

Diplôme National d'ingénieur en informatique

Génie logiciel

Niveau: 3^{ème} année

INGÉNIERIE DIRIGÉE PAR LES MODÈLES

Dr. Dhikra KCHAOU
dhikrafsegs@gmail.com

Présentation générale du cours

► Volume Horaire

- ✓ 1h30 par quinzaine COURS
- ✓ 3h par quinzaine TP

► Objectifs

- ✓ Comprendre les fondements des approches MDA et MDE
- ✓ Comprendre les concepts de modèle, méta-modèle, profil UML, OCL, transformation de modèles
- ✓ Pouvoir développer des applications utilisant les environnements de développement de l'approche MDA



Plan du cours

1. Introduction
2. MDE et MDA
3. Modèle, Méta-modèle et Méta-méta-modèle
4. UML, Profil UML et OCL
5. Transformation des modèles





Introduction

Évolution de la technologie

Évolution de la technologie

- ▶ De l'architecture centralisée vers l'architecture distribuée
 - ✓ Architecture Client/serveur
 - ✓ Architecture distribuée (CORBA : Norme proposée par OMG qui permet de faire communiquer un ensemble des applications en environnement hétérogène (plusieurs systèmes et plusieurs langages)).

- ▶ De la programmation procédurale vers la programmation orientée objet
 - ✓ C : procédural
 - ✓ C++, JAVA, C# : Objet (encapsulation, réutilisation, héritage, spécialisation)
 - ✓ EJB, CCM: Composant (meilleure encapsulation et réutilisation, déploiement,...)



Évolution de la technologie

► Difficulté de la standardisation et de l'universalité

❖ Sun/Oracle : Java

Plate-forme d'exécution universelle : intergiciel
(middleware) intégré (RMI)

❖ OMG : CORBA

Indépendant du langage de programmation et des
systèmes d'exploitation : Intergiciel universel !!!

❖ Microsoft et d'autres : Web Services

Interopérabilité universelle entre composants:
Intergiciel = HTTP/XML



Évolution de la technologie

- ▶ Évolution sans fin !!!

Quelles conséquences en pratique de cette évolution permanente ?

- ▶ Nécessité d'une idée afin de limiter le nombre de technologies : **Normaliser** un standard qui sera utilisé par tous.
- ▶ **Nécessité d'adapter une application à ces technologies**
- ▶ Quel est le coût de cette adaptation/Normalisation ?
Généralement très élevé :
 - ✓ Réécrire presque entièrement l'application
 - ✓ Mélange du code métier et du code technique
 - ✓ Aucune capitalisation de la logique et des règles métiers

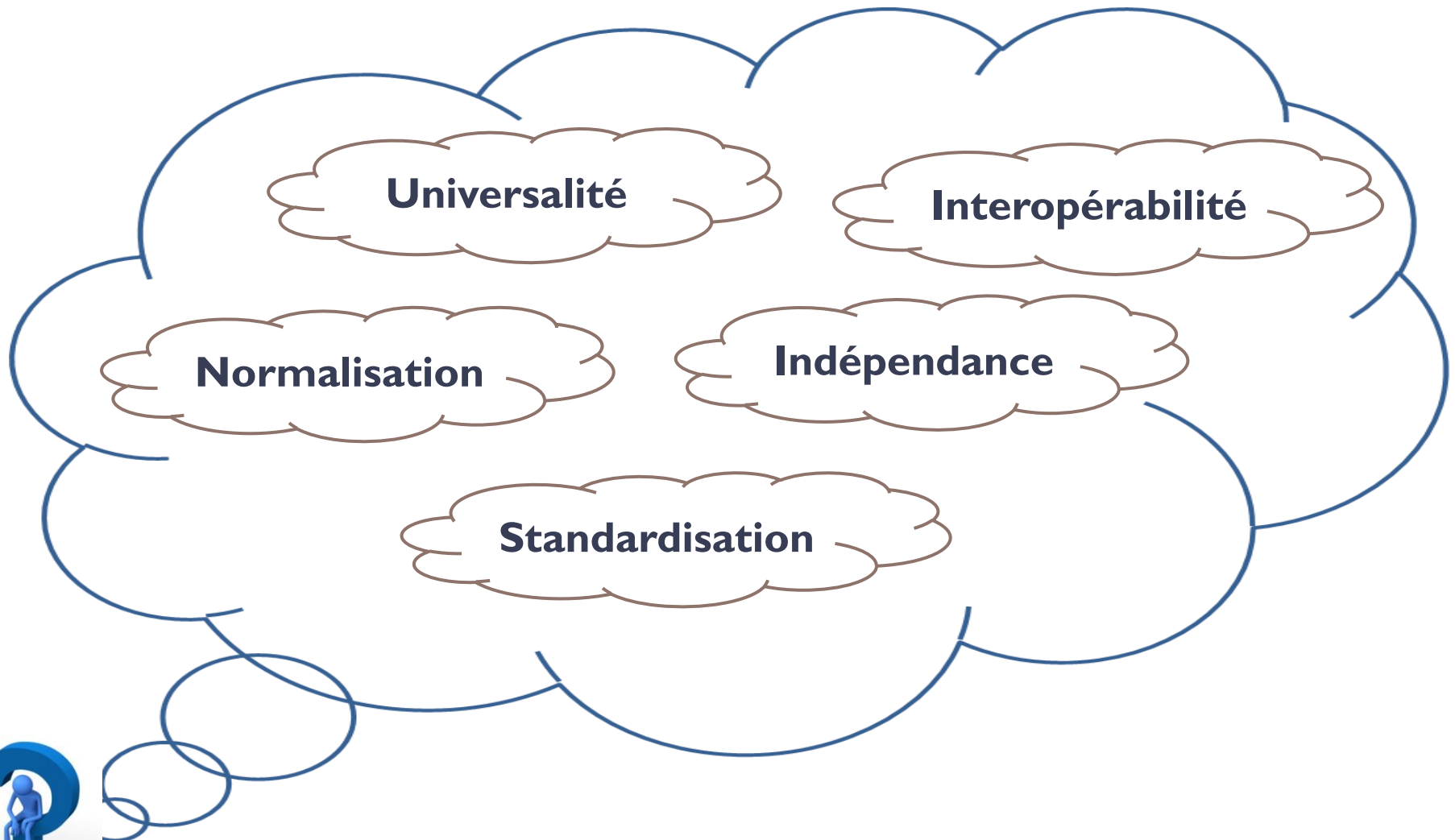
Évolution de la technologie

Exemple

- ▶ Application de calculs scientifiques distribués sur un réseau de machines
- ▶ Passage de C/RPC à Java/EJB
 - ❖ Impossibilité de reprendre le code existant
 - ❖ Paradigme procédural à objet/composant
- ▶ Pourtant
 - ✓ Les algorithmes de distribution des calculs et de répartition des charges sur les machines sont indépendants de la technologie de mise en œuvre
 - ✓ Logique métier indépendante de la technologie



Évolution de la technologie



Évolution de la technologie

Ce qu'il faut faire:

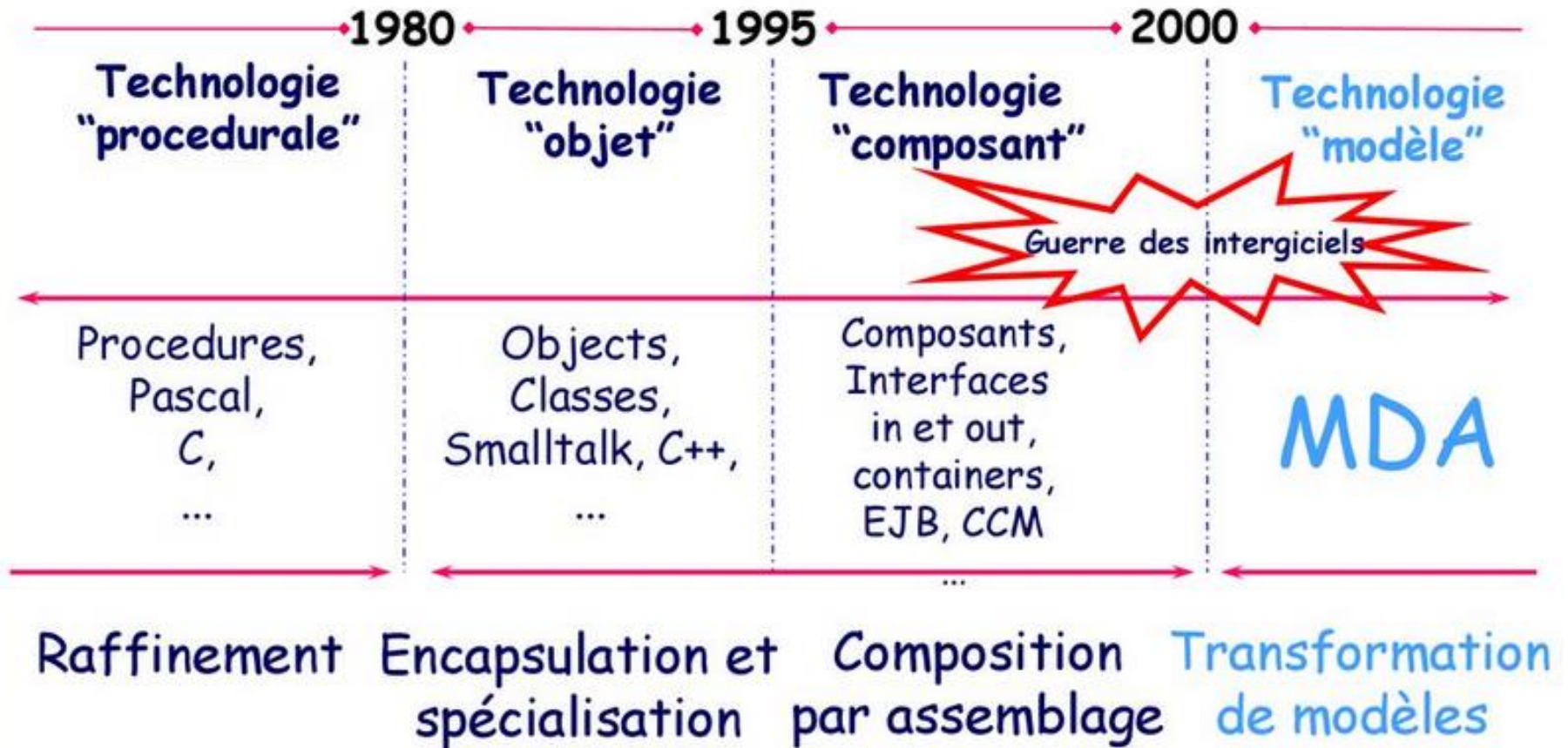
- ▶ Découpler clairement la logique métier et de la mise en œuvre technologique

C'est l'un des principes fondamentaux de l'ingénierie des modèles

- ▶ Séparation des préoccupations (eng. separation of concerns)
- ▶ Besoin de modéliser/spécifier
 - ❖ La partie métier (à un niveau abstrait)
 - ❖ La plate-forme de mise en œuvre
 - ❖ De projeter ce niveau abstrait sur une plateforme



Évolution de la technologie



Les objectifs de L'IDM

- ▶ Couvrir tout le cycle de développement du logiciel
- ▶ Manipuler des modèles
- ▶ Automatiser les transformations entre différents modèles

L'élément de base n'est plus l'objet
C'est le modèle

- ▶ Capitalisation de la logique métier, réutilisation des (ou d'une partie des) modèles.
- ▶ Abstraction des technologies de mise en œuvre : Évoluer bien plus facilement vers de nouvelles technologies.



Les objectifs de l'IDM

- ▶ Séparation des préoccupations
 - Deux principales préoccupations
 - Métier : le cœur de l'application, sa logique
 - Plate-forme de mise en œuvre
 - Plusieurs autres préoccupations possibles
 - Sécurité
 - Interface utilisateur
 - Qualité de service, etc.
- ▶ Chaque préoccupation est modélisée par un modèle




Principe de base de l'IDM

Élaboration de différents modèles partant d'un modèle métier indépendant d'une plateforme cible puis la transformation de ce dernier en modèle spécifique à la plateforme cible.



Les architectures basées sur l'IDM

Microsoft	Software factories	Domain Specific Language
IBM	Eclipse Modeling Framework (EMF)	Ecore (Essentiel MOF)
OMG (Object Management Group)	Model Driven Architecture (MDA)	Meta Object Facility (MOF)





MDA

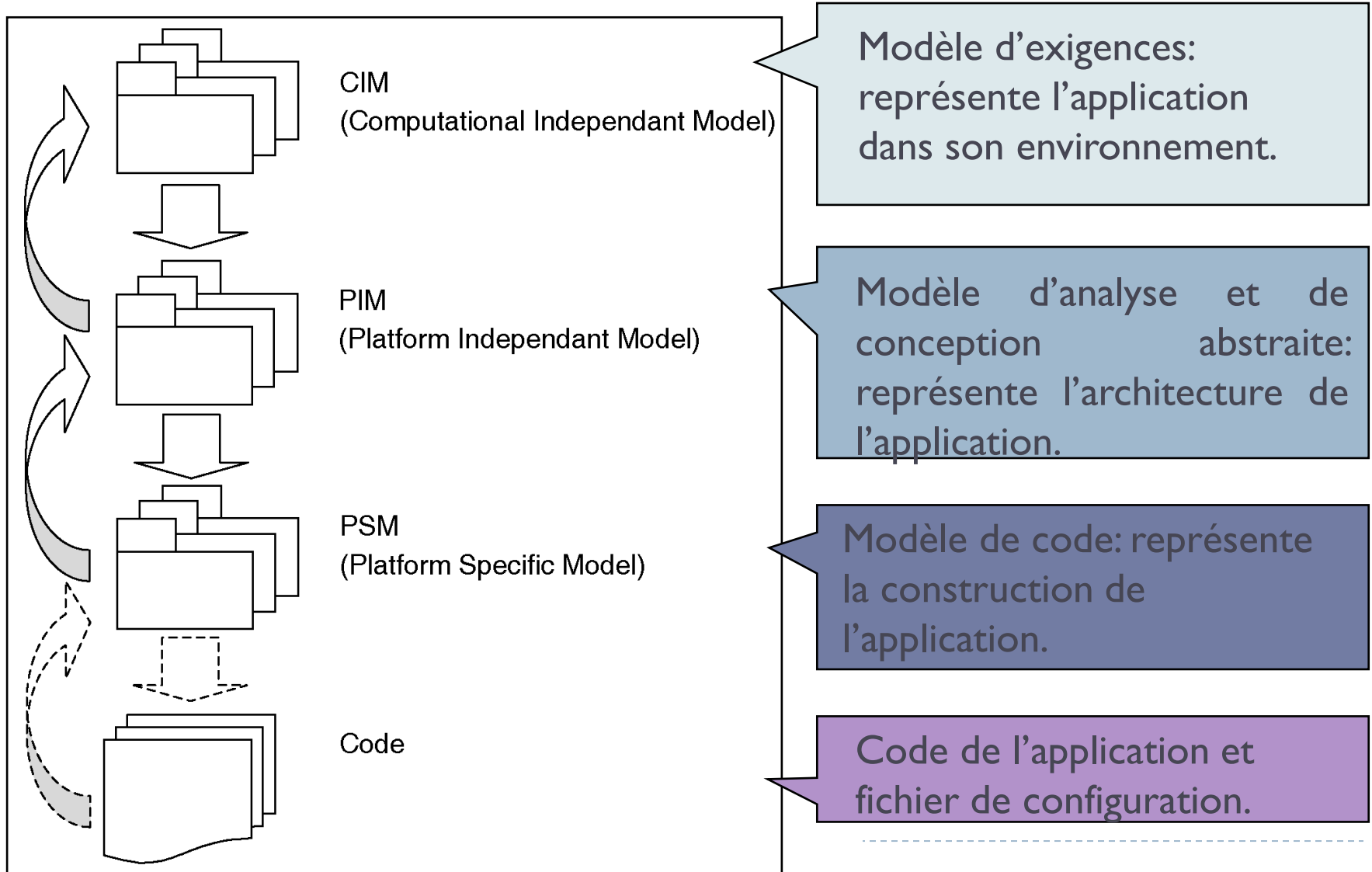
Model Driven Architecture

Model Driven Architecture

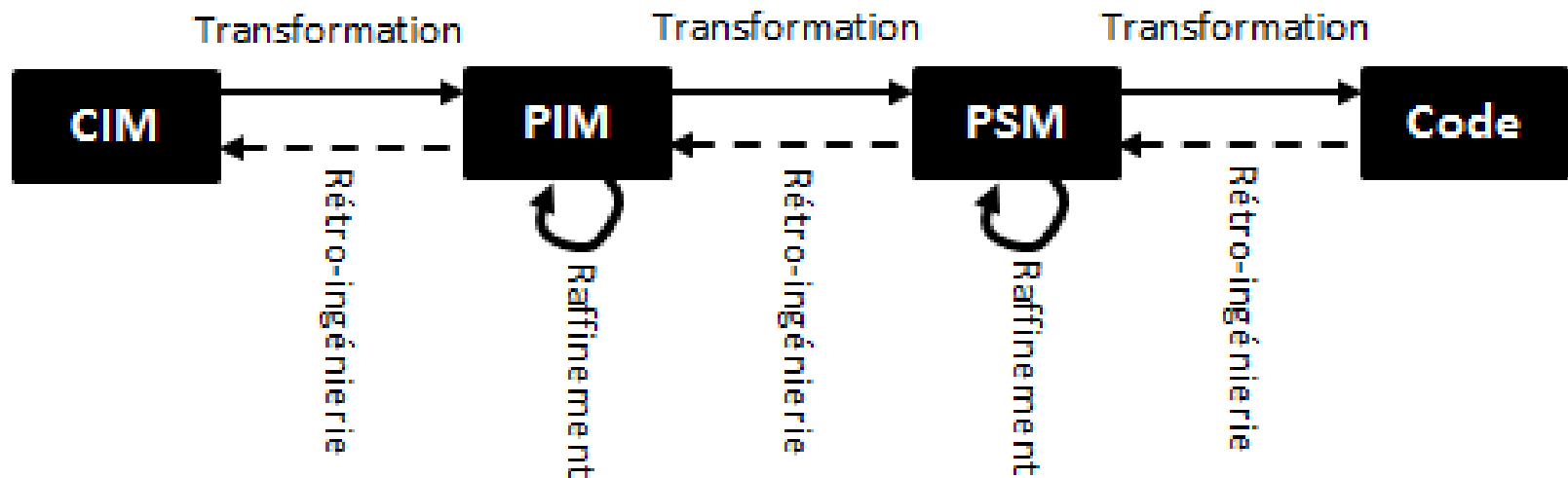
- ▶ Approche de l'OMG
 - ▶ Origine de l'ingénierie des modèles
 - ▶ Date: Fin 2000
- ▶ Le MDA est né à partir de l'évolution continue des technologies
 - ▶ Abstraire les parties métiers de leur mise en œuvre
 - ▶ Basé sur des technologies et standards de l'OMG
 - ▶ Modélisation (UML, MOF, OCL, ...)
 - ▶ Transformation des modèles (QVT, ...)



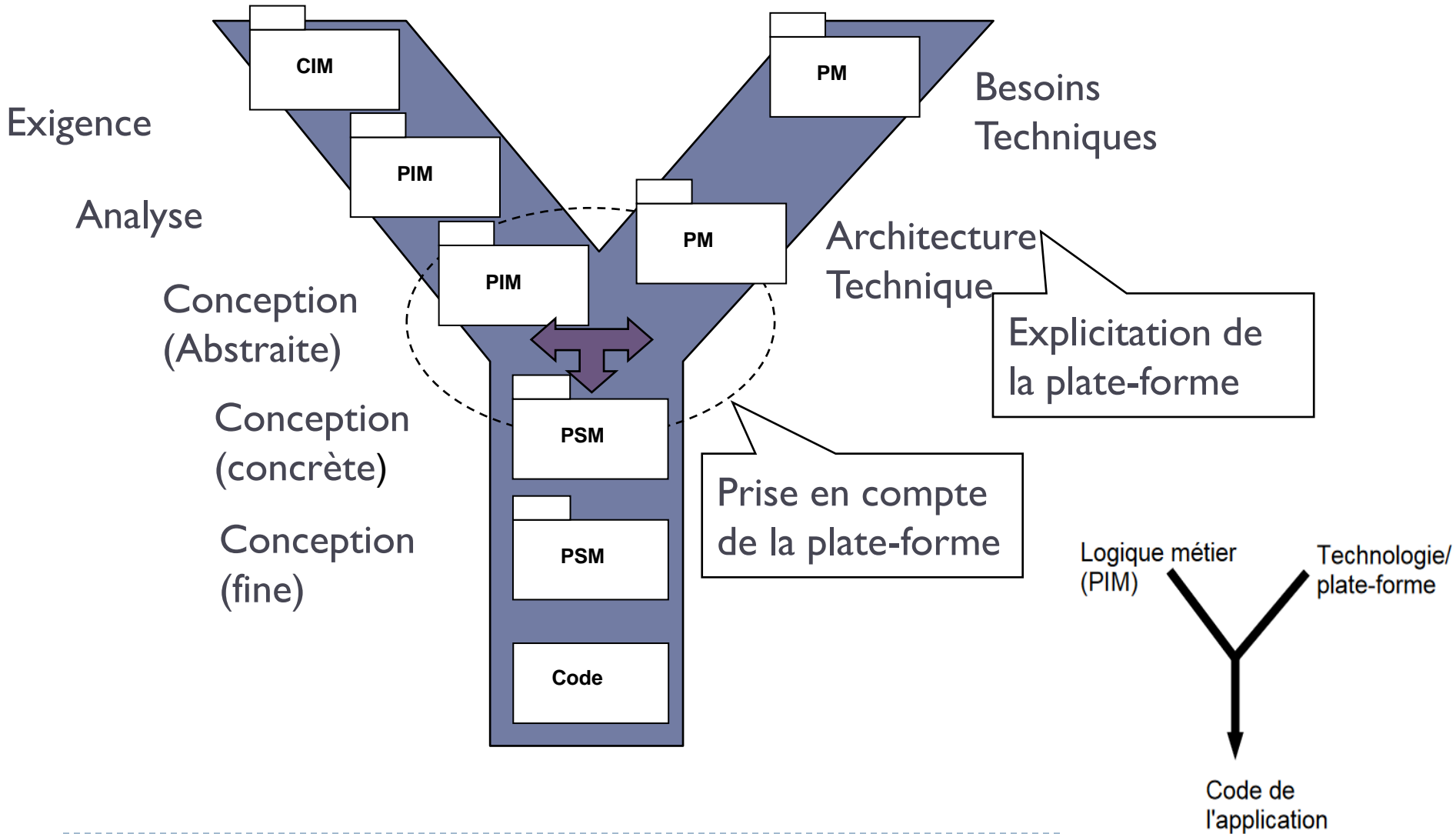
Les quatre niveaux dans MDA



Relation entre les niveaux de MDA

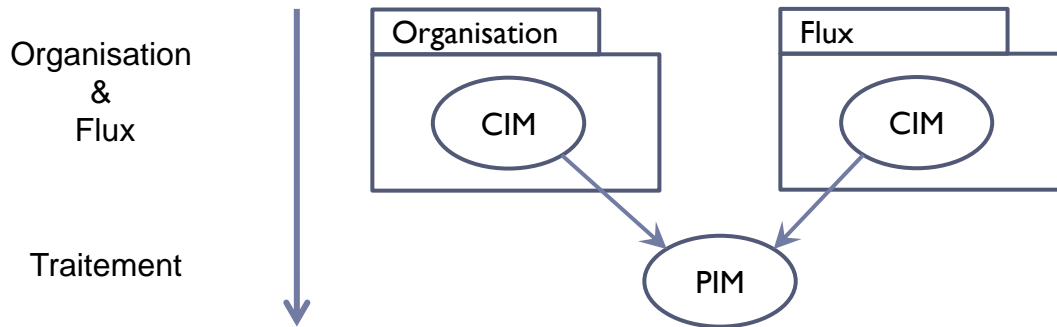


Cycle de développement d'un logiciel selon MDA: cycle en Y

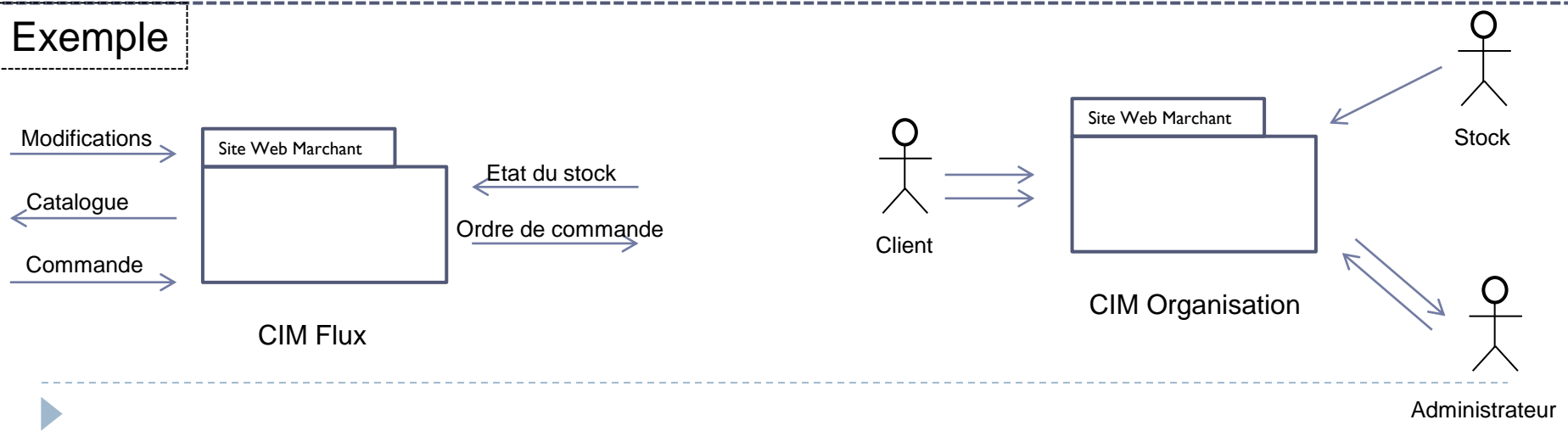


CIM: Computation Independant Model

- Décrire l'organisation ainsi que les flux (les actions) du système)

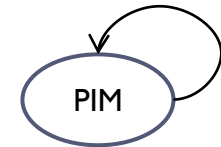


Exemple

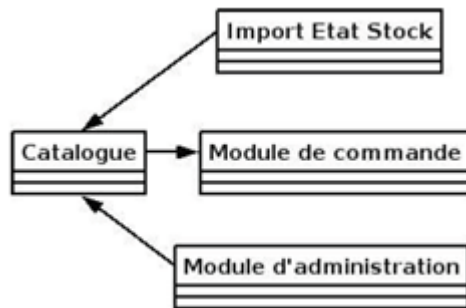


PIM: Plateform Independant Model

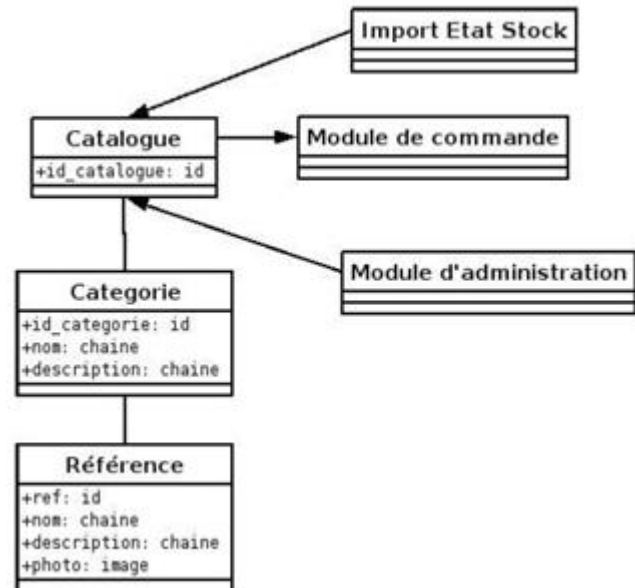
- ▶ Décrit les traitements
- ▶ Raffinements successifs du modèle
- ▶ Indépendant de toute plate-forme



Exemple



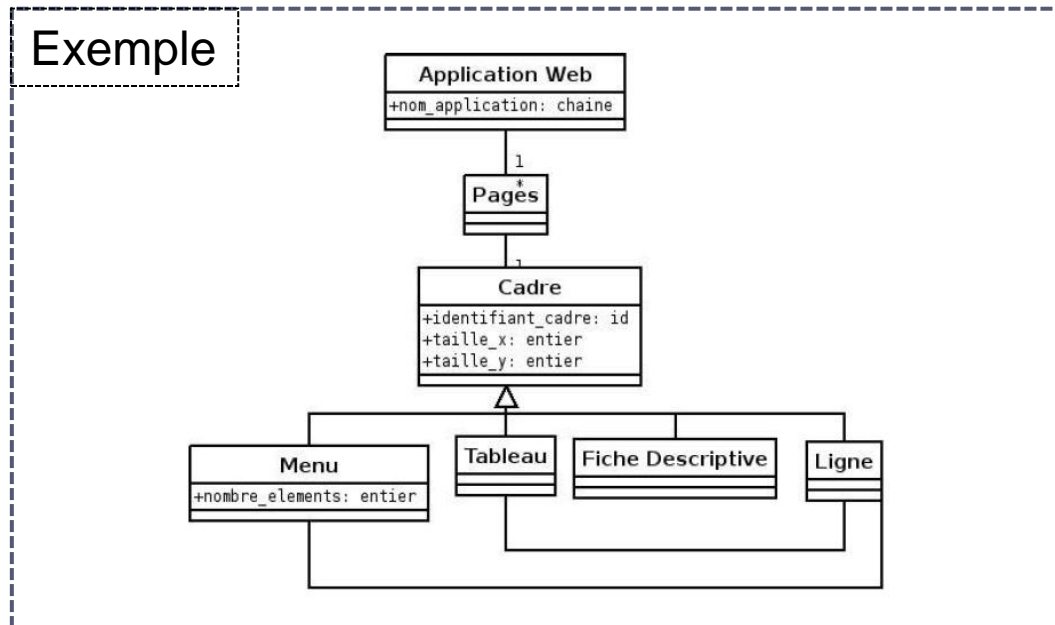
PIM 1



PIM 2

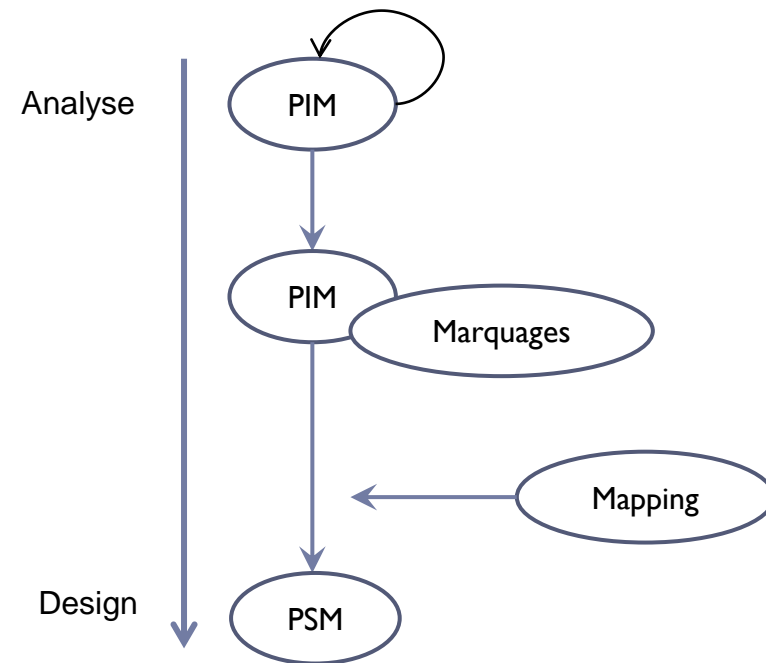
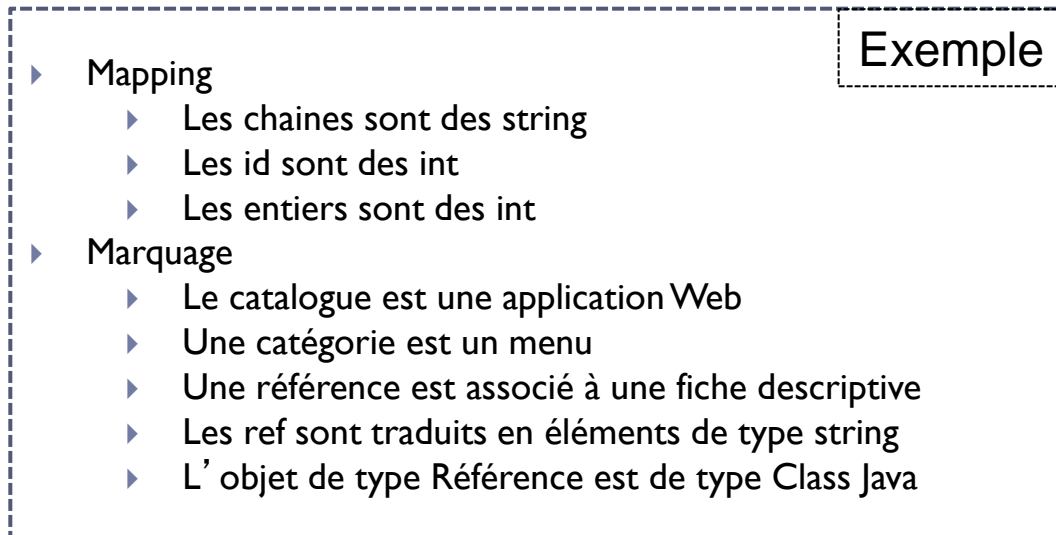
PM: Platform Model

- ▶ Décrit l'architecture technique
- ▶ Divers niveaux de raffinement
- ▶ Plusieurs PM pour un projet



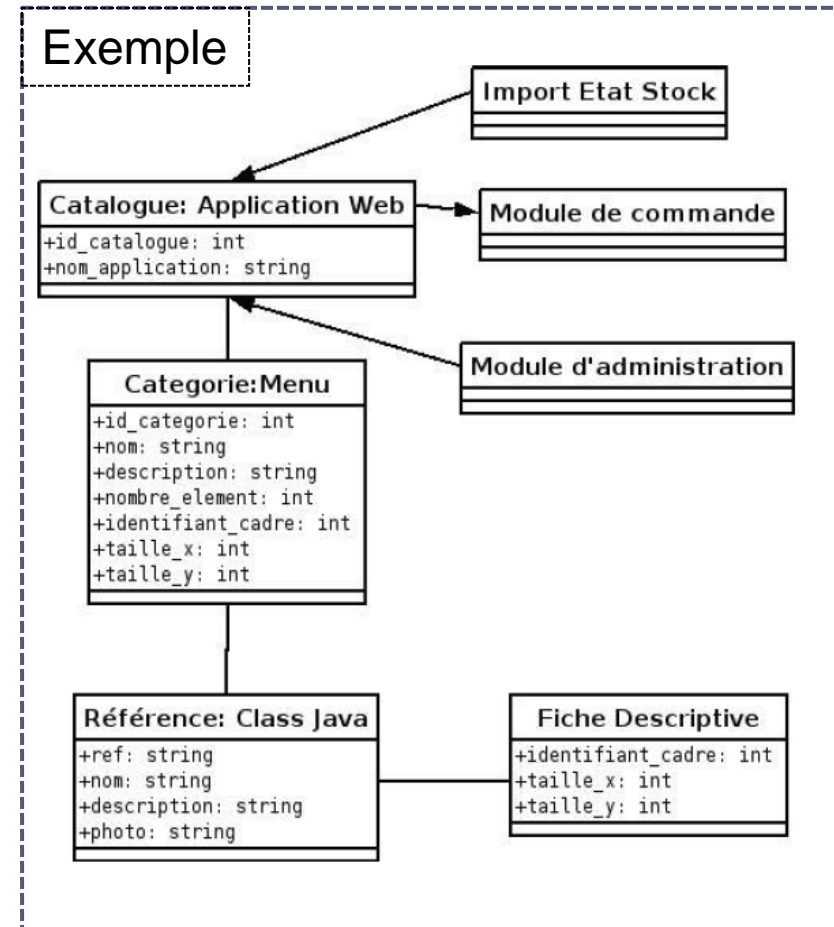
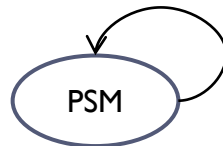
PIM and PM to PSM

- ▶ Le PM décrit l'architecture, le PIM décrit le système, il reste à faire le lien
- ▶ Deux types de transformations:
 - ▶ Mapping : Règles de conversion
 - ▶ Marquage: Cas particulier



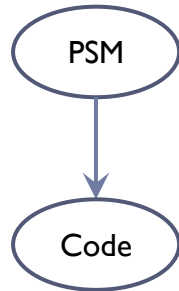
PSM: Platform Specific model

- ▶ Vue spécifique à une architecture du système obtenue après transformation du PIM grâce aux mappings et aux marques
- ▶ Plusieurs itérations sur le PSM sont possibles pour rajouter des détails



Code

- ▶ PSM + OCL = code source généré automatiquement

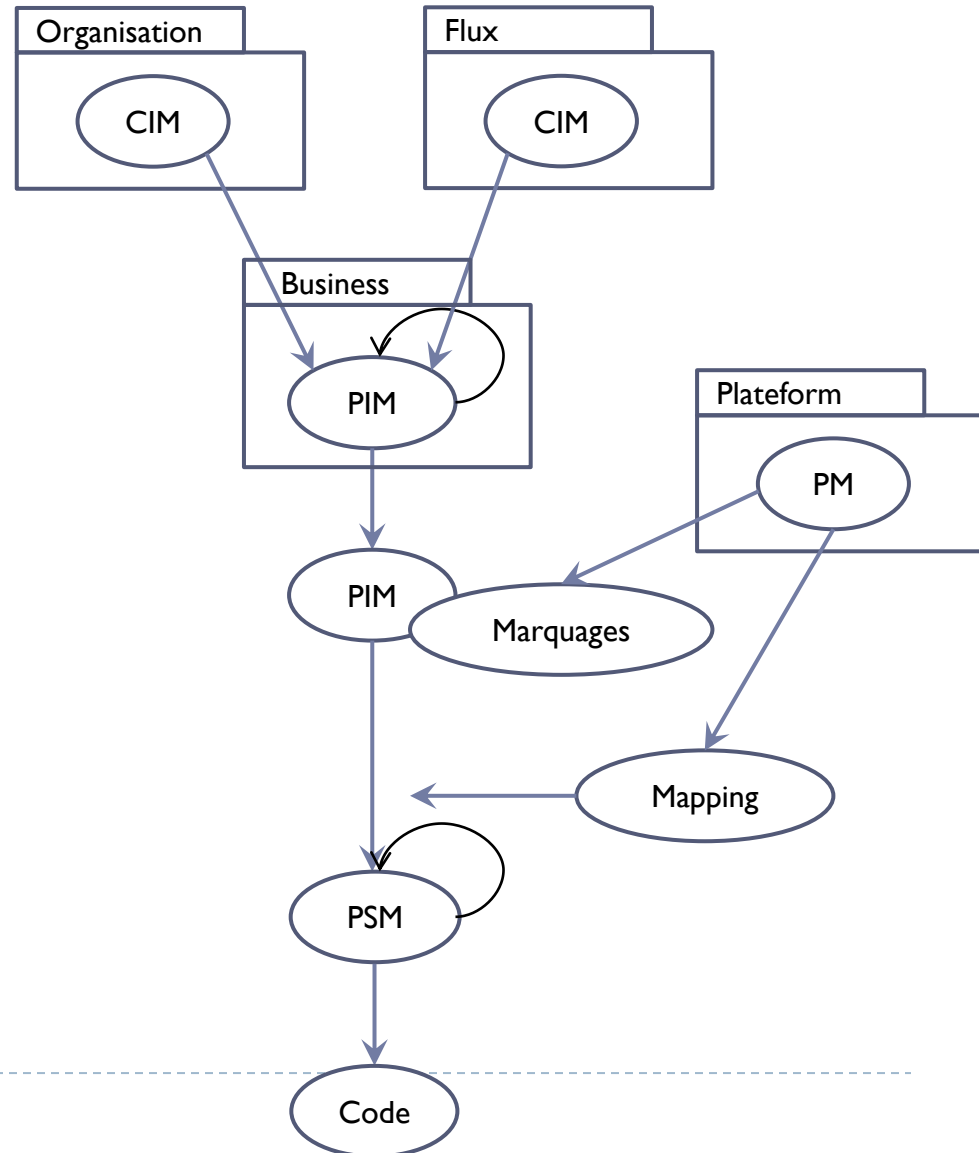


- ▶ La génération de code source suppose de combiner :
 - ✓ Le standard UML, et l'outil de modélisation qui l'implante (ex: Rose, Enterprise Architect, Together, etc.)
 - ✓ Des templates de génération UML → code source,
 - ✓ L'outil de génération de code
 - ✓ Le tout intégré dans une "chaîne" de production



Vue globale de la démarche

- ▶ Nombre important de modèles
- ▶ Divers niveaux de raffinement pour chaque modèle
- ▶ Passage automatique et manuelle entre les modèles



Outils de mise en oeuvre du MDA

**Comment représenter les modèles
(CIM PIM PSM) ?**

Comment représenter les transformations ?



Comment représenter les modèles?

Utilisation des standards de l'OMG

- ▶ **Spécification des méta-modèles : Meta Object Facilities (MOF)**
- ▶ **Spécification des modèles aux différents niveaux**
 - ✓ Langage de modélisation UML et Profils UML
 - ✓ Langage de contraintes OCL
 - ✓ Langage dédiés à des domaines particuliers (CWM ...)
- ▶ **Importation et exportation des modèles**
 - ▶ XML Metadata Interchange (XMI)

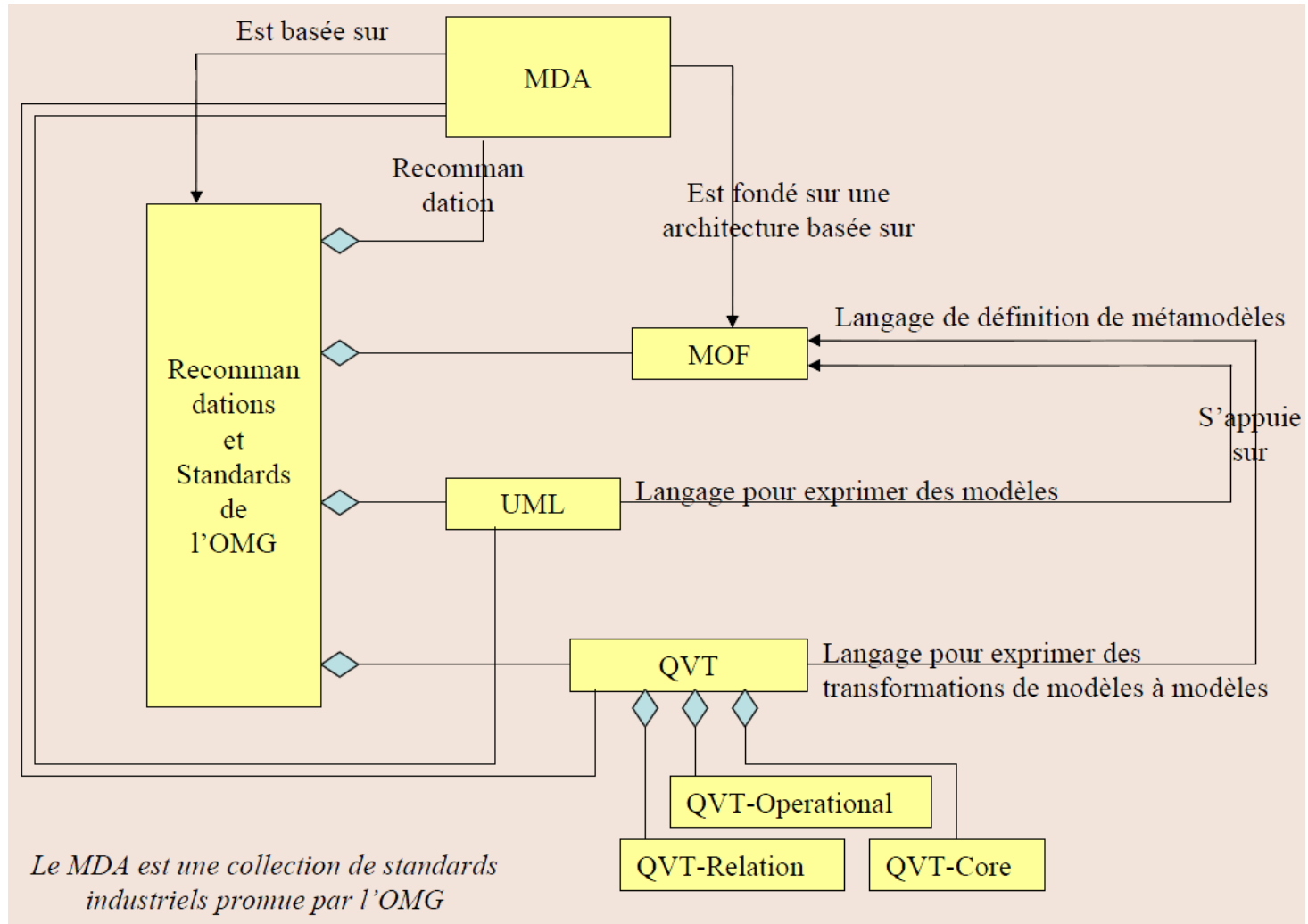


Comment représenter les transformations?

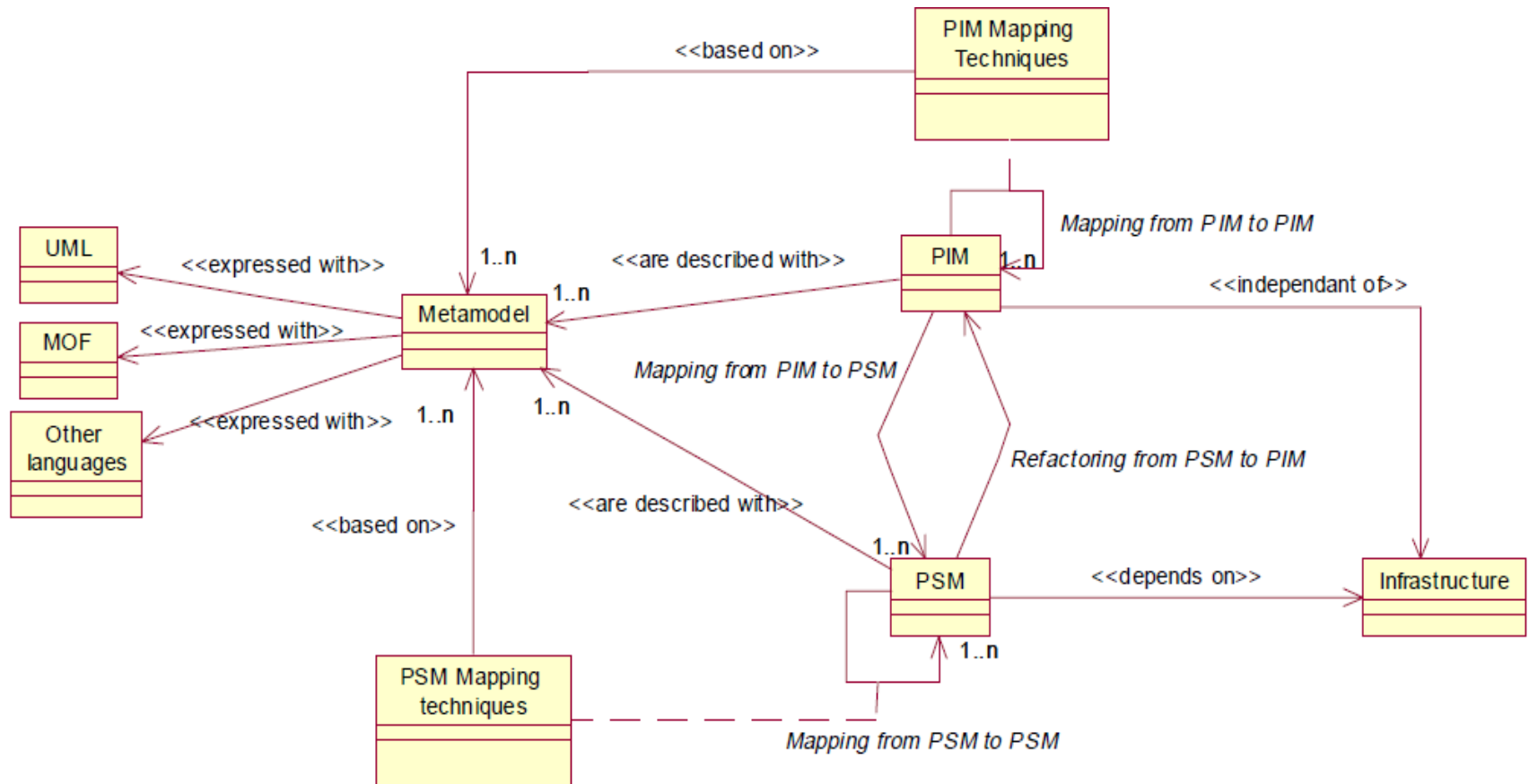
- ▶ **Approche par modélisation: Modéliser la transformation**
 - ▶ Query/View/Transformation (QVT)
- ▶ **Approche par programmation**
 - ▶ Java MetaData Interchange (JMI),
 - ▶ Eclipse Metamodel Framework (EMF)
 - ▶ MOF 2.0 to IDL(MOF vers Interface Definition Langage)
- ▶ **Approche par template**
 - ▶ Paramètres du modèle template



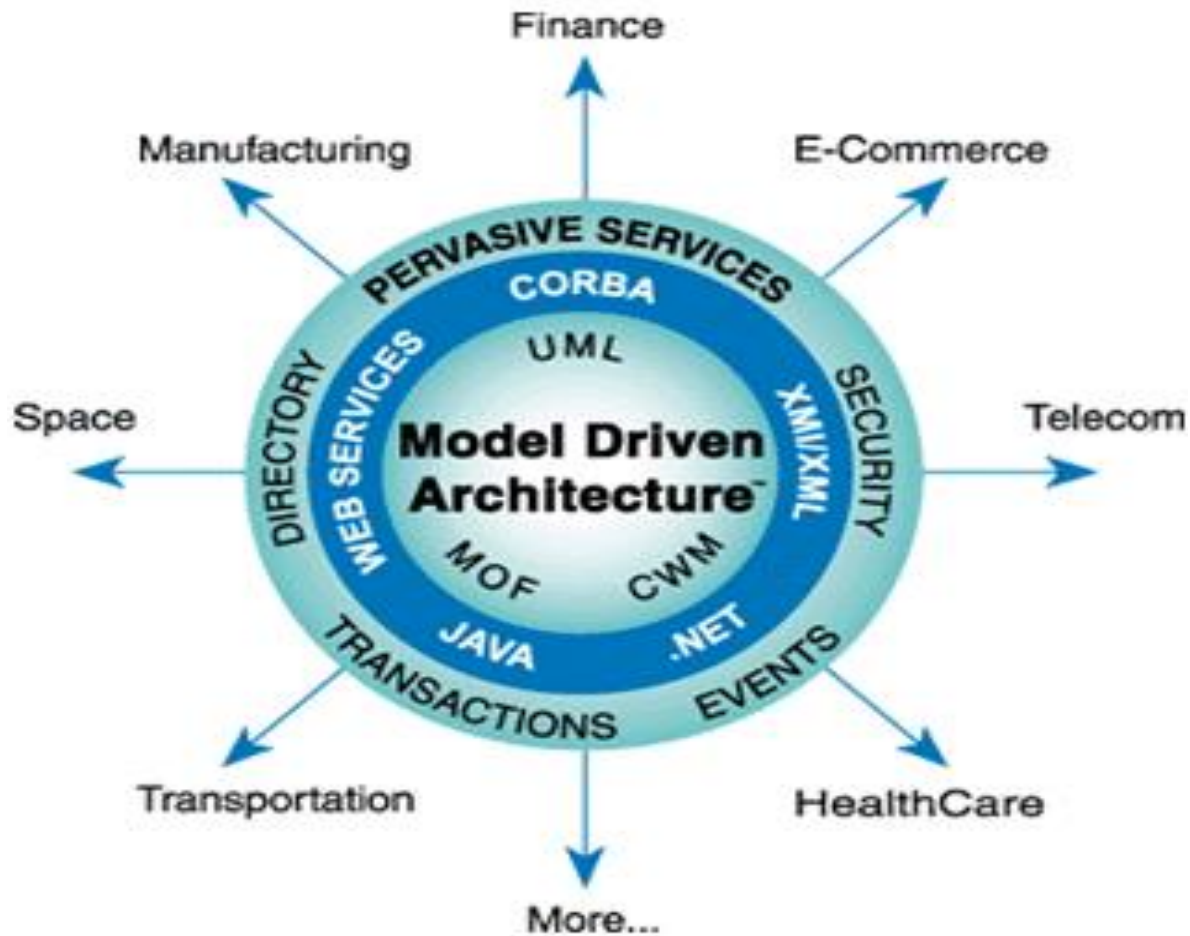
MDA ...



Description du méta-modèle de MDA



Application du MDA



Avantages et inconvénients du MDA

► Avantages

- ❖ L'indépendance de la logique métier vis à vis de la plate-forme technologique pour éviter la valse des technologies
- ❖ Passage d'une démarche corrective à une démarche préventive, voire prédictive
- ❖ Gain du temps dans le processus de développement, grâce à la transformation automatique
- ❖ La portabilité des applications développées via cette approche, etc.

► Inconvénients

- ❖ Nécessité d'une connaissance en développement ainsi qu'en conception
- ❖ Complexité de mettre en œuvre cette approche
- ❖ Le passage du PSM vers le code est ce que c'est à 100%



Bibliographie

- ▶ Xavier Blanc, MDA en action. Ingénierie guidée par les modèles. Eyrolles. 2005.
- ▶ Eric Cariou, Ingénierie des Modèles – Introduction générale, Cours de Mastère, Université de Pau et des Pays de l'Adour, France.
- ▶ Laurent Pérochon, Ingénierie dirigée par les modèles, programme, ENVOL 2008.
- ▶ OMG, Model Driven Architecture , <http://www.omg.org/mda/>

