStructuralFeature(featureID : int) : EStructuralFeature
getEStructuralFeature(featureName : String) : EStructuralFeature
EDataType
serializable : boolean = true
0..*
+eSuperTypes
0..*
+eAllSuperTypes
1
EEnumLiteral
value : int
instance : EEnumerator
+eLiterals
0..*
+eEnum
EEnum
getEEnumLiteral(name : String) : EEnumLiteral
getEEnumLiteral(value : int) : EEnumLiteral
+eAttributeType
+eAllStructuralFeatures
0..*
0..*
+eContainingClass
+eStructuralFeatures
1
+eReferenceType
EStructuralFeature
changeable : boolean = true
volatile : boolean
transient : boolean
defaultValueLiteral : String
defaultValue : EJavaObject
unsettable : boolean
derived : boolean
getFeatureID() : int
getContainerClass() : EJavaClass
+eAllContainments
0..*
EReference
containment : boolean
container : boolean
resolveProxies : boolean = true
0..*
+eAllReferences
0..*
+eReferences
+eOpposite
0..1
0..*
+eAllAttributes
0..*
+eAttributes
0..1
+eIDAttribute
EAttribute
iD : boolean

Intérêt de disposer d'un métamodèle : exemple

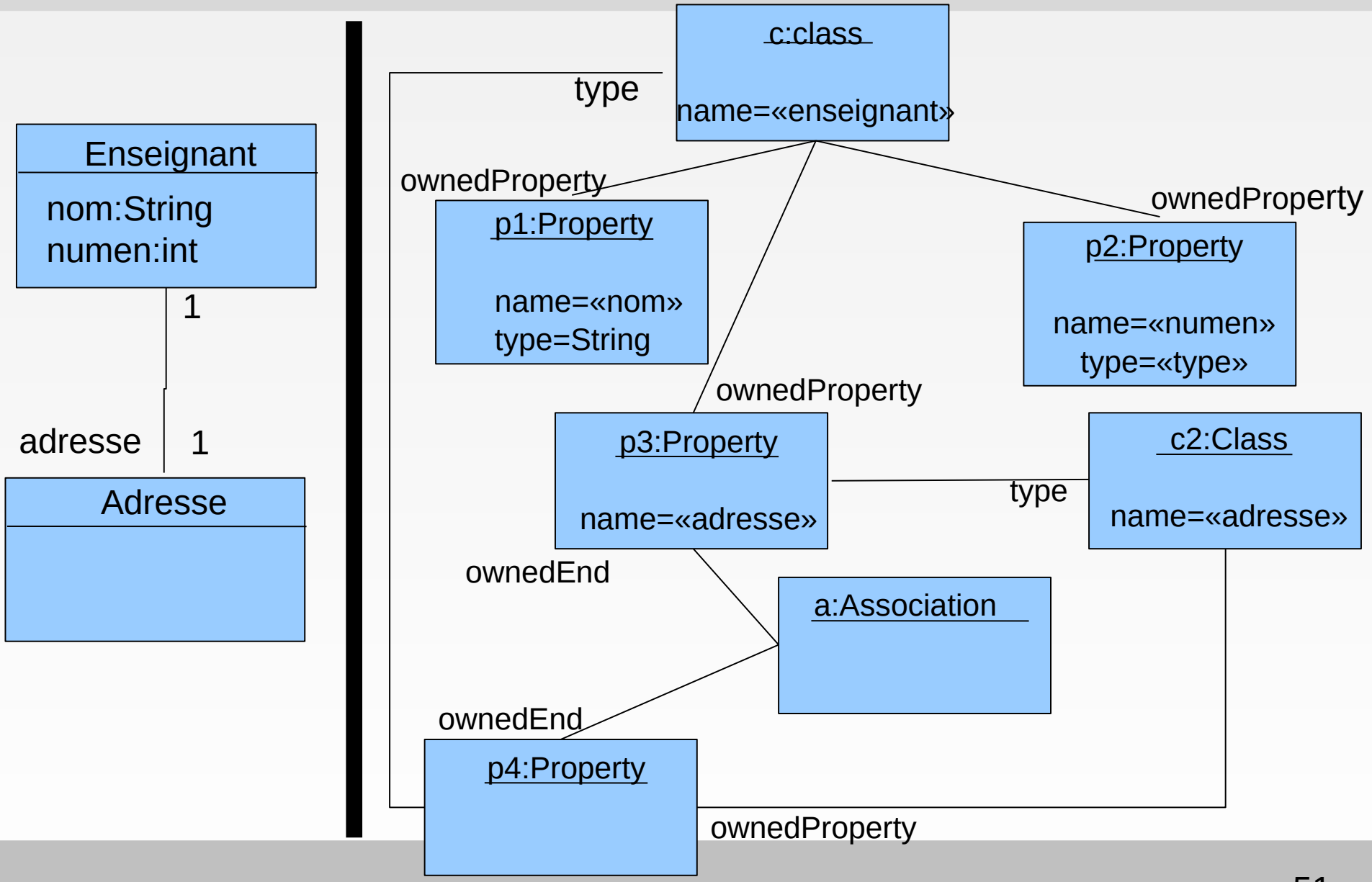


- On voudrait bien restructurer automatiquement
- Ce diagramme est-il autre chose qu'une image ?
- Comment le manipuler par programme ?

Que nous dit le métamodèle UML ?



Le modèle devient une instance de son métamodèle

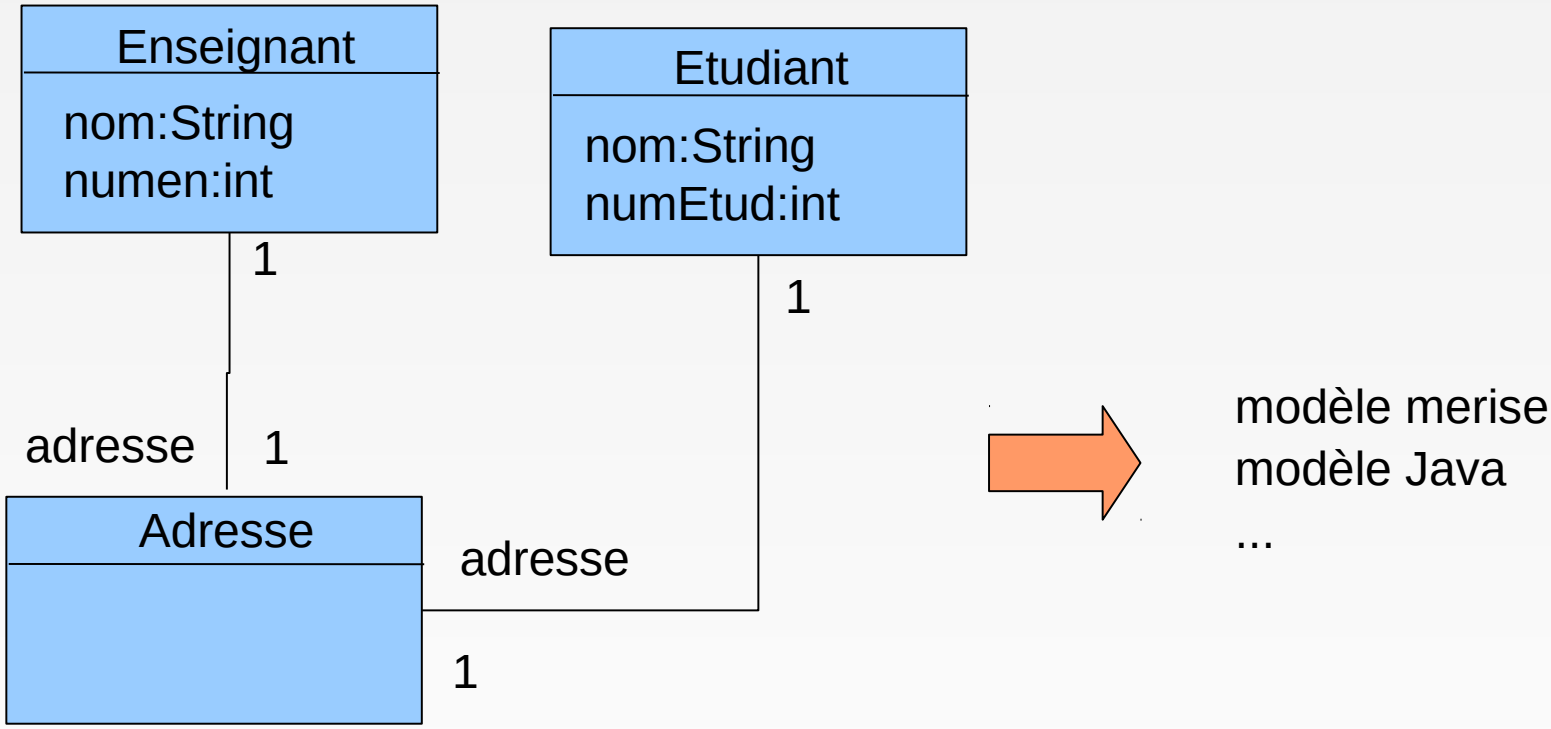


On peut maintenant manipuler l'instance de méta-modèle

```
pour tout c1:Class
  pour tout c2:Class avec c1!=c2
    pour tout p1:Property tq p1 est dans c1.ownedAttribute
      pour tout p2:Property tq p2 est dans c2.ownedAttribute
        si p1.name==p2.name et p1.type==p2.type
        alors abstractionDetectee(c1,c2,p1,p2)
```

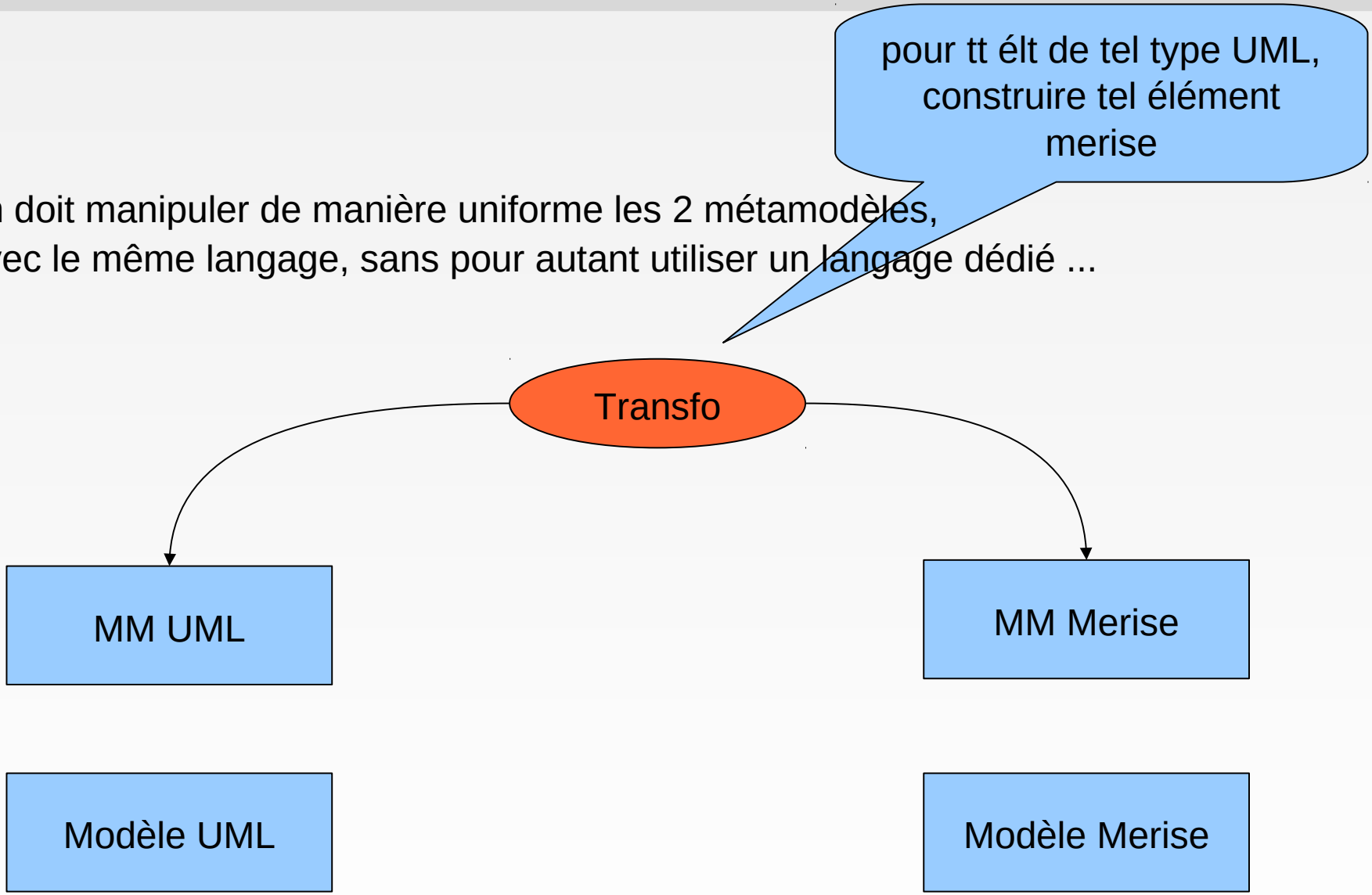
- On pourra alors si c1 et c2 n'ont pas d'ancêtres communs :
  - créer une nouvelle classe c
  - y déplacer tout ce qui est commun à c1 et c2
  - créer une généralisation entre c1 et c, et entre c2 et c
- Si on a un ancêtre commun direct, on y déplace ce qui est commun à c1 et c2.
- Ou alors on utilise une technique de transformation + maligne

Intérêt de disposer d'un métamétamodèle

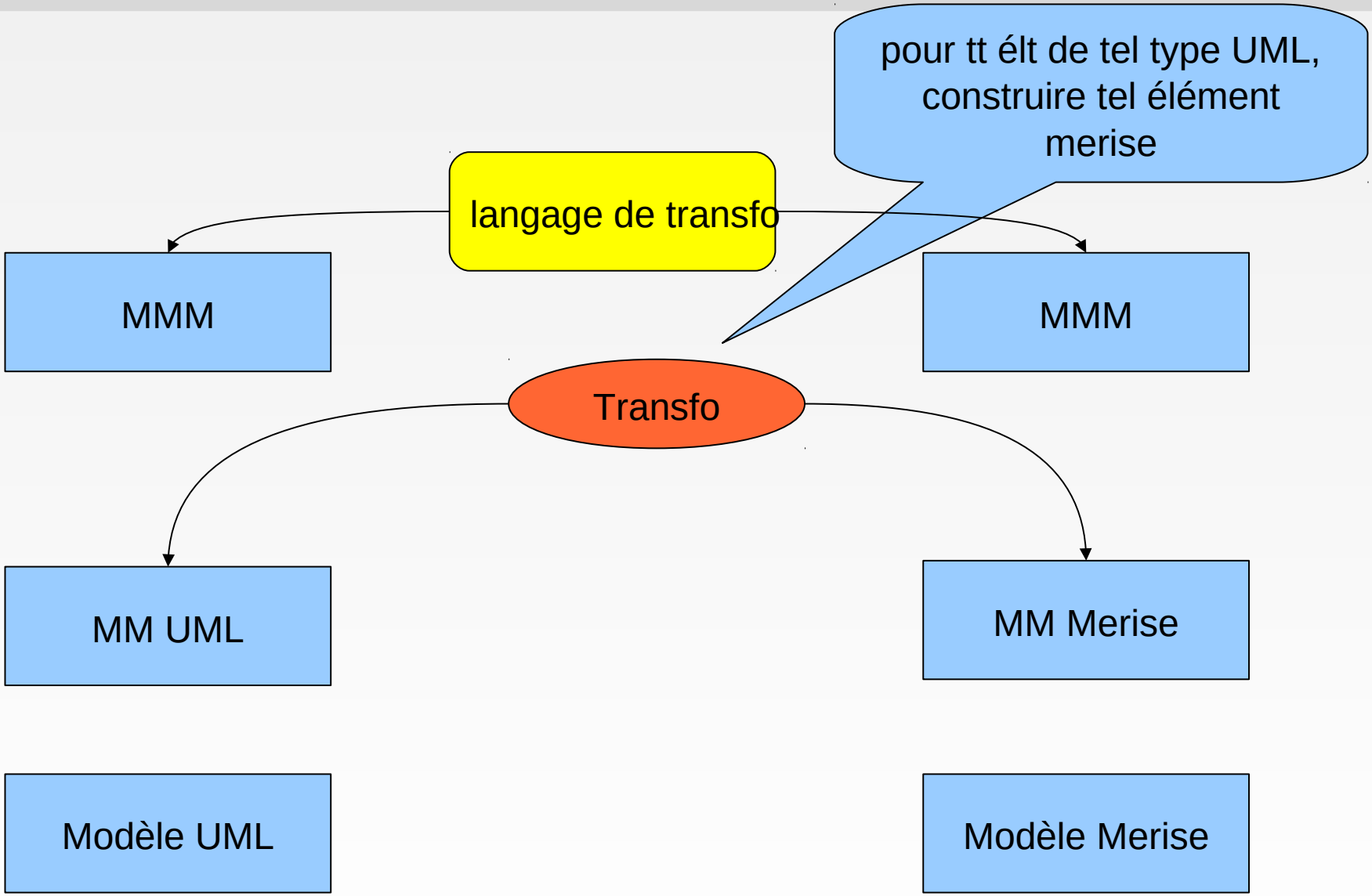


Intérêt de disposer d'un métamétamodèle

on doit manipuler de manière uniforme les 2 métamodèles,
avec le même langage, sans pour autant utiliser un langage dédié ...



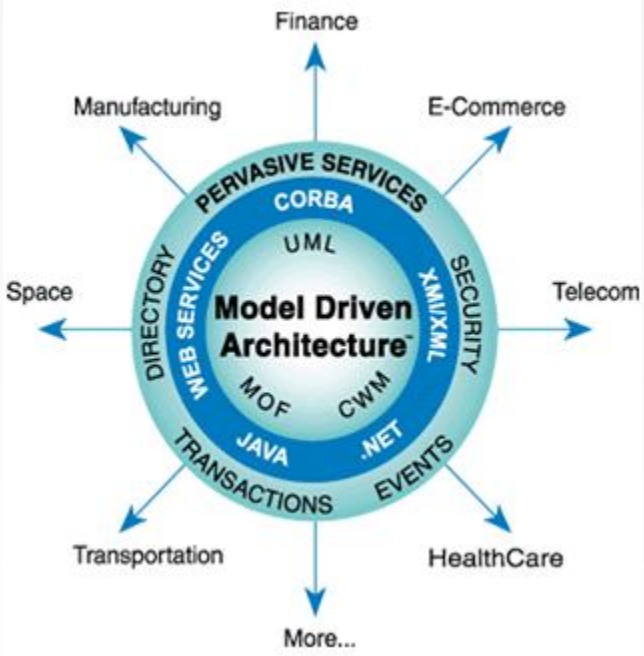
Intérêt de disposer d'un métamétamodèle



Le MDA

L'OMG et le MDA

"OMG is in the ideal position to provide the model-based standards that are necessary to extend integration beyond the middleware approach… Now is the time to put this plan into effect. Now is the time for the Model Driven Architecture."



Richard Soley and the OMG staff,

MDA Whitepaper Draft 3.2

November 27, 2000

Model Driven Architecture

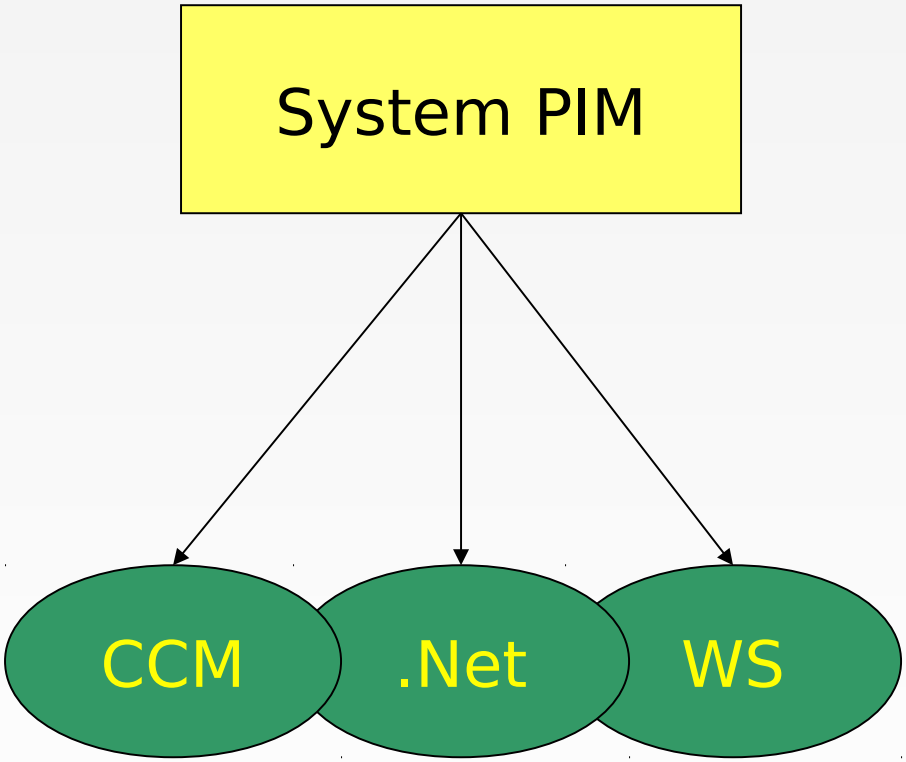
- Définition de méta-modèles
  - Standardisation d'un domaine d'activité
  - Standardisation des technologies
- Définition d'une méthodologie
  - Démarche à suivre pour produire une application, dans une technologie, à partir d'un modèle abstrait
- Il existe différents types de modèles

Model Driven Architecture

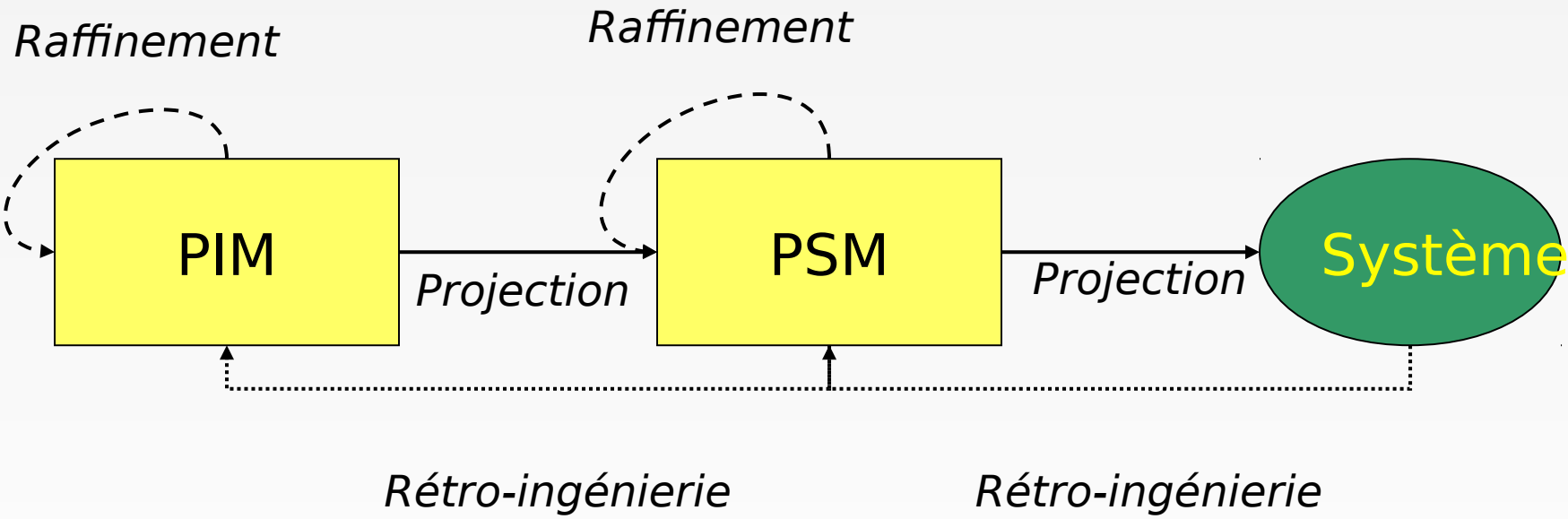
- CIM: Computational Independant Model
- PIM: Platform Independant Model
  - Définit la structure et les fonctions d'un système
  - Indépendant des détails technologiques
- PSM: Platform Specific Model
  - Définit la mise en œuvre de la structure et des fonctions dans une technologie particulière
- PDM: Platform Definition Model
  - Définit une technologie particulière

Model Once, Generate Everywhere

- Modèle indépendant de toute technologie
- Génération de code pour plusieurs plates-formes

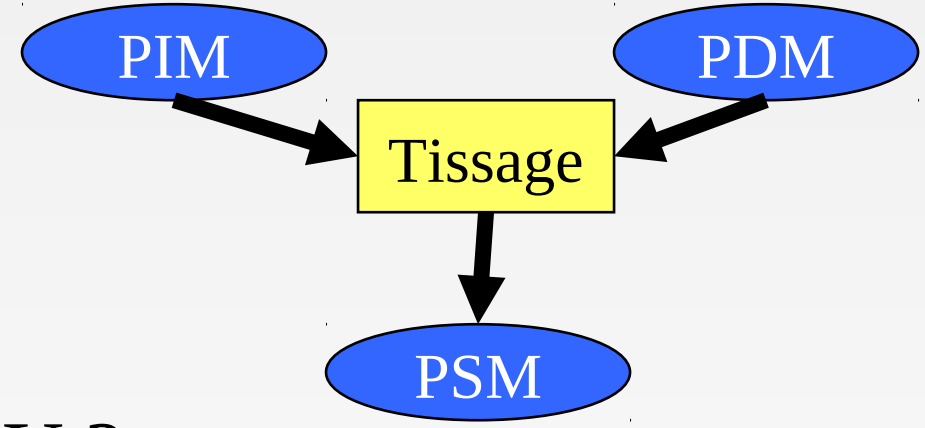


Un processus logiciel naïf

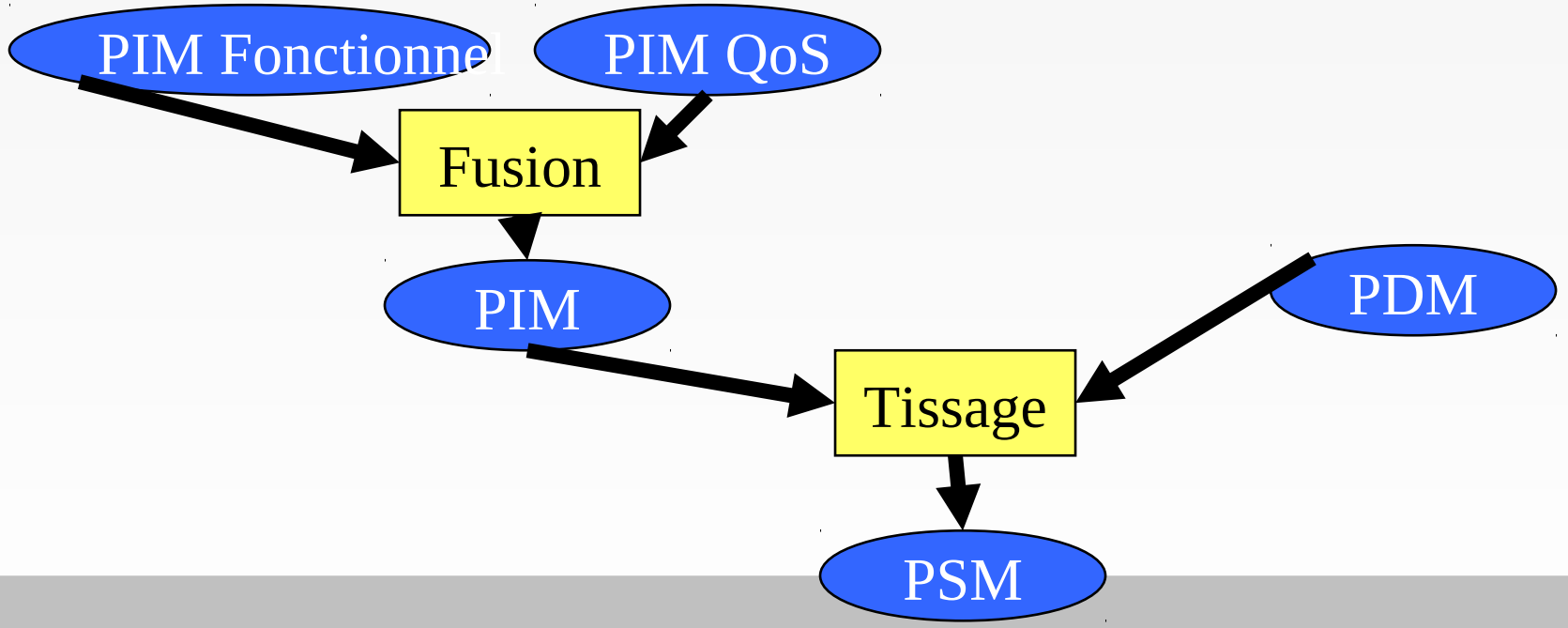


De PIM à PSM ?

- Cycle en Y ?



- Ou cycle en double Y ?



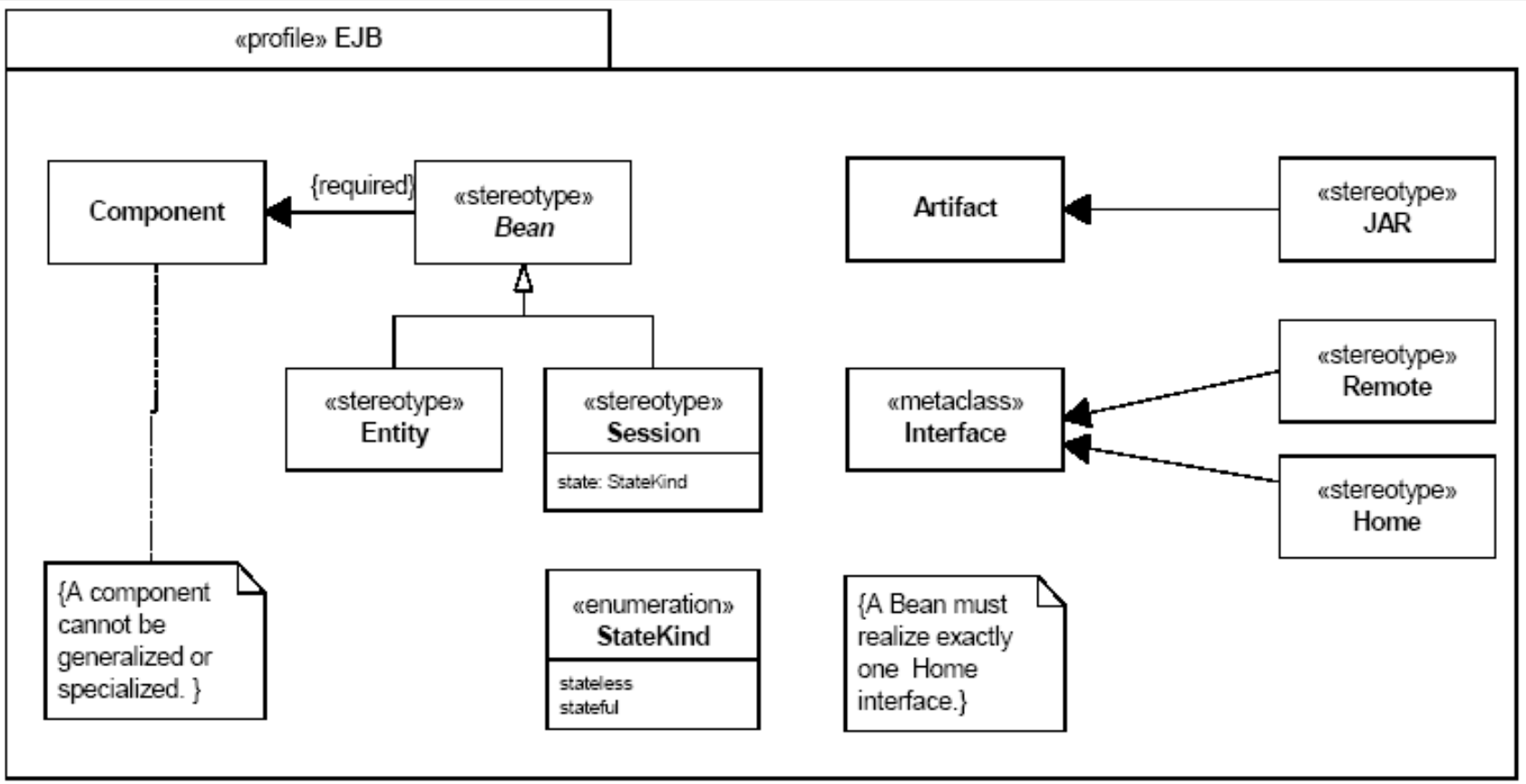
Le MDA et les profils UML

- Profil UML
  - mécanismes standard d'extension d 'UML
  - permet la spécialisation d'un schéma UML pour un domaine particulier
  - Ensemble cohérent de stéréotypes, valeur marquées (tagged value) et contraintes
  - Principe général : on n'ajoute pas de méta-classes mais des annotations aux méta-classes UML existantes.
  - Des profils standards existent : le profil CORBA (OMG), le profil EJB 1.0 (JCP), . . .

Le MDA et les profils UML

- Les stéréotypes :
  - Un élément stéréotypé porte la sémantique du stéréotype
    - exemple : <<persistant>>, <<interface>>
- Les valeurs marquées :
  - Ajouter des informations sur les éléments de modèle
    - Exemple : EJBSessionType, le type d'EJB Session (Stateless, Stateful)
- Les contraintes :
  - relations entre les stéréotypes et les valeurs marquées
  - sémantique du profil
    - exemple : Toute classe stéréotypée « EJBRemoteInterface » doit être réalisée par une classe stéréotypée « EJBImplementationClass »

Exemple : profil EJB



Le MDA et les profils UML

- Approche MDA par profils
  - on ne définit pas de nouveau métamodèle, on adapte le méta-modèle UML …
  - on définit des règles de génération
  - Exemple d ’outil : UML Profile Builder
- Problèmes :
  - contorsions pour se rapprocher d ’UML
  - on dépend d’un outil propriétaire

Conclusions

Gestion de l’obsolescence technologique ?

- On ne peut pas capitaliser en se basant sur le code
  - trop dépendant des technologies
- On a besoin d’abstraction : les modèles
- Modèle métier
  - indépendant des technologies
  - peut être projeté vers un modèle d ’implémentation
- Vision MDA : PIM et PSM

Une nouvelle évolution de l'ingénierie du logiciel

- Qui était dans l'air depuis quelques temps …
- Une approche innovante et prometteuse
  - maturité ?

Le MDE, utopie ou réalité ?

- Attention ne pas confondre avec le Michel Drucker Experience, groupe de rock psychédélique belge
- Écrire des programmes qui génèrent des programmes ?
  - Les programmeurs deviendront-ils programmeurs de transformations de modèles ?
  - De plus en plus d'outils basés modèles

Les méthodes de modélisation et le MDE

- Des méthodes de modélisation sont utilisées depuis longtemps
  - Merise (France), SSADM (UK), Information Engineering (US)
  - Modèles = base d ’activité d ’ingénierie humaine
  - Impact mitigé, pratique industrielle basée sur le code
  - Modèles contemplatifs, codage = cœur du métier d ’informaticien
  - Pas de liaison avec le code source
    - développement de techniques ad hoc (wizard génération de code, fichier xml pour le déploiement, …)

Les méthodes de modélisation et le MDE

- MDE : rendre les modèles productifs
  - modèle interprétable et manipulable par une machine.
- Caractéristique originale de MDE
  - ~~Utilisation systématique de modèles~~
  - Utilisation systématique de méta-modèles

Sources

- Support inspiré de ceux de :
  - Jean-Marc Jézéquel
  - Jean Bézivin
  - Frédéric Fondement
  - Raphaël Marvie