Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Centre Universitaire de Mila
Laboratoire MISC, Université Constantine 2–Abdelhamid Mehri

Institut des Sciences et de la Technologie

Département: MI, Master 1 : STIC

Module: Ingénierie Des Logiciels (IDL)

Chapitre02 : Ingénierie Dirigée par les Modèles (IDM)

Dr MEGHZILI Said
Département MI
said.meghzili@univ-mila.dz

Résumé

Prérequis

- ***Notions sur le génie logiciel***

Objectifs du chapitre

- **L'IDM (Ingénierie dirigée par les modèles) est le domaine de l'informatique mettant à disposition des outils, concepts et langages pour créer et transformer des modèles. Ce que propose l'approche de l'ingénierie des modèles (IDM, ou MDE en anglais pour Model Driven Engineering) est simplement de mécaniser le processus que les ingénieurs expérimentés suivent à la main.**
- **L'intérêt pour l'IDM a été fortement amplifié lorsque l'organisme de standardisation OMG (Object Modeling Group) a rendu publique son initiative MDA (Model Driven Architecture).**
- **Ce cours permettra aux étudiants d'intégrer les principes de l'approche MDA pour le développement d'applications logicielles. Il présentera aussi un tour d'horizon des outils liés à MDA**
- **Il présente les différentes étapes d'application du MDA et de la transformation de modèles.**

Sommaire

Introduction: Évolution des technologies

- **Les technologies logicielles sont en Évolution permanente**

  Exemple dans les* systèmes distribués *:

  Évolution pour faire communiquer et interagir des éléments distants:

  1-C et sockets TCP/UDP →2-C et RPC →3-C++ et CORBA →4-Java et RMI

  →5-C# et Web Service→6-Java et EJB →7- A suivre .....

  Avantages: développement plus rapide et plus efficace.

- **Si on veut profiter des nouvelles technologies et de leurs avantages :**

  →Nécessite d'adapter une application à ces technologies.

  →Le coût de cette adaptation est très élevé : on doit réécrire le code d'application

Exemple :Application de calculs scientifiques distribués sur un réseau de machines
Passage de C/RPC (Paradigme procédural)à Java/EJB (objet/composant):

→Impossibilité de reprendre le code existant (différence des Paradigmes)

- **Nécessité de découpler la logique métier et de la mise en œuvre technologique**

→C'est l'un des principes fondamentaux de l'ingénierie dirigée par les modèles :
Séparation des préoccupations

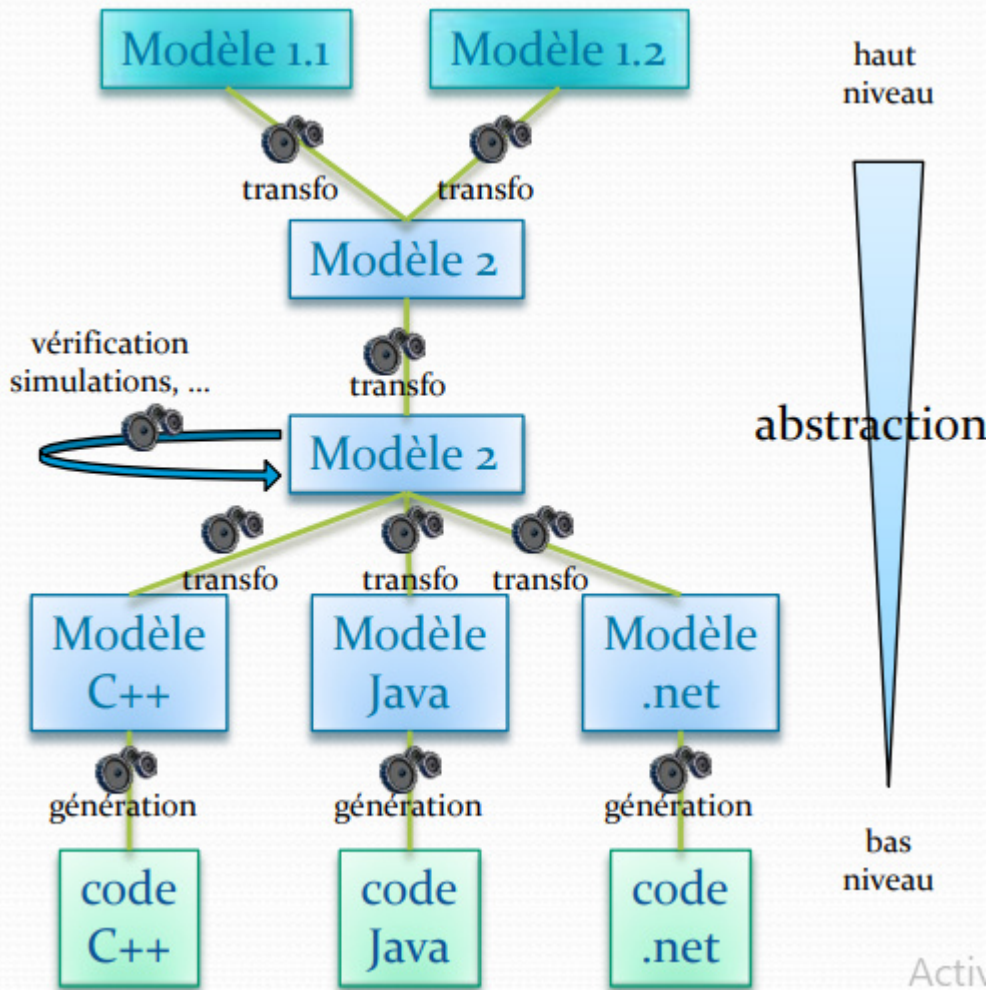
Introduction: Que propose l'IDM ?

- **Besoin de modéliser/spécifier**
  - **A un niveau abstrait la partie métier**
  - **La plate-forme de mise en oeuvre**
  - **De projeter ce niveau abstrait sur une plateforme**

- **Modéliser les applications**

à un haut niveau d'abstraction

- Place le modèle au cœur du processus de conception
- transformation de modèles pour passer d'un niveau à l'autre
- Puis génère le code de l'application à partir des modèles
- possibilité de contrôler, simuler et tester à différents niveaux



Introduction

- ***« Modéliser est le futur, et je pense que les sociétés qui travaillent dans ce domaine ont raison » Bill Gates***

- ***« Obtenir du code à partir d'un modèle stable est une capacité qui s'inscrit dans la durée » Richard Soley (OMG)***

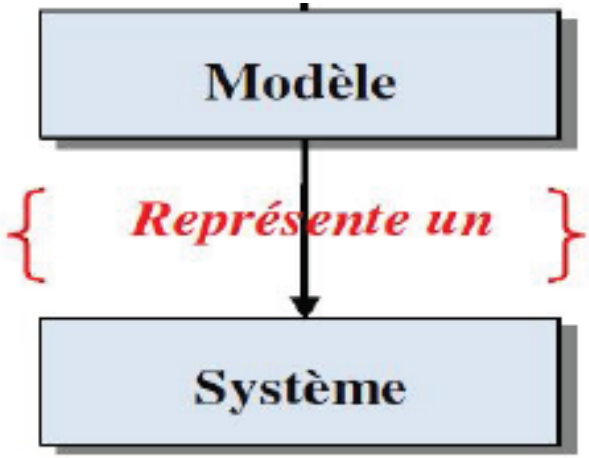
Introduction

- **L'Ingénierie Dirigée par les Modèles (IdM, ou MDE en anglais pour Model Driven Engineering, MDD pour Model Driven Development, MDSD pour Model Driven Software Development)**
  - Apporte des solutions concrètes afin de mieux maitriser cette complexité mais aussi d'augmenter la productivité, la qualité, la réutilisabilité et l'évolution de ces systèmes d'information.
- **Raffinement systématique des modèles, des modèles abstraits aux modèles plus concrets**
- **Domaine croissant de recherche et en industrie.**
- **Il considère les modèles comme les éléments de base dans la production, le fonctionnement et l'évolution des systèmes d'information.**
- **L'IdM raisonne entièrement à ce niveau d'abstraction et non plus à celui des langages de programmation classiques.**
- **Une application sera alors générée en tout ou partie à partir de modèles, en utilisant notamment des techniques de transformation pour opérer la conversion entre le code et les modèles et vice-versa.**

Sommaire

2-Modèles

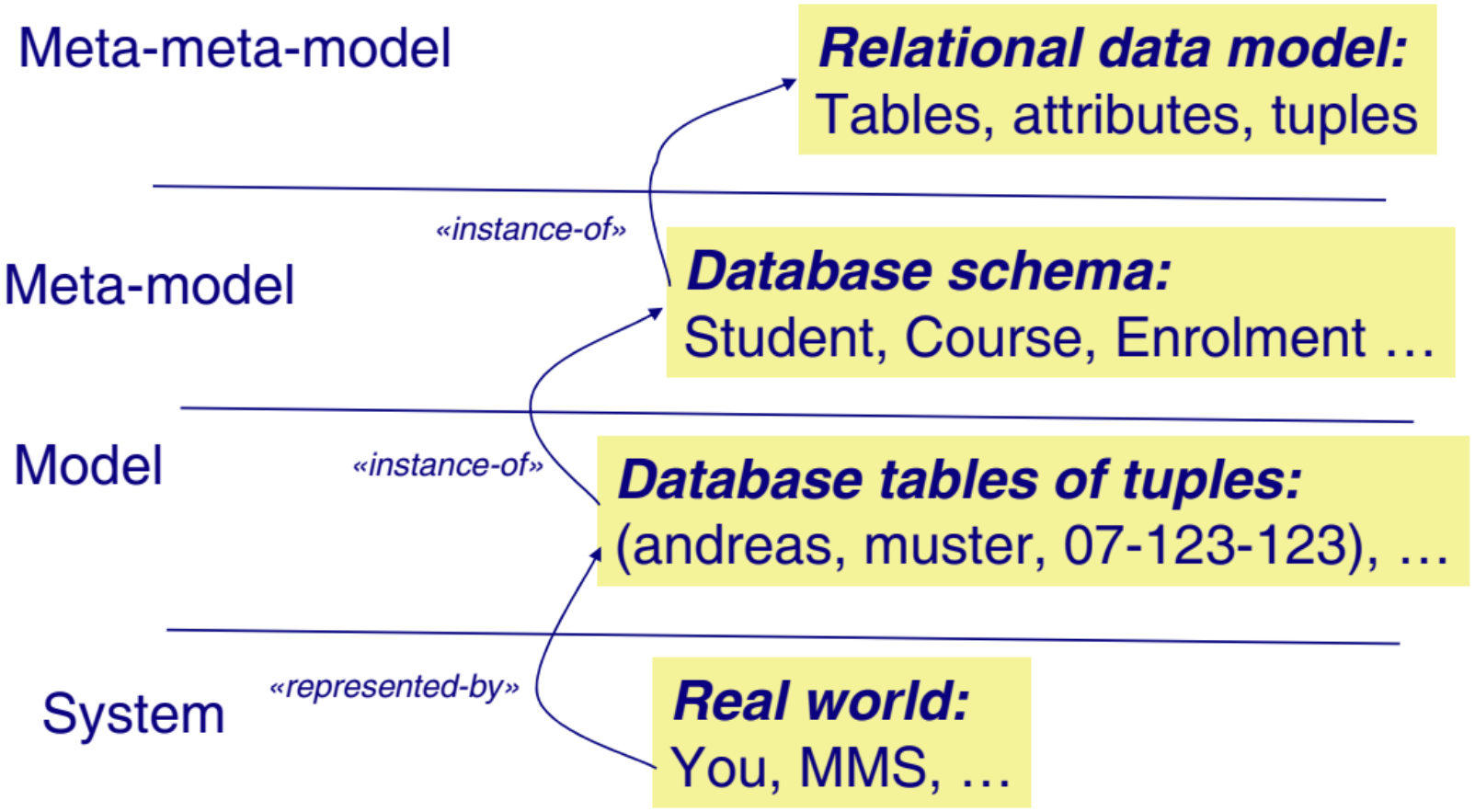
- **Qu'est-ce qu'un modèle?**
  - **Description / abstraction des objets du monde réel**
  - **Quelque chose avec une méta description de la façon dont il devrait être structuré**
  - **Objets et relations (un graphique?)**
  - **Vue simplifiée d'un système**



2-Modèles

- **But d'un modèle**
  - Faciliter la compréhension d'un système complexe
  - Simuler le fonctionnement d'un système complexe
- **Exemples de Modèles**
  - Un réseau de Petri,
  - Un diagramme UML
  - Modèle Entité/Association
  - Modèle météorologique

- **Exemple: Bases de Données**

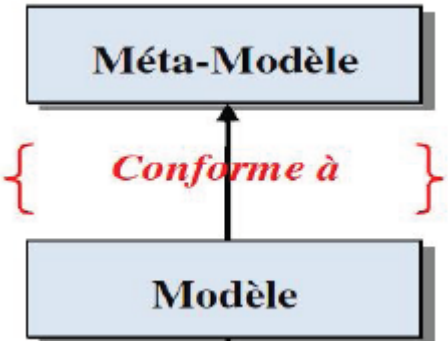


Sommaire

3-Méta-modèles

Qu'est-ce qu'un méta-modèle?

- **Pour passer à une vision productive, il faut que les modèles soient bien définis**
- **La définition d'un langage de modélisation prend la forme d'un modèle, appelé Méta-modèle**
- **Un méta-modèle est un modèle qui définit précisément les concepts manipulés dans les modèles ainsi que les relations entre ces concepts**
- **Base de l'IDM permettant de développer des outils capables de manipuler les modèles**
  - **Assurer la correction syntaxiques des modèles**



3-Méta-modèles

- **But de la méta-modélisation**
  - Définir des langages de modélisation ou des langages de manière générale
- **Architecture MOF de l'OMG**
  - 4 niveaux de (méta)modélisation
- **Syntaxes abstraite et concrète**
- **Définition de méta-modèles**
  - Profils UML
  - MOF
  - Ecore

❑*Principales normes* de *modélisation OMG*

- **MOF :** Meta-Object Facilities

  Langage de définition de méta-modèles

- **UML :** Unified Modelling Language

  Langage de modélisation

- **CWM :** Common Warehouse Metamodel

  Modélisation ressources, données, gestion d'une entreprise

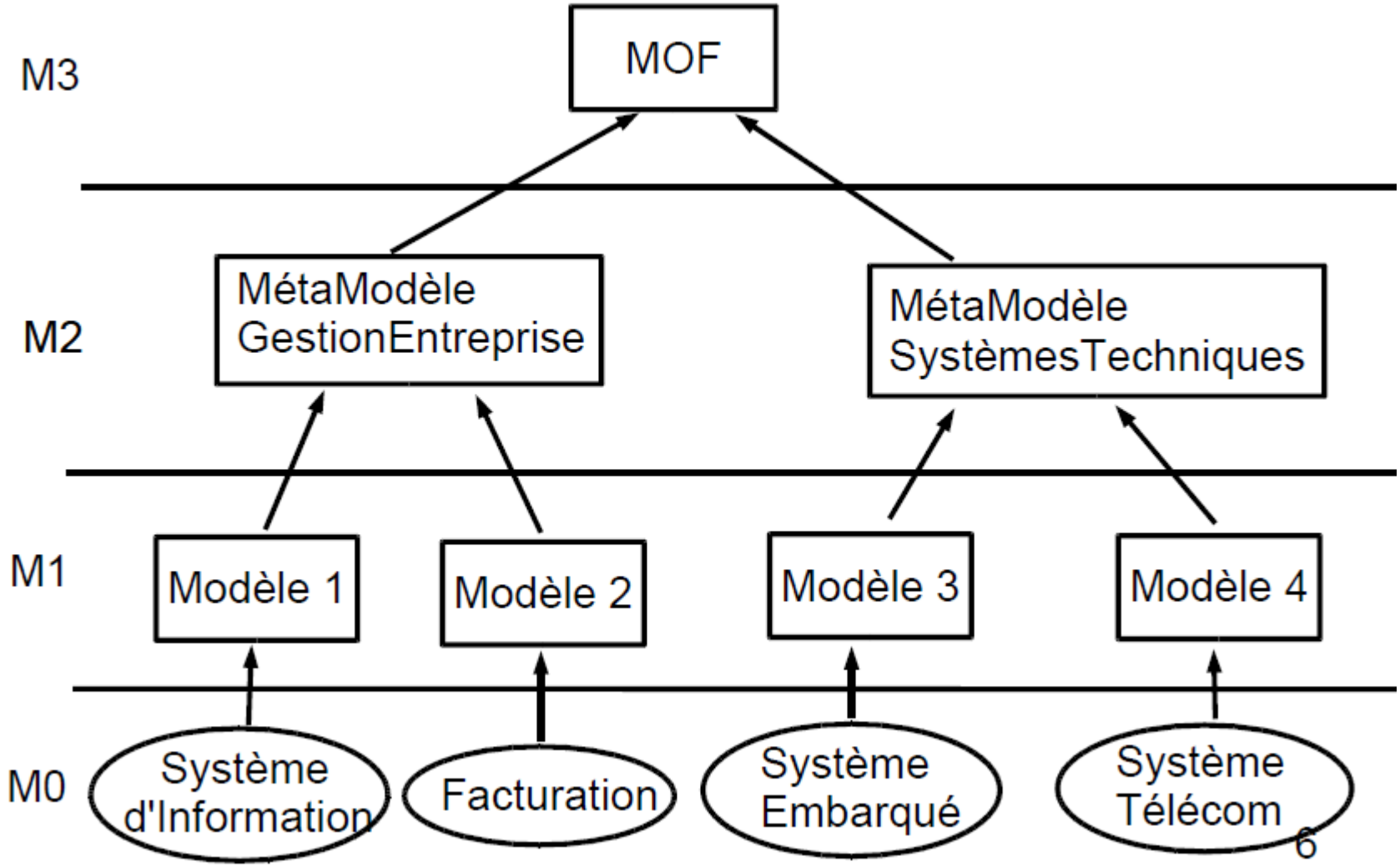
- **OCL :** Object Constraint Language

  Langage de contraintes sur modèles

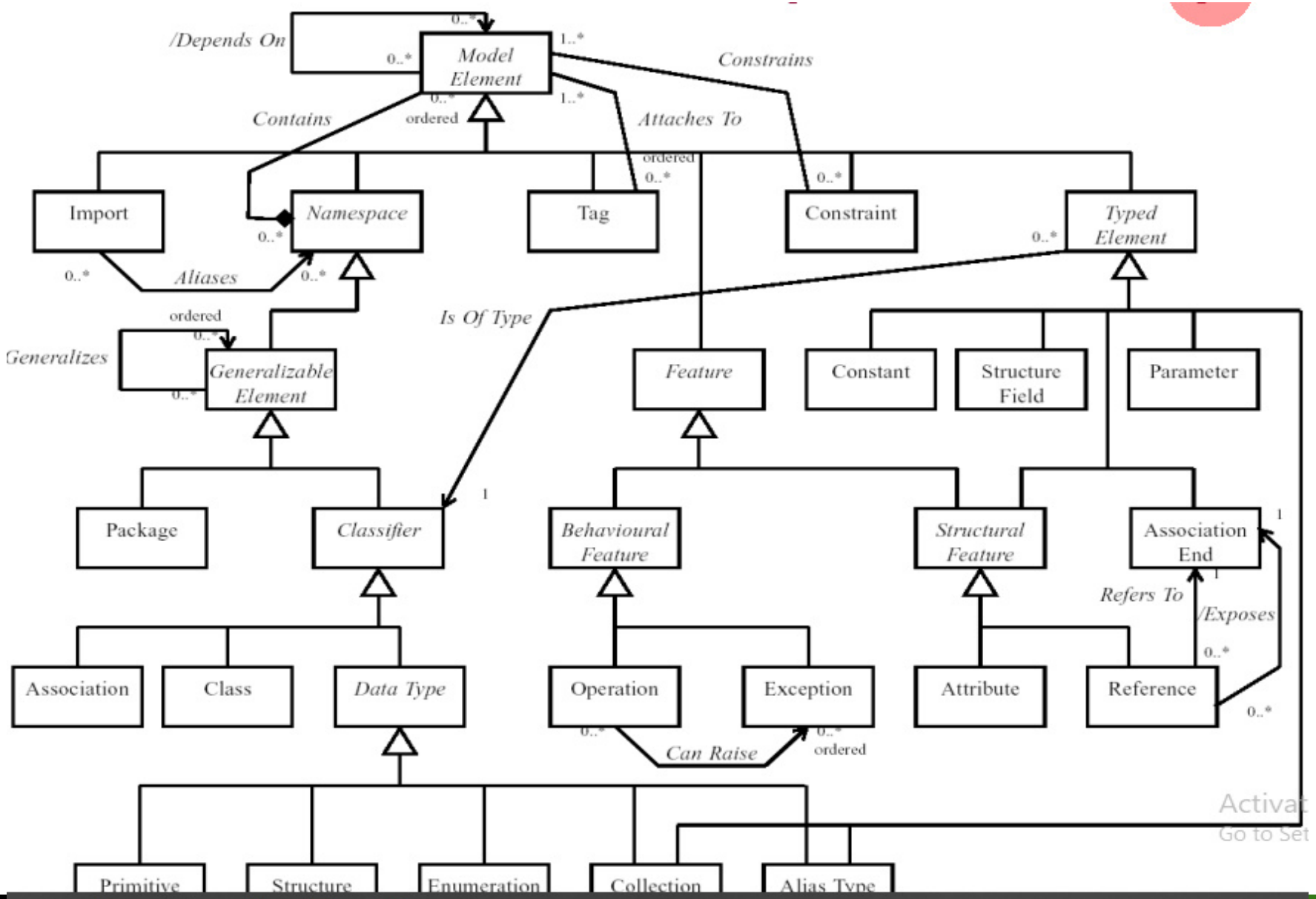
- **XMI :** XML Metadata Interchange

  Standard pour échanges de modèles et méta-modèles entre outils

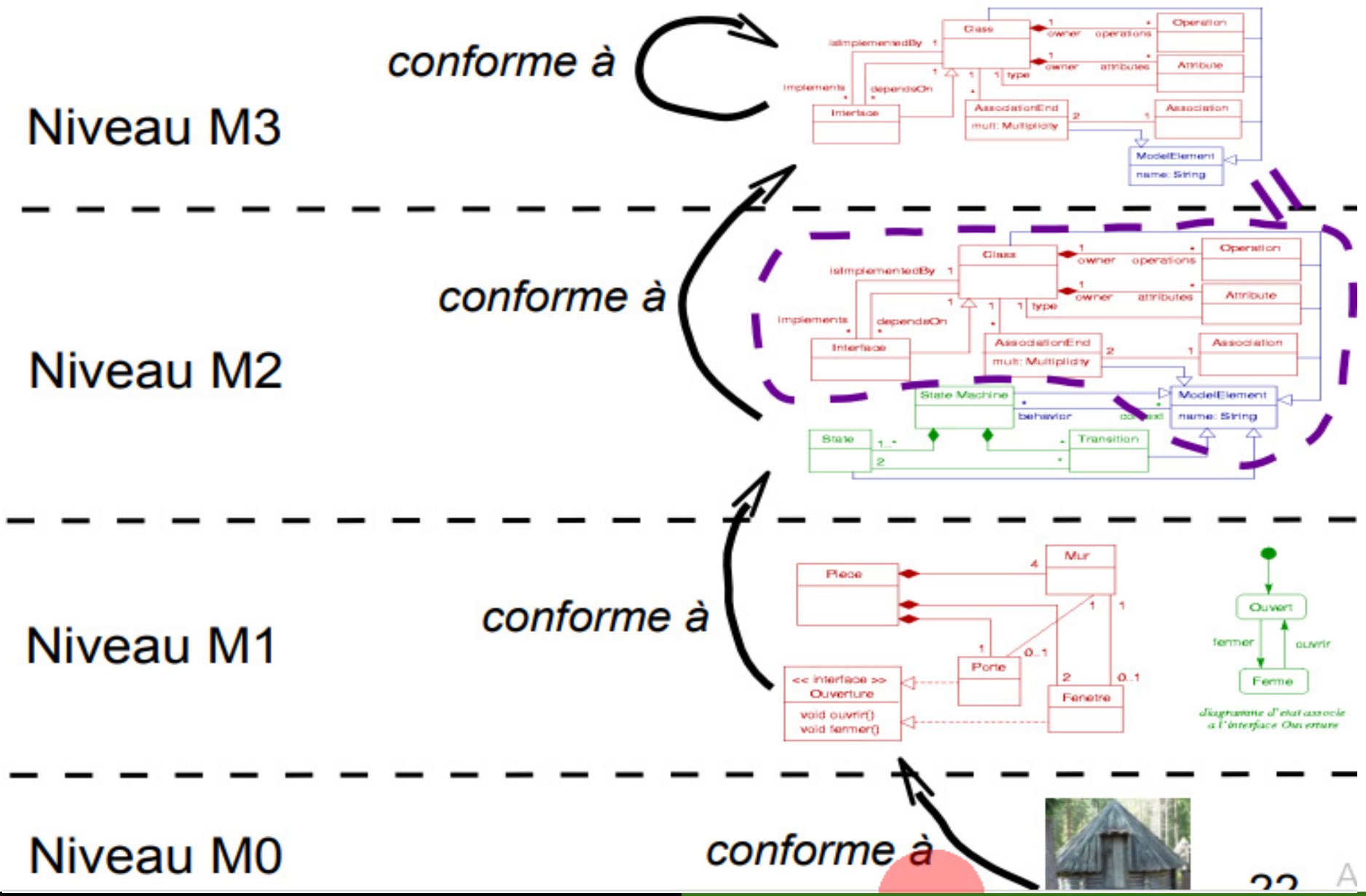
3-Méta-modèles



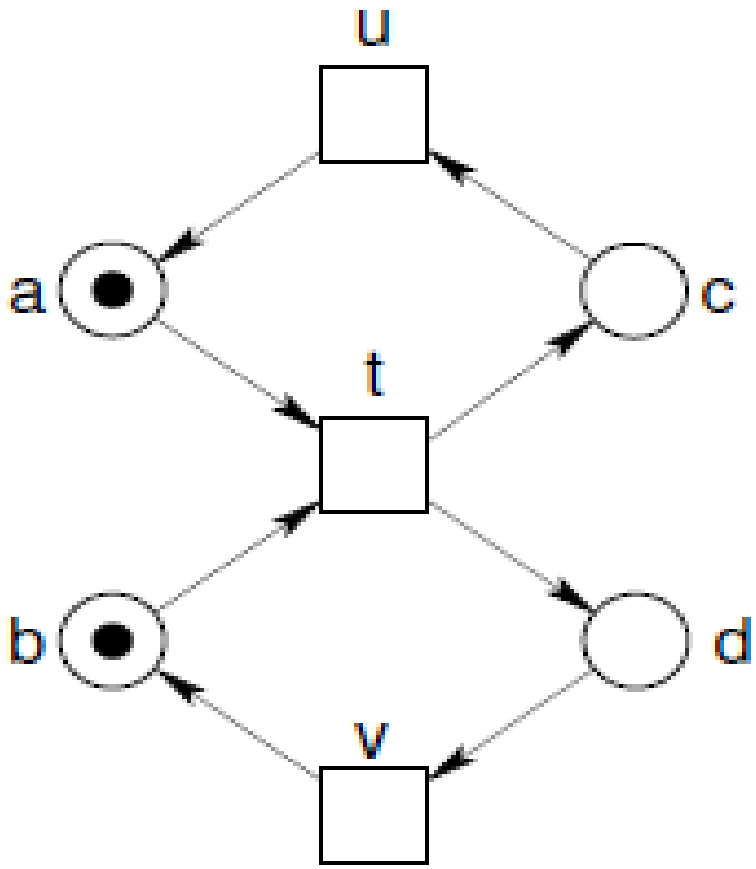
3-Méta-modèles : MOF (version 1.4)



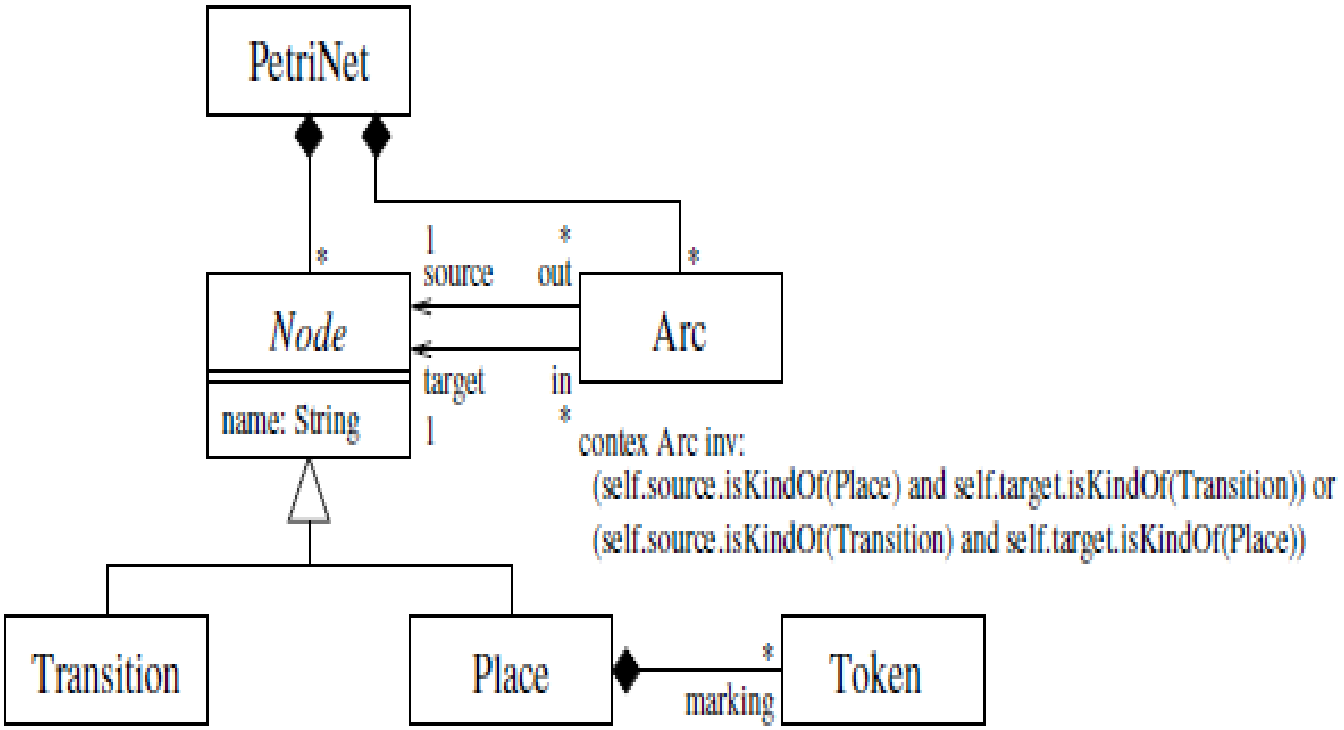
3-Méta-modèles:Exemple Hiérarchie de Modélisation



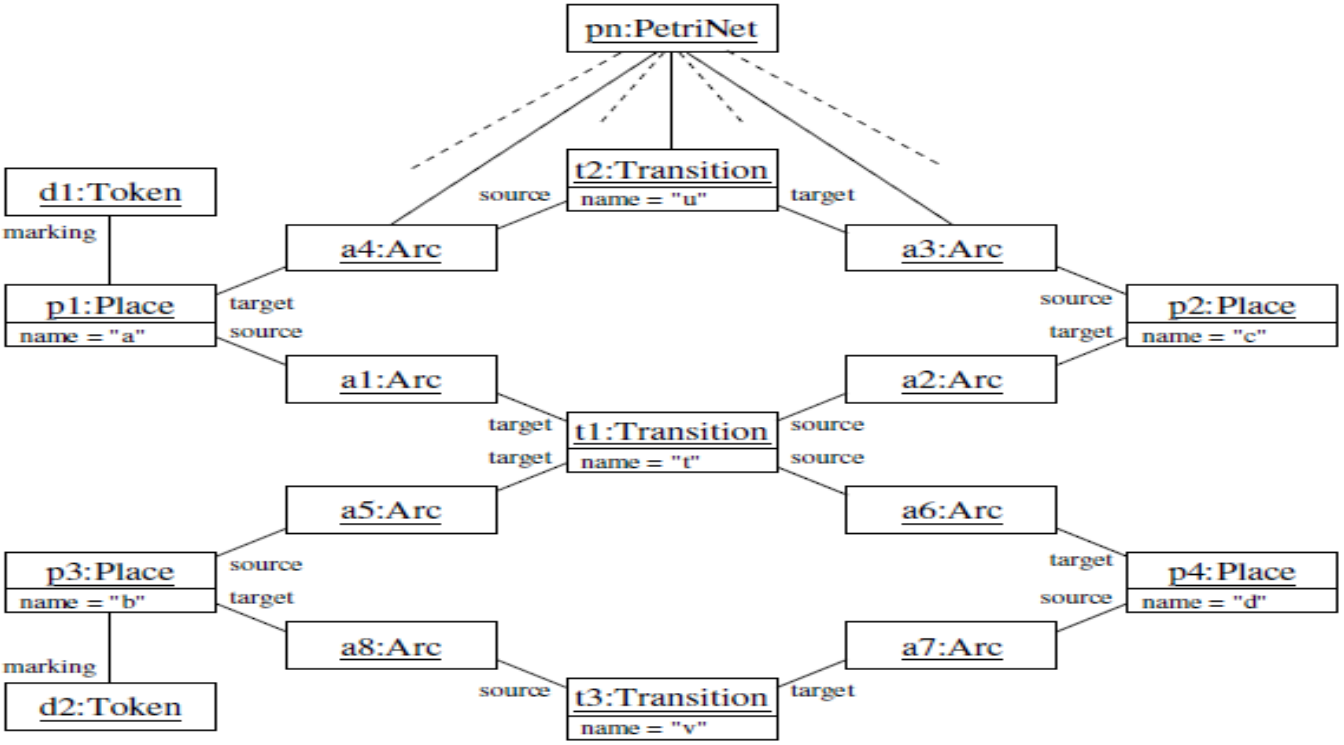
- **Exemple: Un modèle de réseau de Petri**
  Syntaxe Concrète



- **Exemple: Un meta-modele de réseau de Petri**
  Diagramme de Classe UML + OCL



- **Exemple: Un meta-modèle de réseau de Petri**
  Syntaxe Abstraite : Diagramme Objet UML (Outil TGG)

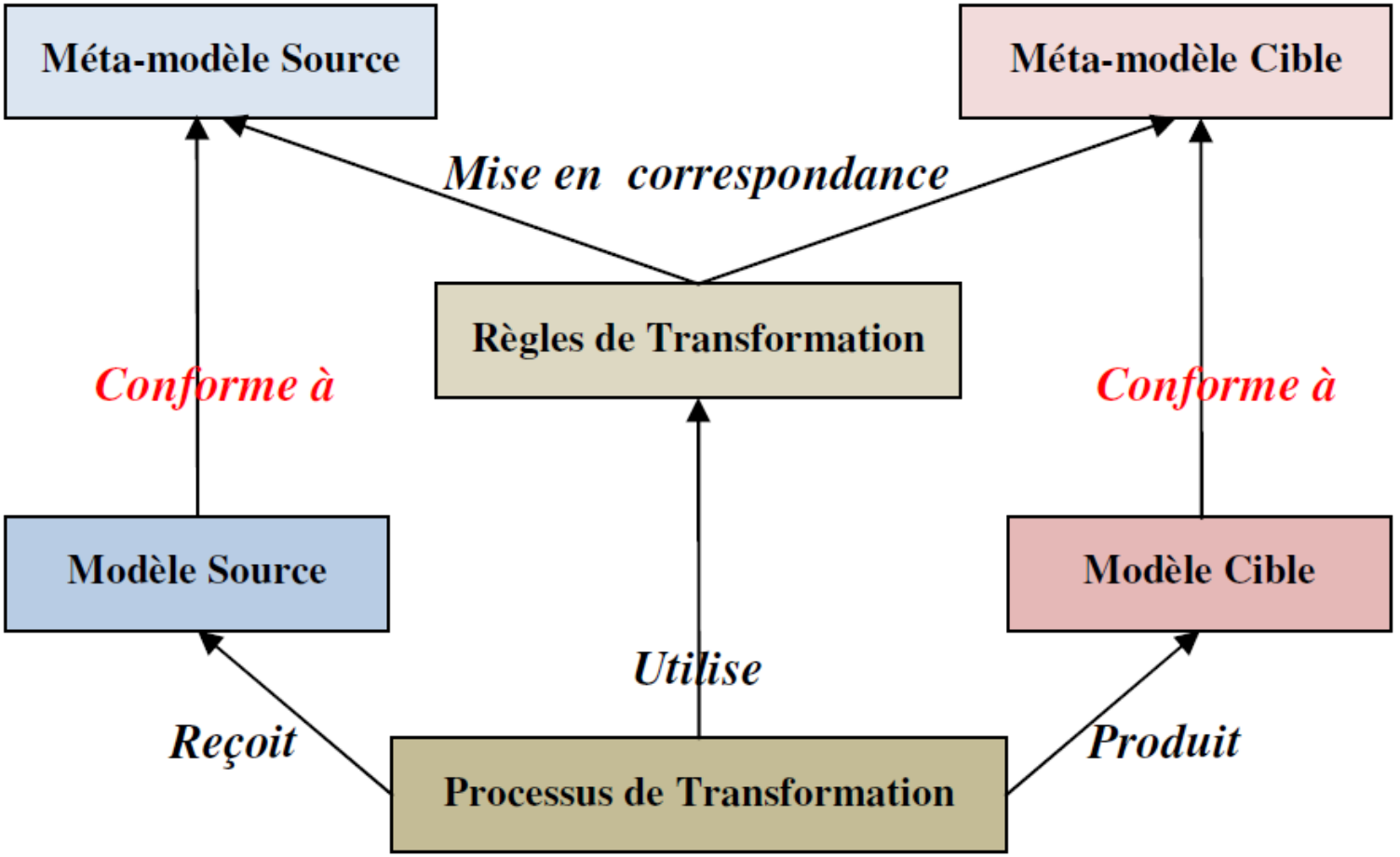


Sommaire

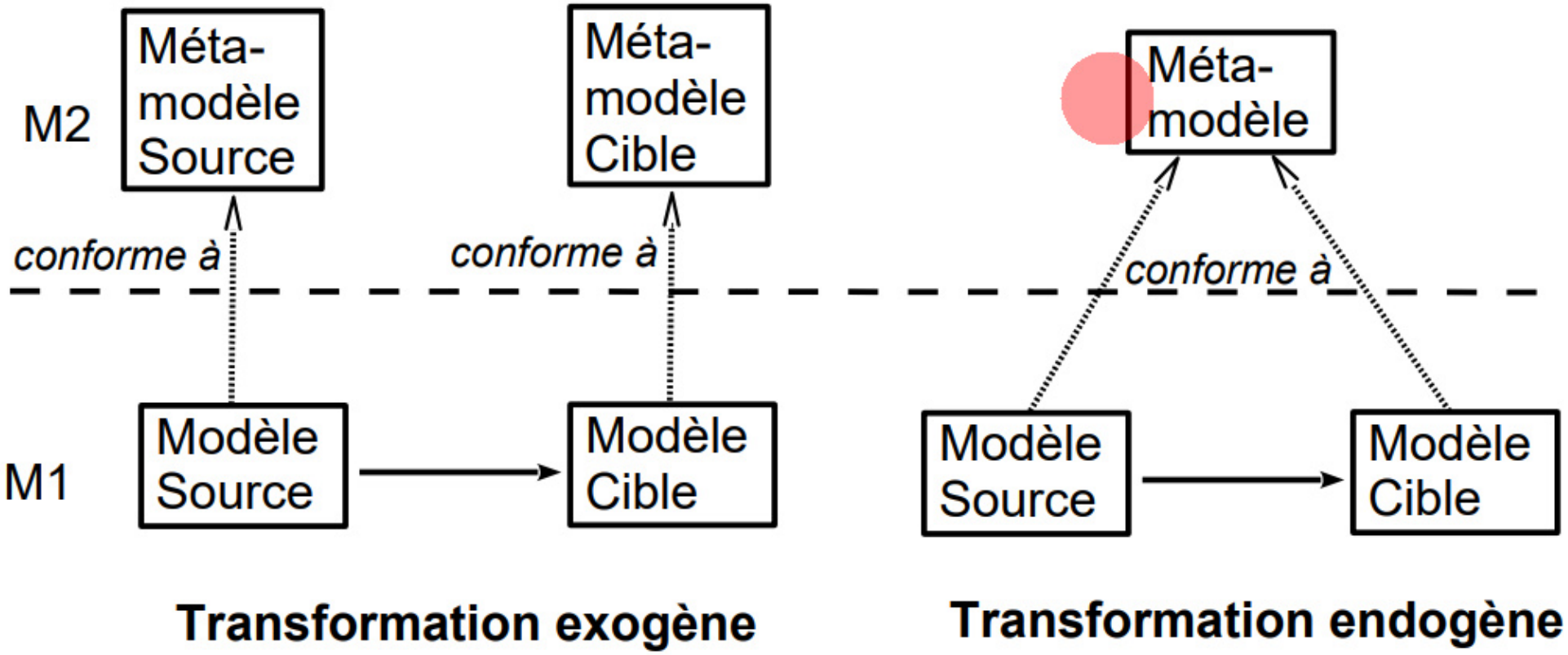
4-Transformation de Modèles

- **Une transformation de modèles consiste à passer d'un modèle source à un modèle cible**
- **Différentes sémantiques**
  - **Optimisation,**
  - **Génération de code,**
  - **...**
- **Deux grandes classes de transformation de modèles:**
  - **Transformations de type: *Modèle vers Code***
  - **Transformations de type: *Modèle vers Modèle***
- **Le passage de l'un à l'autre est décrit par des règles de transformation**
- **Ces règles sont exécutées sur les modèles sources afin de produire les modèles cibles**

- **Processus de Transformation de Modèles**



4-Transformation de Modèles



Sommaire

5-ARCHITECTURE MDA : PRINCIPES

- MDA vise à modéliser des applications et des systèmes indépendamment de l'implémentation cible (niveau matériel ou logiciel), ce qui permet la réutilisation des modèles développés.

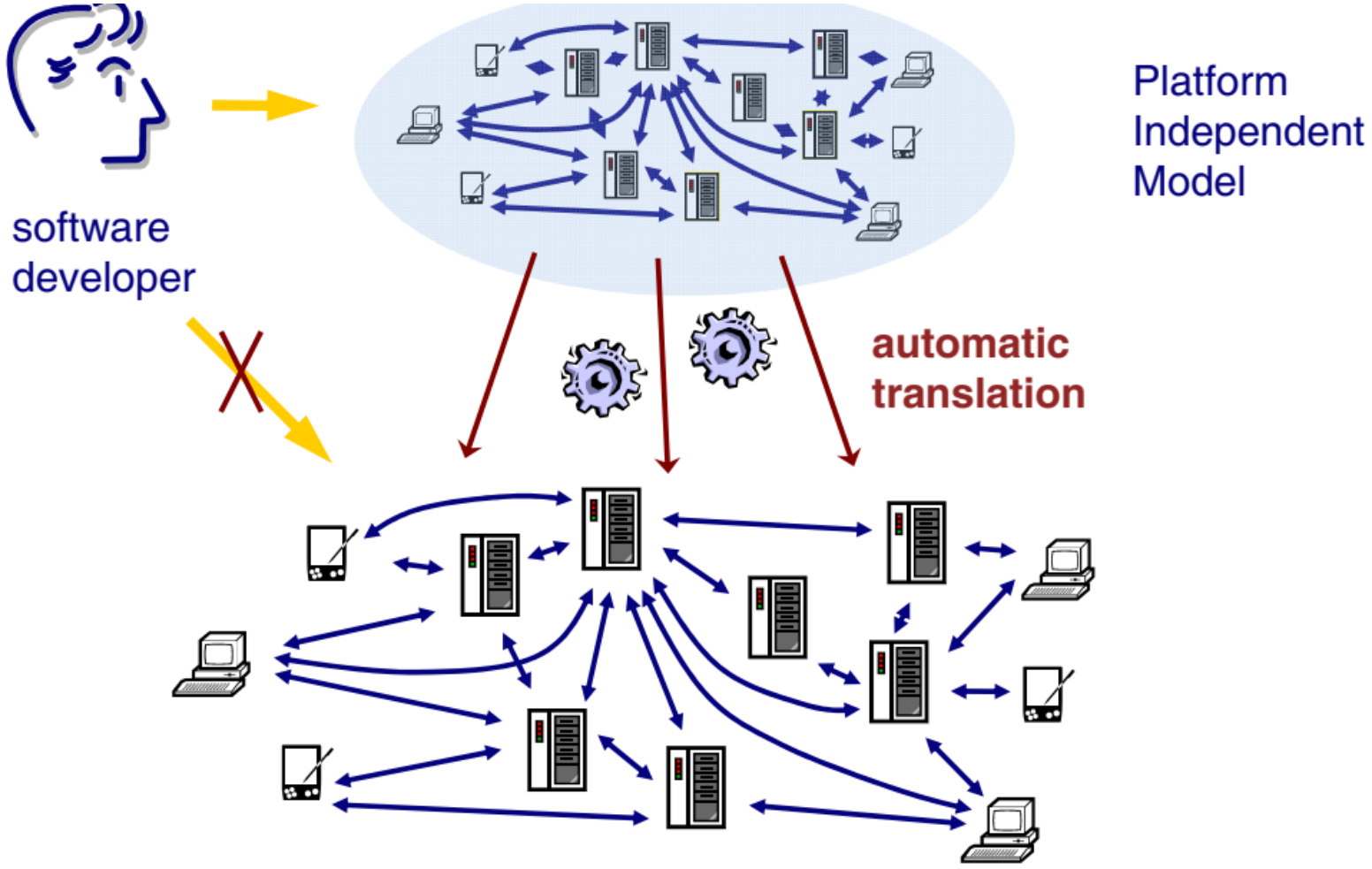
- Les modèles ainsi créés (PIM :Platform Independant Model) seront transformés pour obtenir des modèles d'applications spécifiques à la pateforme cible (PSM : Platform Specific Model).

- Des outils de génération automatique du code, permettent par la suite de générer les programmes directement à partir des modèles.

➢ Cette approche, permet en plus de faire évoluer aisément les applications et leurs architectures à partir des modèles.

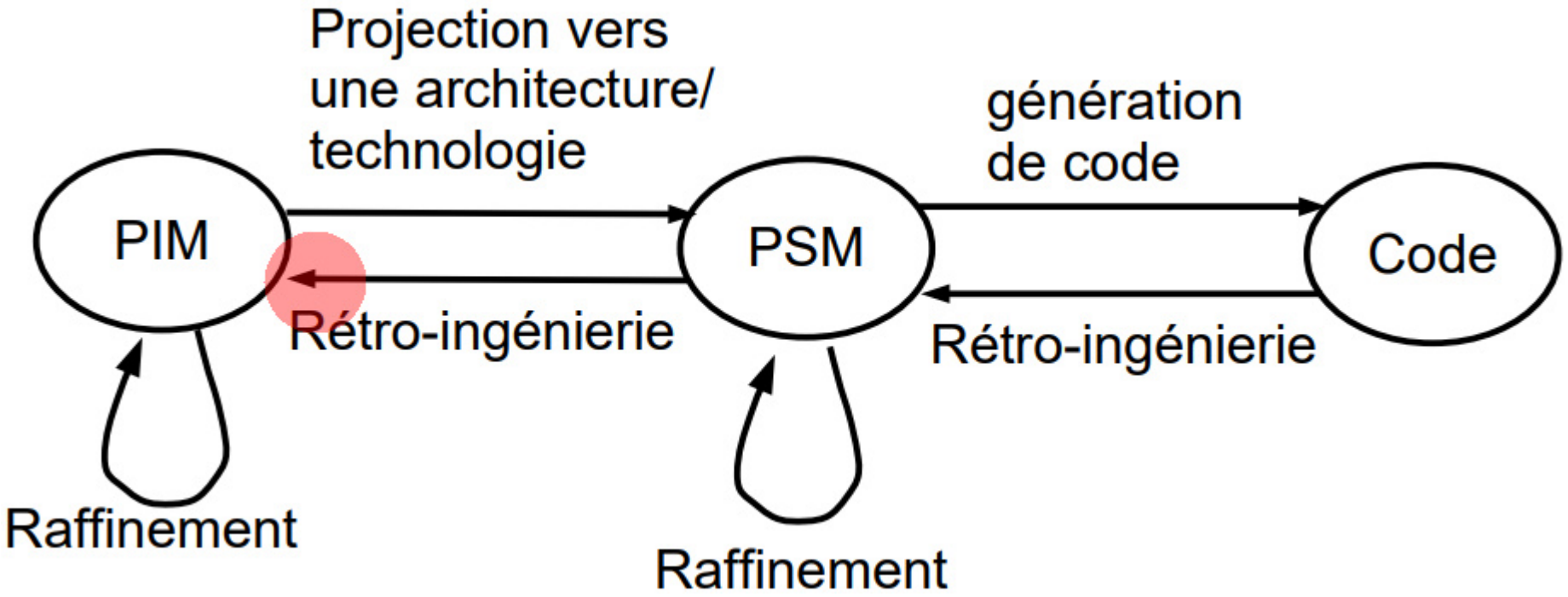
❖ Par conséquent, Le MDA s'investit dans un grand atelier dont le défit relève techniquement de la manipulation des modèles. Dans ce cadre spécifique, la transformation de modèle basé sur les techniques de **Méta-modélisation** et **l'ingénierie de modèles** ouvre un champ d'investigation prometteur pour la recherche.

5-ARCHITECTURE MDA :*La Vision MDA*

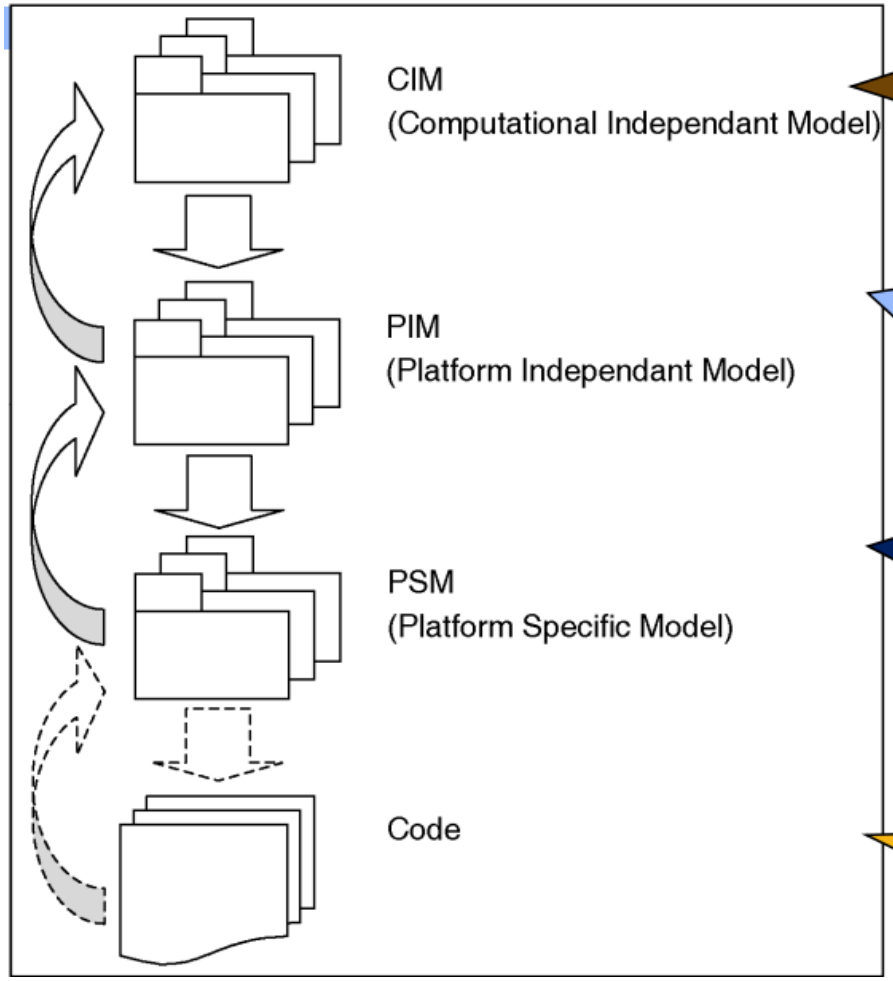


5-ARCHITECTURE MDA

- **Le MDA définit 2 principaux niveaux de modèles**
  - PIM : Platform Independent Model
    - Modèle spécifiant une application indépendamment de la technologie de mise en oeuvre
    - Uniquement spécification de la partie métier d'une application
  - PSM : Platform Specific Model
    - Modèle spécifiant une application après projection sur une plate-forme technologique donnée



5-Architecture MDA



Modèle d'exigences: représente l'application dans son environnement.

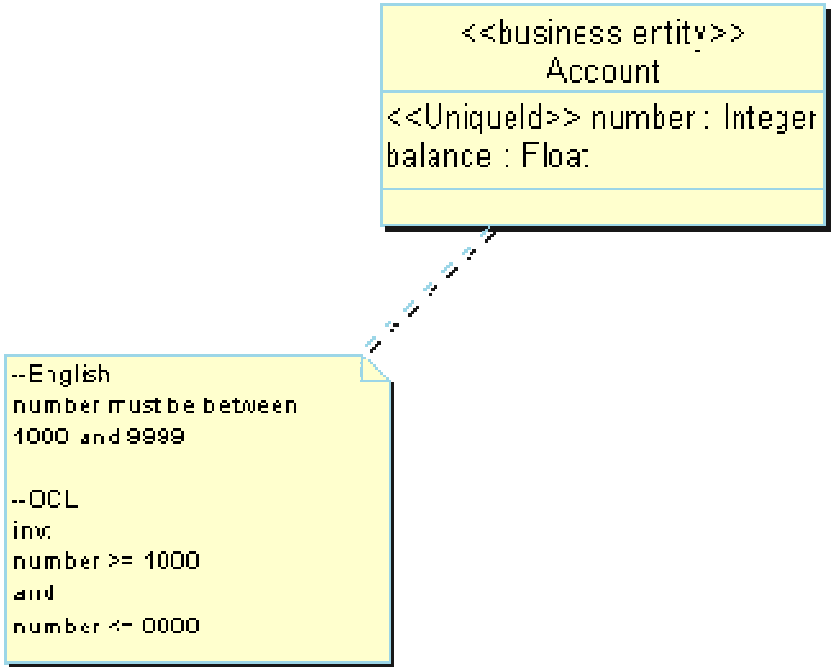
Modèle d'analyse et de conception abstraite: représente l'architecture de l'application.

Modèle de code: représente la construction de l'application.

Code de l'application et fichier de configuration.

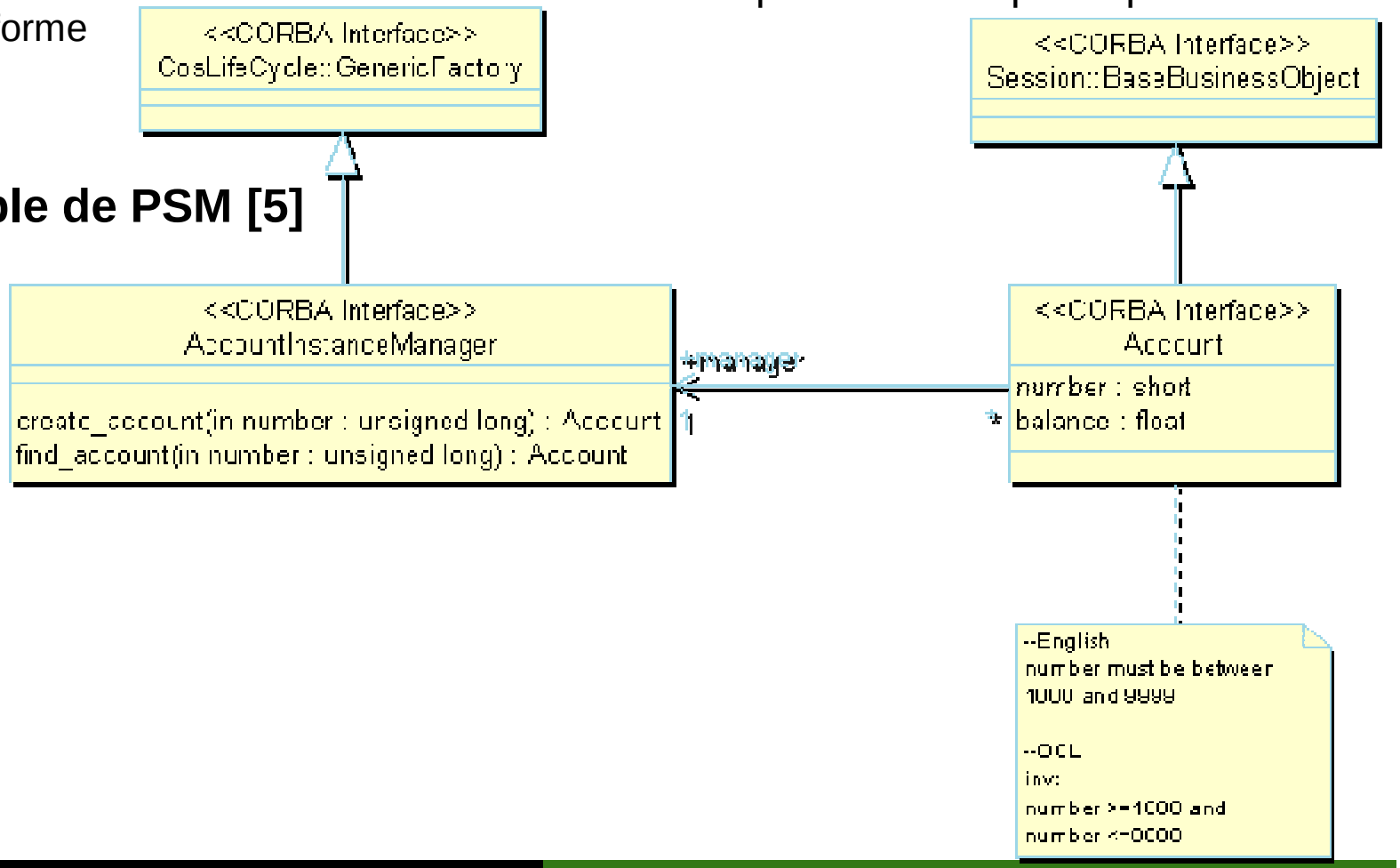
NDM

- **Platform Independant Model (PIM)**
  - Une spécification «formelle» de la structure et de la fonction d'un système qui fait abstraction des détails techniques
  - Exprimé en UML
  - **Exemple de PIM [5]**



- **Platform Specific Model (PSM)**
  - Spécifie comment la fonctionnalité spécifiée dans un PIM est réalisée sur une plate-forme particulière
  - Exprimée en utilisant UML étendu avec des profils UML spécifiques à la plate-forme

- **Exemple de PSM [5]**

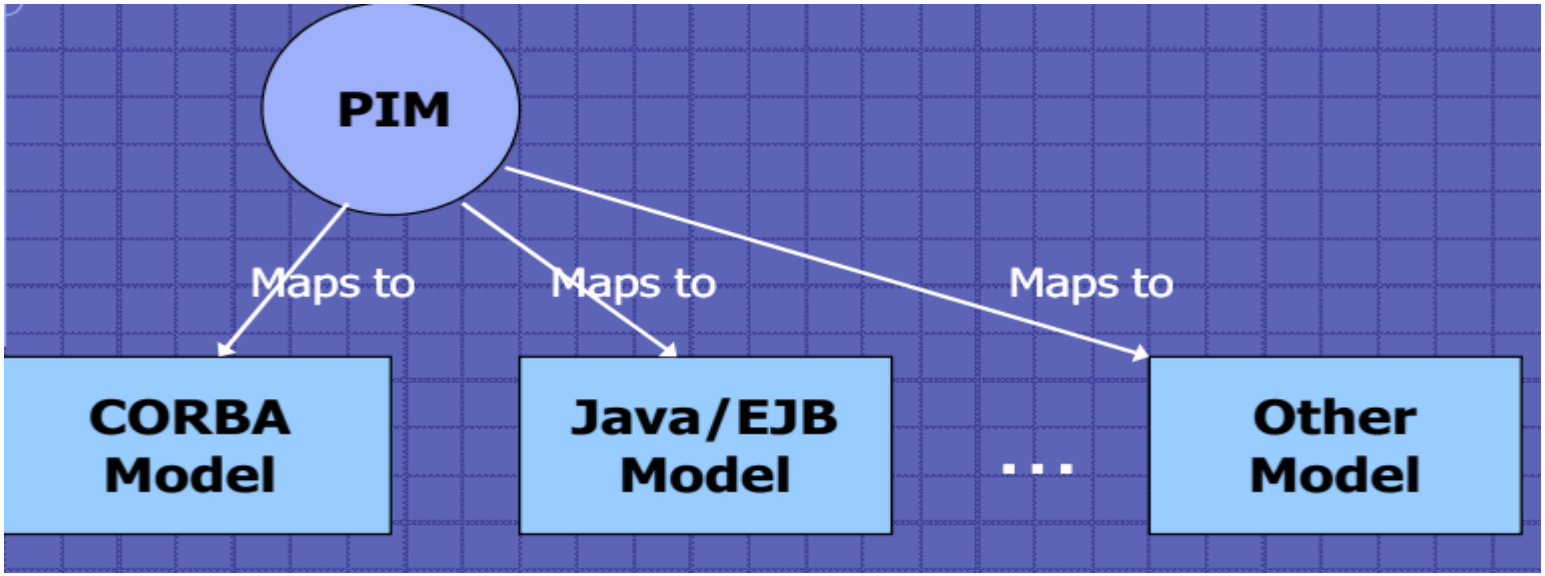


- **Développement en MDA**
  - Tous les projets de développement MDA commencent par la création d'un PIM
  - Le PIM à ce niveau représente la fonctionnalité et le comportement de l'entreprise, non faussés par les détails technologiques
  - Les outils de modélisation d'application MDA contiennent des représentations des services omniprésents et des installations de domaine leur permettant d'être utilisés et / ou incorporés dans l'application via une sélection de menu

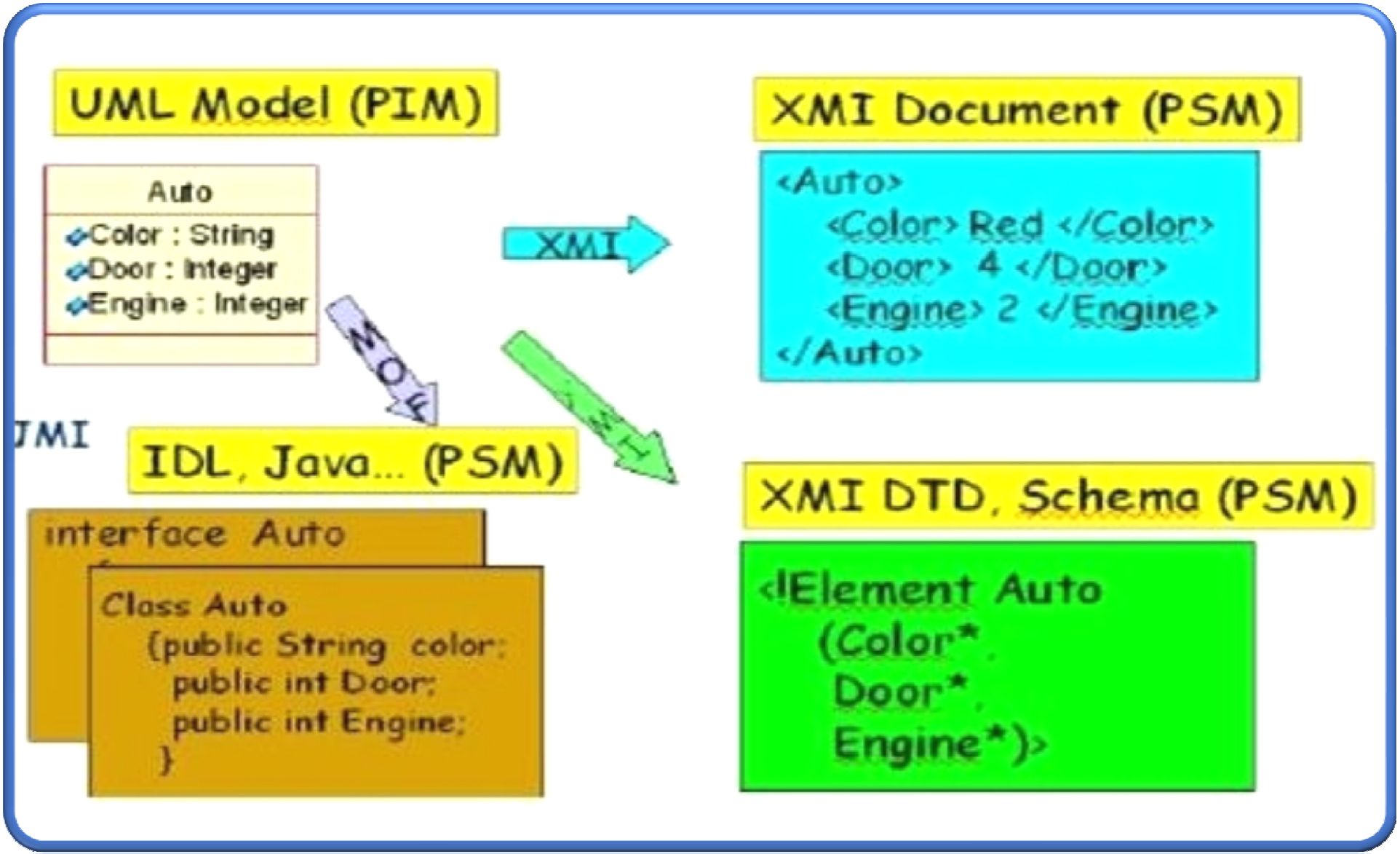
5-Architecture MDA

Exemple :

1. **Modèle de l'application au niveau abstrait, avec un modèle de composant abstrait : modèle PIM**
2. **Projection du modèle vers un modèle de composant EJB : modèle PSM**
3. **Raffinement de ce modèle pour ajouter des détails d'implémentation : modèle PSM**
4. **Génération du code de l'application modélisée vers la plateforme EJB**



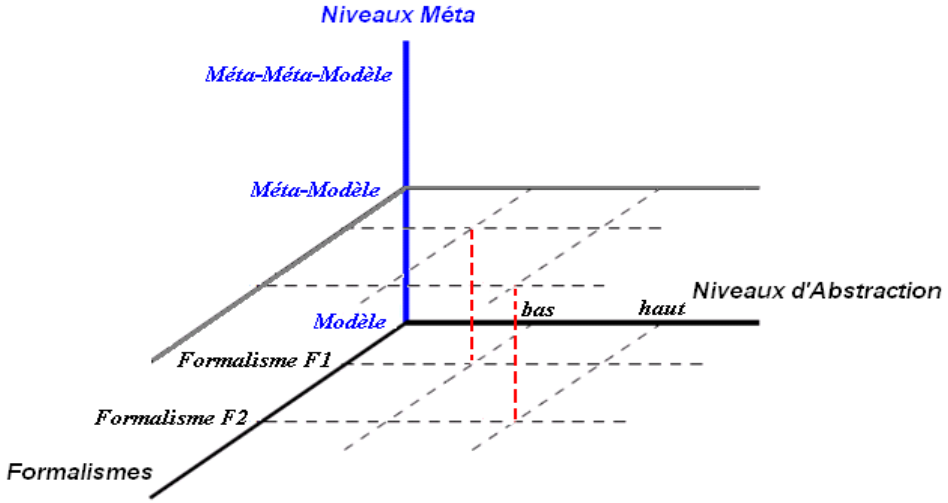
5-ARCHITECTURE MDA :*LES MODÈLES*



Sommaire

Modélisation Multi-Paradigme

Trois directions de recherche orthogonales



- ***Modélisation Multi-Abstraction:*** La relation entre des modèles à des niveaux d'abstraction différents
- ***Modélisation Multi-Formalisme:*** Le couplage et la transformation entre des modèles décrits dans plusieurs formalismes
- ***Méta-Modélisation:*** La description des formalismes et des langages de modélisation

6-MMP, les 3 directions: Modélisation Multi-Abstraction

- Un système peut être décrit par plusieurs modèles, et dans plusieurs niveaux d'abstraction ou de détail
- Chaque niveau d'abstraction est mieux adapté à une tâche particulière
- Le processus de changement d'abstraction est un type de transformation de modèle
- La dérivation automatique de modèles à différent niveaux d'abstraction augmente la qualité des modèles pour mieux maitriser et comprendre tous les détails du système

6-MMP, les 3 directions: Modélisation Multi-Formalisme

- La complexité croissante des systèmes exige l'introduction de plusieurs formalismes
- Chaque composant/aspect du système est modélisé en utilisant le formalisme et l'outil d'analyse approprié
- Le comportement global du système est évalué en transformant les modèles des composants/aspects vers un formalisme unique
- L'automatisation de ces transformations par des supports outillés augmente la productivité de la modélisation
  - L'ajout de nouveaux formalismes sans fournir beaucoup d'éfforts
  - L'utilisation de nouveaux environnements de simulation et d'analyse

6-MMP, les 3 directions: Méta-Modélisation

- Besoin de multiples éditeurs
- La réalisation de ces éditeurs est relativement complexe et coûteuse

Solution: principe de la Méta-modélisation

- Modéliser les formalismes
- Générer automatiquement des éditeurs pour ces formalismes

- La seule information à fournir est le méta-modèle du langage sans se préoccuper des détails d'implémentation de l'éditeur
- Avantages: préservation des acquis et flexibilité
  - Adaptation aux nouveaux besoins de modélisation
  - Ajout de contraintes spécifiques à un domaine

6-MMP, les 3 directions: Mettre en relation

- Transformation de modèles permet de relier ces trois dimensions pour cumuler leurs avantages
  - Combiner et transformer les différents formalismes
  - Utiliser des formalismes et des outils spécifiques au domaine d'application
  - Vérifier la cohérence entre les différentes vues/aspects du système

- Différentes manipulations de modèles
  - La transformation de formalismes
  - L'optimisation de Modèle
  - La simulation
  - La génération de code

- Les modèles et les méta-modèles sont des graphes

  ⇘ les transformations entre modèles peuvent être décrites et réalisées par des ***Transformations de Graphes***

7-Conclusion: IDM

- Approche de développement qui met à la disposition de l'utilisateur des **concepts**, des l**angages** et des **outils**
- Les **modèles** sont considérés comme des éléments de base
- Raisonnement est entièrement à un haut niveau d'abstraction
- L'application sera générée (*en tout ou en partie, automatiquement ou semi-automatiquement*) à partir de modèles
  - ☛ *Ingénierie générative*
- Les outils permettant de créer et d'exploiter ces modèles sont construits autour des concepts de:
  - Méta-modélisation
  - Transformation de modèles

Références

1. Thèse de Doctorat en sciences de Mr Kerkouche Elhillali (Université Mentouri Constantine, 2010)

2. Davide Buscaldi, OMG's Model Driven Architecture

3. Cours de Alain Cariou, Université de Pau

4. Cours de Alloua Chaoui, Université de Constantine 2

5. Livre de Xavier Blanc , MDA en action, Editions Eyrolles

6. Oscar Nierstrasz, OMG's Model Driven Architecture

7. AToMPM home page : https://atompm.github.io/