Qu’est-ce que l'IDM ?

Thierry Millan

Vocabulaire et objectifs

Intérêt pour les modèles scientifiques et mathématiques

L'IDM : QU'EST-CE QUE C'EST ?

Principes fondateurs

Motivations

MDA

Approche Ingénierie Dirigée par les Modèles (IDM/MDE)

- **Evolution**

Programmation orientée objets

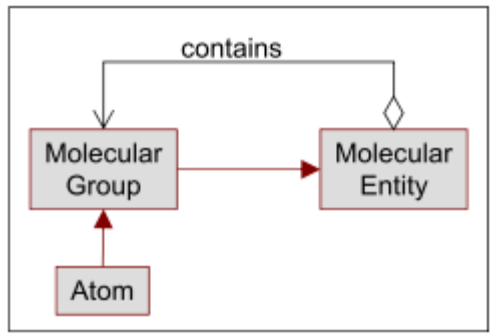
Programmation orientée composants

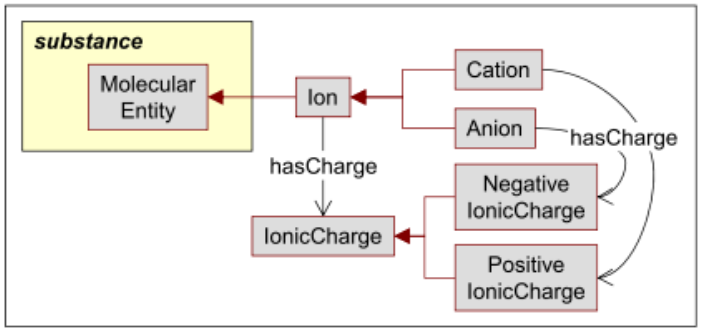
Programmation orientée modèles

- **Modèle = "Citoyen de première classe"**

Qu'est-ce que l'approche IDM ?

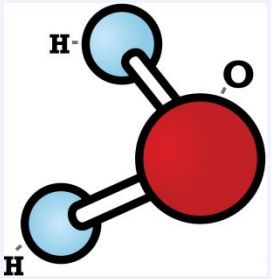
La description du modèle

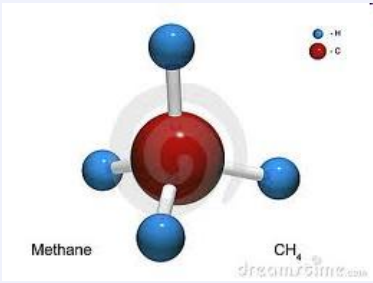




Abstrait

Le modèle





Abstrait

Le monde réel

IDM : Motivation

- **Evolution des technologies**
  - *EJB, Dot Net, Androïd, Eclipse,…*
- **Séparation métier/architecture**
  - *Séparer ce qui est pérenne de ce qui évolue vite*
- **Stabilité des algorithmes**
- **Capitalisation du savoir faire métier**
  - *Patrons de conception*
  - *Objet métier*

Il faut s'abstraire du code ➡ architecture

MDA : Fondements (1/2)

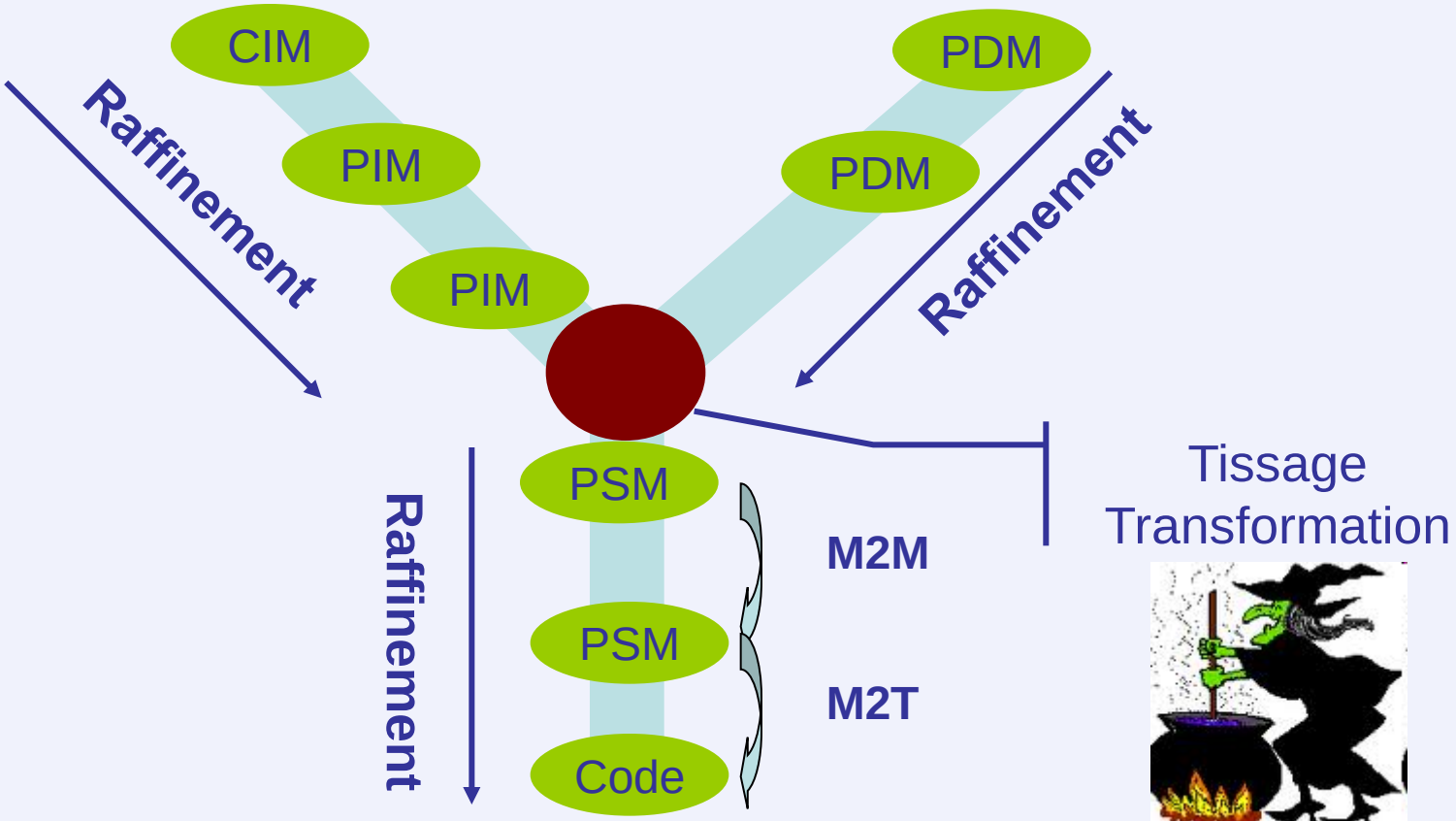
- **Séparation des préoccupations**
  - *Modèles indépendants de calculs (CIM)*
    - Besoins fonctionnels de l'application
  - *Modèles indépendants des plates-formes (PIM)*
    - Modèles d'analyse et de conception de l'application
    - Modèles mathématiques
  - *Modèles de description des plates-formes (PDM)*
    - Structure et fonctions techniques de la plate-forme d'exécution
      - Description d’une plate-forme androïd
      - Description d’un calculateur embarqué
  - *Modèles spécifiques aux plates-formes (PSM)*
    - Modèle qui se rapproche le plus du code final de l'application

MDA : Fondements (2/2)

- **Transformations de modèles**
  - *Des modèles PIM vers les modèles PSM*
  - *Des modèles PSM vers le code*

Vers des modèles productifs…

MDA : Transformation



QVT

- **Standard de l'OMG**
- **Constituants**
  - *Requête*
    - Filtrer et sélectionner des éléments d'un modèle
  - *Vues*
    - Vue : modèle déduit d'un autre pour en révéler des aspects spécifiques
  - *Transformation*
    - QVT-Relation : langage déclaratif (Prolog)
    - QVT-Core : la sémantique des concepts déclaratifs (Pascal, C)
    - QVT-Operational : langage hybride
      - Structure déclarative à base de règles
      - Utilisation d'expressions impératives

LES STANDARDS DE L'IDM

UML

DSL

XMI

MOF/Ecore

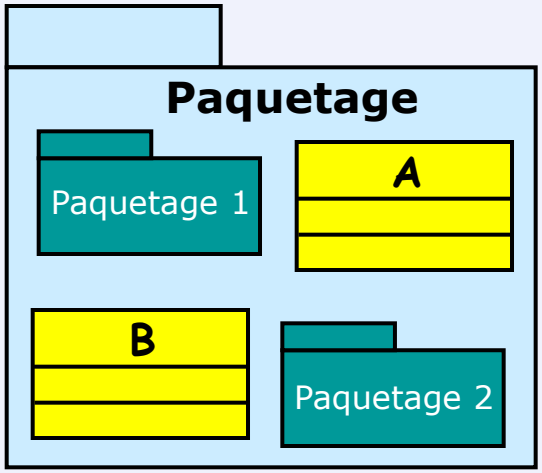
OCL

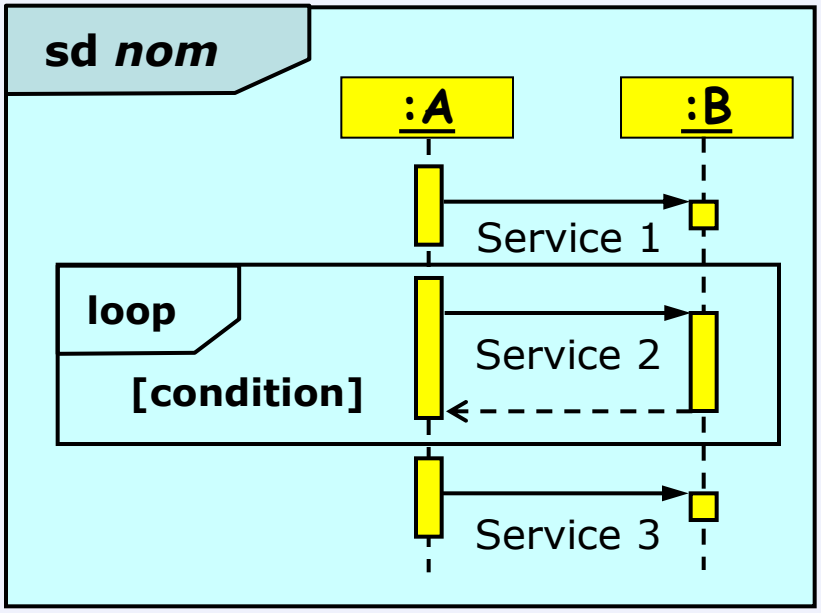
Référence à des standards

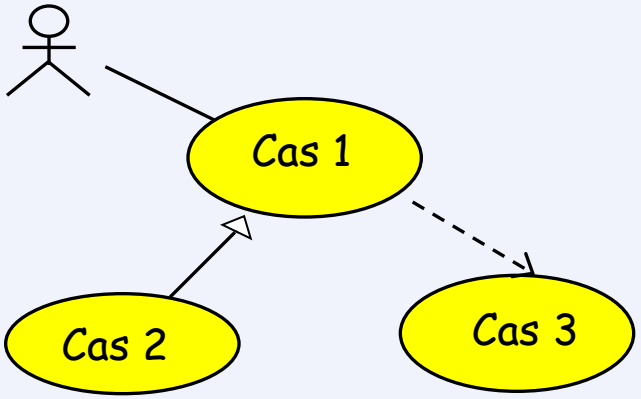
- **Modéliser**
  - *UML (Unified Modeling Language)*
  - *DSL (Domain Specific Language)*
- **Métamodéliser**
  - *MOF (Meta Object Facility de l'OMG)*
  - *Ecore (Eclipse)*
- **Echanger les modèles et des métamodèles**
  - *XMI (XML MetaData Interchange)*
- **Naviguer et interroger les modèles et métamodèles**
  - *OCL (Object Constraint Language)*

UML


A1
Evénement [Garde]
A2


Paquetage
Paquetage 1
A
B
Paquetage 2

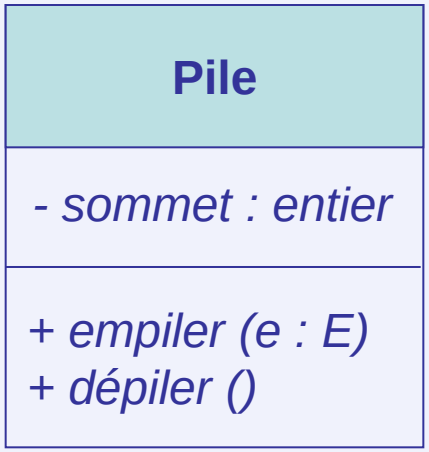

sd nom
:A
:B
Service 1
loop
[condition]
Service 2
Service 3


Cas 1
Cas 2
Cas 3

OCL

- **Langage déclaratif typé**
  - *Valeurs, expressions*
  - *Sans effet de bord*
- **Spécification formelle d'un modèle UML**
  - *Pré-conditions et post-conditions de méthode*
  - *Invariants de classe*
- **Sémantique d'UML**
  - *Règles de bonne formation des modèles UML (WFR)*
  - *Vérification statique de modèles UML*

OCL : Exemples



context Pile **inv** :

self.sommet >= 0

context Pile::dépiler()

pre : **self**.sommet > 0

post : **self**.sommet = **self**.sommet**@pre** - 1

Profils

Mécanisme d'extension générique pour UML

- **Ajouter :**
  - *Des concepts relatifs à un domaine particulier* **(stéréotype, extensions, …)**
  - *Des contraintes sur les associations entre ces concepts* **(OCL)**
  - *Des contraintes sur l'utilisation ou non de certains concepts selon le contexte* **(OCL)**
- **Changer la représentation de ces concepts (stéréotype)**
- **Fixer les points de variation sémantique (langage naturel)**

DSL

- **Utilisation de tournures idiomatiques au niveau d'abstraction du domaine traité**
  - *Proche des experts car logique métier*
- **Documentation du code simplifiée**
- **Amélioration de la qualité, la productivité, la fiabilité, la maintenabilité, la portabilité et les possibilités de réutilisation**
- **Validation au niveau du domaine**

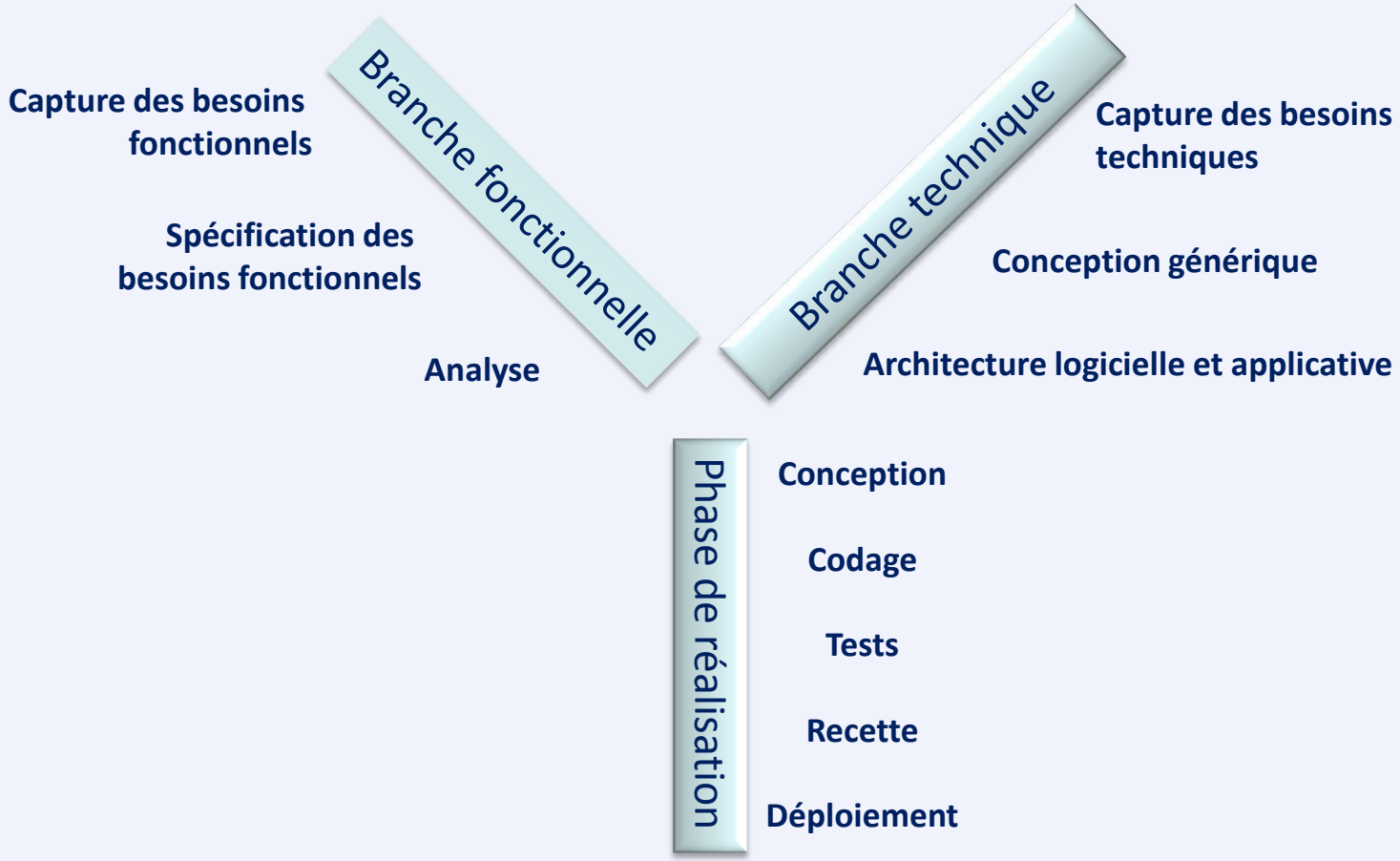
Un domaine = un langage ➡ beaucoup de langages

IDM ET INGÉNIERIE

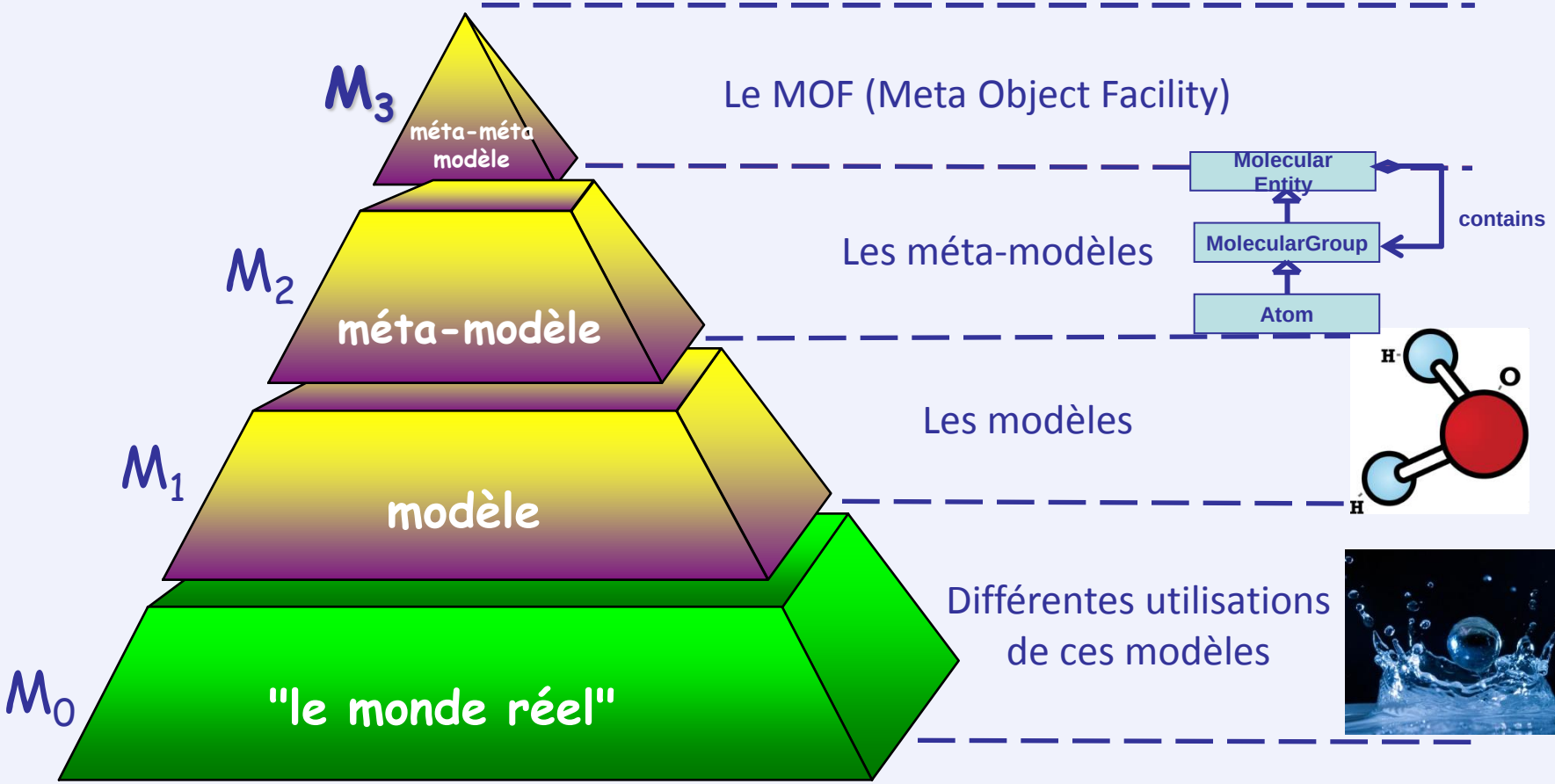
Qu'est ce que l'ingénierie des modèles ?

Démarche de développement qui conçoit l'intégralité du cycle de développement du logiciel comme un processus de production, de raffinement itératif et d'intégration de modèles

Démarche


Branche fonctionnelle
Capture des besoins fonctionnels
Spécification des besoins fonctionnels
Analyse
Branche technique
Capture des besoins techniques
Conception générique
Architecture logicielle et applicative
Phase de réalisation
Conception
Codage
Tests
Recette
Déploiement

architecture à 4 niveaux

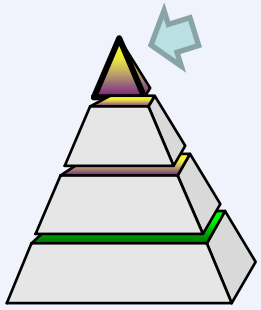


Qu'est-ce qu'un métamodèle ?

Modèle qui définit un langage d'expression d'un modèle

- **Syntaxe**
  - *Sous-ensemble d'un diagramme de classes UML*
- **Sémantique**
  - *Axiomatique*
    - Syntaxe abstraite + OCL par exemple
  - *Opérationnelle*
    - Langage d'action : Kermeta, EMF, USE
    - Langage de transformation : QVT, transformation de graphes
  - *Dénotationnelle*
    - Transformation vers un autre espace de modélisation
      - Model checking

MOF - Ecore



- **Métamétamodèle**
  - *Modèle des concepts d'un métamodèle (d'un langage)*
  - *Définition d'une syntaxe et d'une sémantique*

Concepts pour définir tout métamodèle

- *Diagramme de classes pour la syntaxe abstraite*
- *Règles OCL pour la sémantique*

- **Vision OMG : noyau d'UML**
  - *Métamodèle UML (première expérience) conforme au MOF*
  - *Description d'un métamodèle par conformité au MOF*

Approche formelle

- **IDM utilisant des formalismes basés sur les mathématiques**
  - *Langage B, Z, …*
    - Utilisation de la logique et de prouveur

Complexe à mettre en œuvre

Ne permet pas de modéliser la totalité d'un modèle mais uniquement les parties critiques

Approche semi-formelle

- **IDM basé sur les approches graphiques**
  - *UML*
  - *SysML*

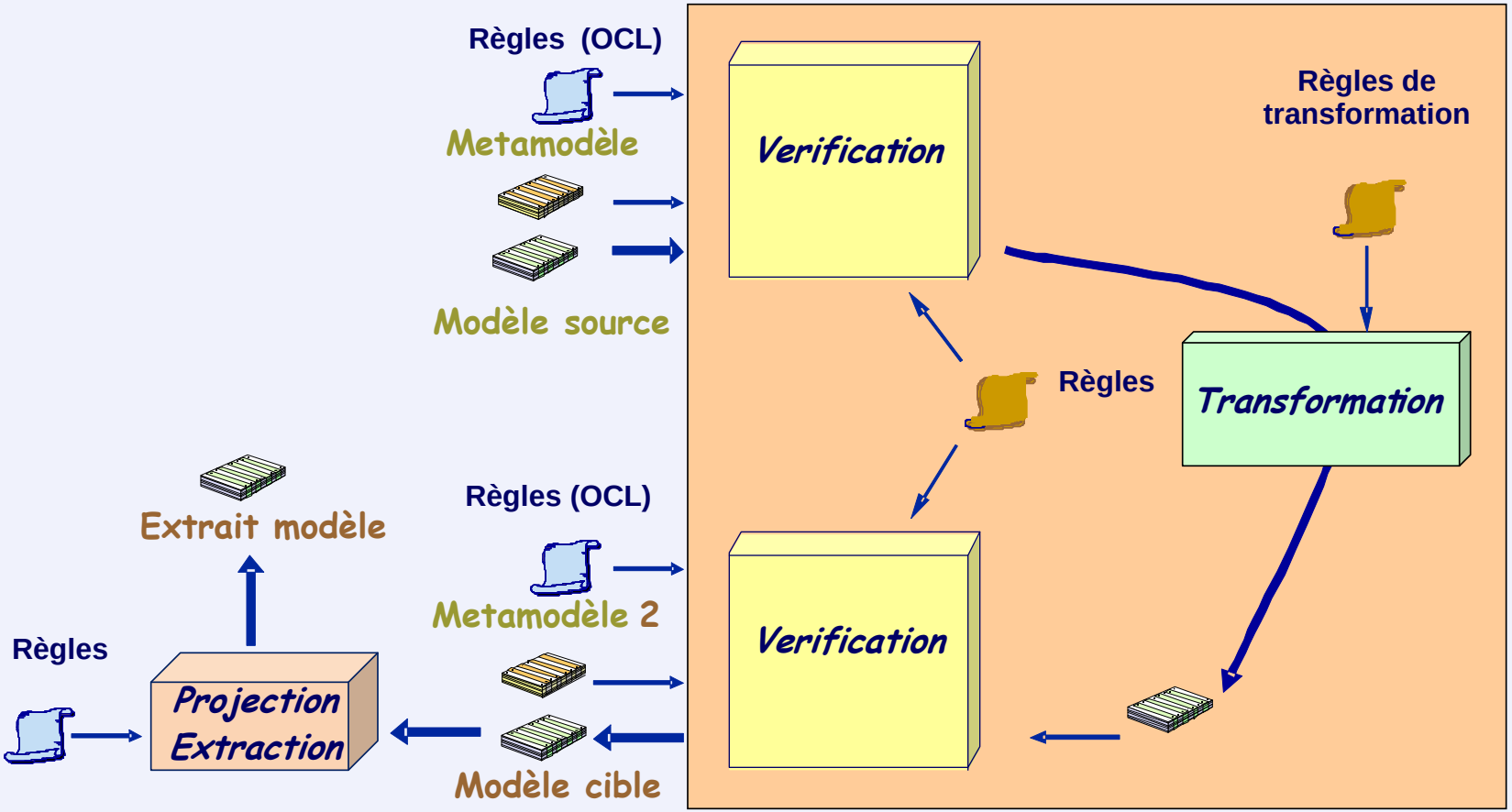
Sémantique souvent incomplètement définie

⇒ Utilisation de formalisme comme OCL

Utilisation massive du langage naturel pour décrire les modèles

Besoin de vérification et de simulation pour les valider

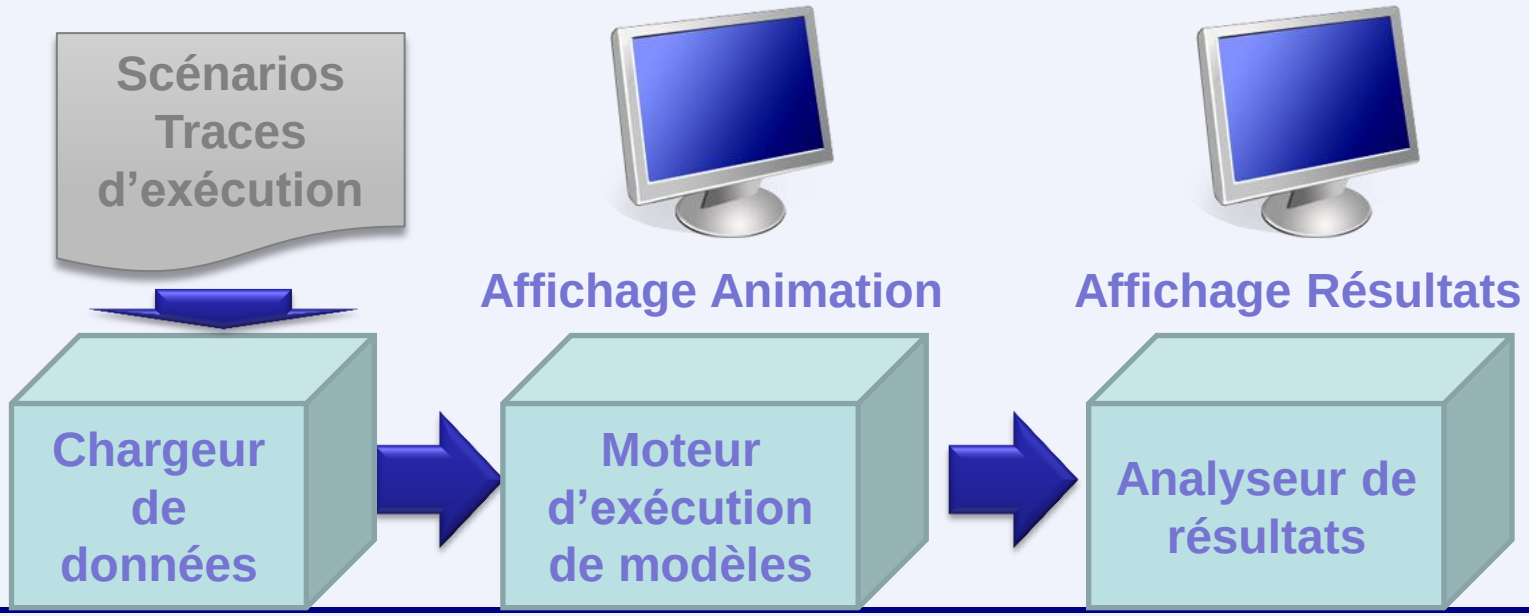
Transformer, vérifier et projeter



Simuler

Reproduction artificielle de ce que pourrait être le programme exécutable

Sémantique opérationnelle/dénotationnelle



Quelques Outils IDM

Offre	Editeur	Licence	Type d'outil
Acceleo	Obeo	EPL (com.)	Générateur de code qui permet de transformer des modèles vers du code (M2T) en utilisant une approche MDA
Xtext	Fondation Eclipse	EPL (gratuit)	Framework permettant de mettre en œuvre votre propre DSL textuel ou même de mettre sur pied un langage de programmation générique
Topcased	Topcased	EPL (graduit)	Boîte à outils IDM pour le développement d'applications critiques et de systèmes
Polarsys	Eclipse Industry WG	EPL (gratuit)	Développement d'un outillage d'ingénierie pour les systèmes embarqués du monde de l'aérospatial, de la Défense, de l'avionique et des infrastructures télécoms
Kermeta	IRISA	EPL (gratuit)	Langage de métamodélisation disposant d'un environnement de développement de métamodèles basé sur EMOF
Modelio	Modeliosoft	GPL V3, APL (com.)	Environnement complet d'ingénierie pilotée par le modèle
SmartQVT	France Telecom	EPL (gratuit)	Compilation des transformations de modèles exprimées en QVT pour produire du code Java qui exécute ces transformations.

L'IDM EN EXEMPLE

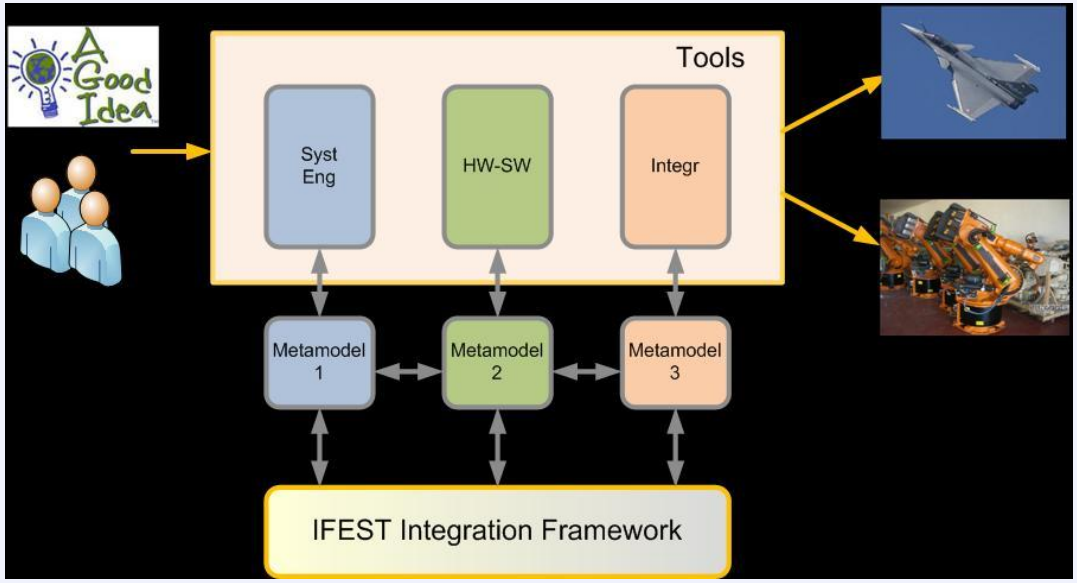
Projet iFest

Transformation SysML – Modelica

Transformation SysML – MATLAB/Simulink

iFest

- **Objectif global :**
  - *fournir un environnement intégré de développement de systèmes embarqués logiciel et matériel basé sur l'utilisation des modèles pour le processus, l'application et le modèle d'exécution support*



Transformation SysML modelica

- **Standard de l'OMG : *formal/2012-11-09***

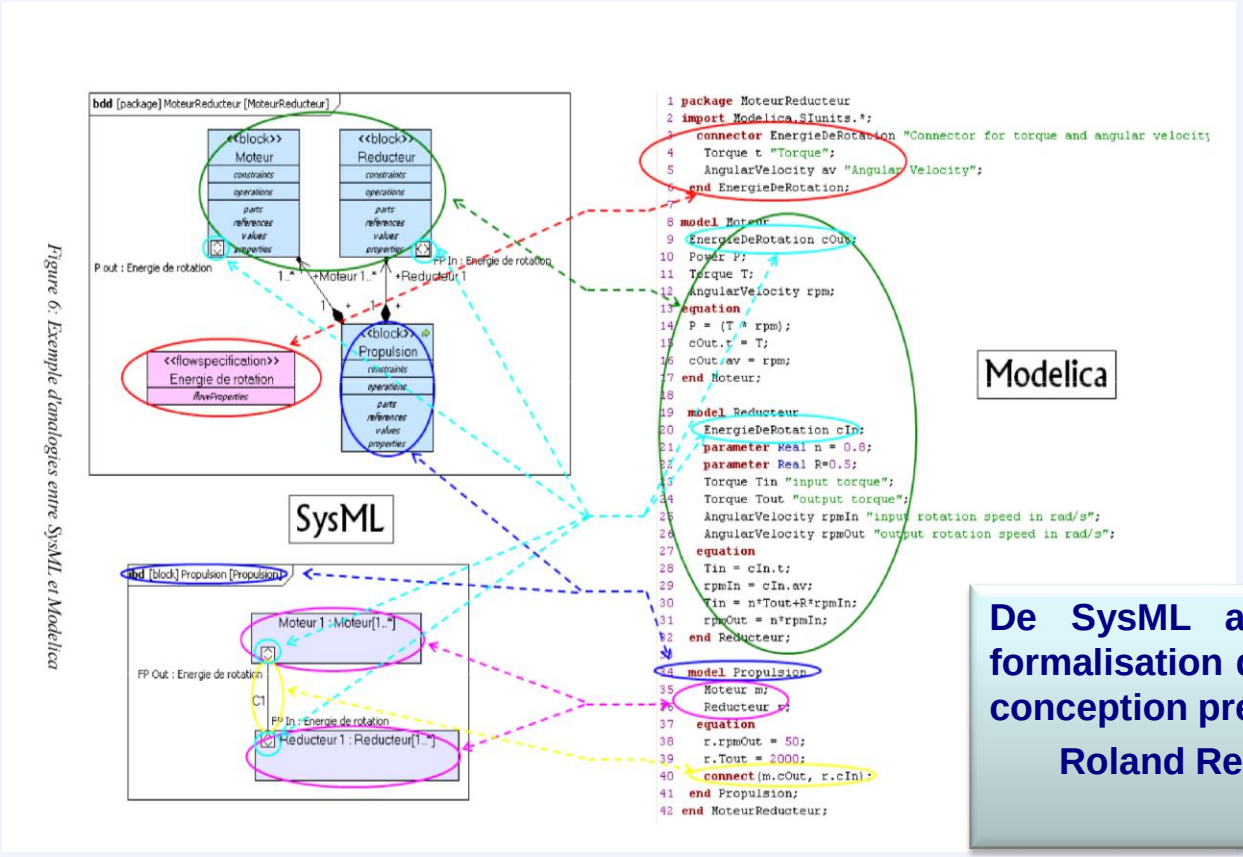


Figure 6: Exemple d'analogies entre SysML et Modelica

De SysML a MODELICA : aide a la formalisation de modèles de simulation en conception préliminaire

Roland Renier, Raphaël Chenouard

ECN-IRCCyN,

Transformation SysML MATLAB/Simulink

- **Functional validation with a practical SysML / Simulink transformation**
  - *Ray Snyder, David Bocktaels, et Xavier Feigenbaum - B2i Automotive (présentation disponible sur le site NEPTUNE)*
- **From UML/SysML to Matlab/Simulink: Current State and Future Perspectives**
  - *Yves Vanderperren, Wim Dehaene - Dept. of EE, Katholieke Universiteit Leuven*

Merci de votre attention

Questions