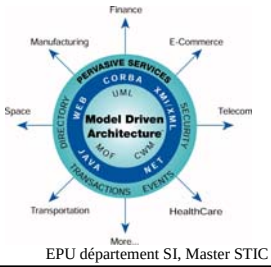
Model Driven Development /Engineering
Ingénierie Dirigée par les Modèles (IDM)

Mireille Blay-Fornarino

http://www.polytech.unice.fr/~blay/ENSEIGNEMENT/IDM/



Plan du module

Toutes les infos :
http://www.polytech.unice.fr/~blay/ENSEIGNEMENT/IDM/

Principes : Cours + Pratique + Projets

Evaluation : Projet + devoir sur table

Outils : presque au choix pour les projets
Obligatoire en cours/TP : http://**openembedd**.inria.fr/Download/

Avez-vous déjà vu OCL?

3

PLAN du jour

- **I. Introduction**
  - Pourquoi une « encore » nouvelle approche du développement logiciel ?
  - Vision de l'OMG
- **II. Principes généraux de l'ingénierie dirigée par les modèles**
  - Modèle ?
  - Correspondances entre modèles
  - Transformations de modèles
- **III. Quelques Ingrédients MDA**
  - a) MOF, b) UML c) XML
- **IV. Au delà du MDA**
- **V. Développement dirigé par les domaines : Programmation orientée domaine**
- **VI. Quelques outils …**
- Discussion sur les projets

4

I. Introduction

a) **Pourquoi une « encore » nouvelle approche du développement logiciel**
   - Etat des lieux
   - **Des exemples:**
     - Filtres
     - Interactions

b) Vision de l'OMG

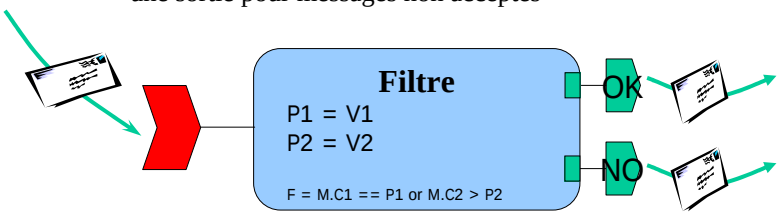
I. Introduction

a) **Pourquoi une « encore » nouvelle approche du développement logiciel**
   - **Etat des lieux**
   - **Des exemples:**
     - Filtres
     - Interactions

b) Vision de l'OMG

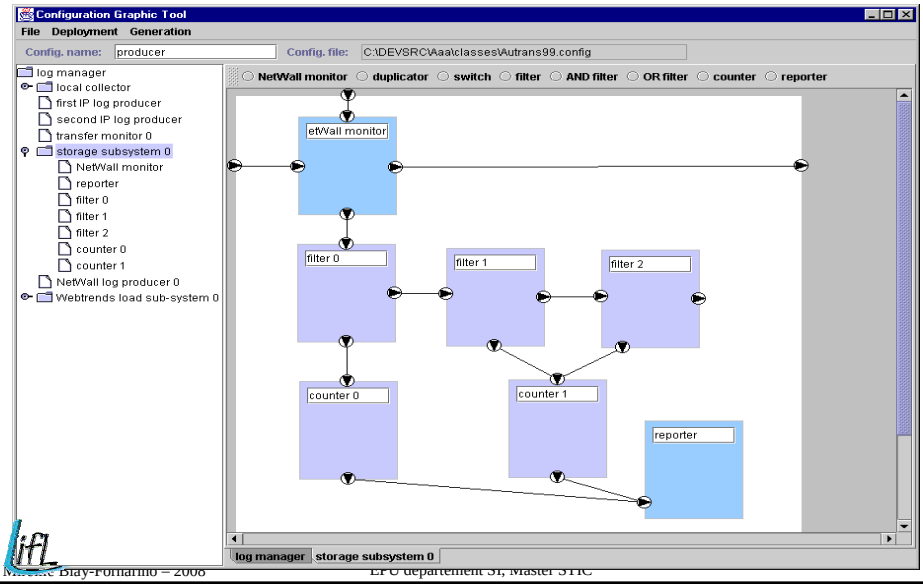
Exemple : filtrage de messages

- Filtre
  - une entrée pour recevoir les messages à filtrer
  - des propriétés de filtrage configurables
  - une fonction de filtrage
  - une sortie pour messages valides
  - une sortie pour messages non acceptés



- Application = assemblage de filtres

Application = Assemblage de Filtres



8

Exemple de filtre implémenté avec le CCM

```
eventtype Message {
  . . . Message fields . . .
};
component Filter {
  consumes Message input;
  publishes Message accepted;
  publishes Message rejected;
  . . . Configurable filtering properties . . .
};
home FilterHome manages Filter {
  factory create_filter(. . .);
};
```

CCM = Corba Component Model

Plusieurs plates-formes : Ingénierie des filtres

9

- Matérialisation d'un filtre
  - si AAA alors 1 agent
  - si EJB alors 1 Message Bean
  - si CCM alors 1 composant avec ports asynchrones
- Programmation explicite
  - Java pour AAA ou EJB
  - multi-langages pour CCM
- Exécution répartie
  - AAA = MOM fiable
  - EJB = JMS (Java Messaging Service (JMS))
  - CCM = service de notification
- Si peu de filtres à produire alors OK
- Quid si des millions de filtres différents ?

Problèmes

10

- Tout doit être fait à la main!
  - Définition des OMG IDL pour les évènements, composant filtre and home ...
  - Implémentations
  - Ecriture des descripteurs CCM XML
- Très Verbeux, consommateur de temps et générateur d'erreur!!!
- Comment améliorer la productivité et la qualité?
- Comment gérer des milliers de messages et filtres?
- Comment déployer cette application sur d'autres technologies comme Message Oriented Middleware (MOM)?
  - ex EJB 2.0 message beans

A Message Filtering XML Descriptor


```
<MFL>
  <message id=«Email»>
    <field name=«to» type=«string»\>
    <field name=«from» type=«string»\>
    <field name=«title» type=«string»\>
    <field name=«contents» type=«string[]»\>
  </message>
  <filter id=«FromFilter»
          appliedOn=«Email»
          function=«e.from == from»>
    <property name=«from» type=«string»\>
  </filter>
</MFL>
```


The Message Filtering XML DTD


```
<!ELEM MFL (message, filter)*>
<!ELEM message (field)*>
<!ATTRLIST message id ID>
<!ELEM field EMPTY>
<!ATTRLIST field
    name CDATA
    type CDATA>
<!ELEM filter (property)*>
<!ATTRLIST filter
    id ID
    appliedOn IDREF
    function CDATA>
<!ELEM property EMPTY>
<!ATTRLIST property
    name CDATA
    type CDATA>
```

XML est-il la solution ?

13

- => Non juste une syntaxe de transfert !
- => « stringware » !

- Mais de nombreux parsers XML !

- Quid de la vérification sémantique ?
- Quid de la génération de code ?
- Quid élégance / complétude de ce langage ?

Définition d'un langage (DSL?) : Exemple de formalisme BNF

14

```
message Email {
  field string to, from, title;
  field string[] contents;
};
filter FromFilter appliedOn Email {
  property string from;
  function(e) is e.from == from;
};
```

- Génération du compilateur, e.g. YACC, JavaCC, ...
- Quid de la vérification sémantique ?
- Quid de la génération de code ?
- Quid élégance/évolution de ce langage ?

15

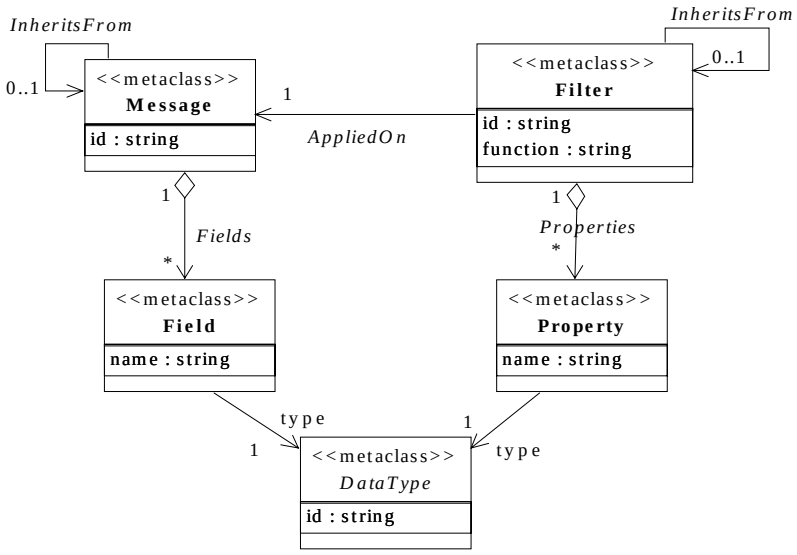
Exemple de Filtrage de Messages avec MDA

- Définir la notion de filtre dans un métamodèle
  - Structure des messages, i.e. champs des messages
  - Propriétés de filtrage et fonction
- Utiliser XMI pour obtenir un XML DTD or Schema
  - Syntaxe pour sérialiser les modèles conformes au métamodele.
- Définir un “UML profile” pour les utilisateurs finaux
- Définition des transformations vers un MOM Plateform Specific Model (PSMs)
  - Génération des composants filtres de messages automatiquement

=> Amélioration de la productivité et de la qualité
=> Réduction des coûts de développement
=> Capitalisation des Modèles (PIM) quand de nouveaux MOM apparaissent.

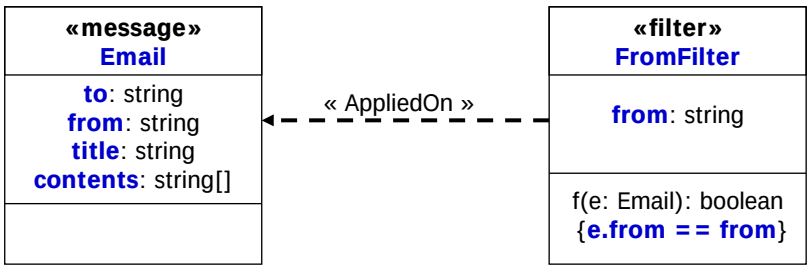
16

The Message Filtering Meta-Model



17

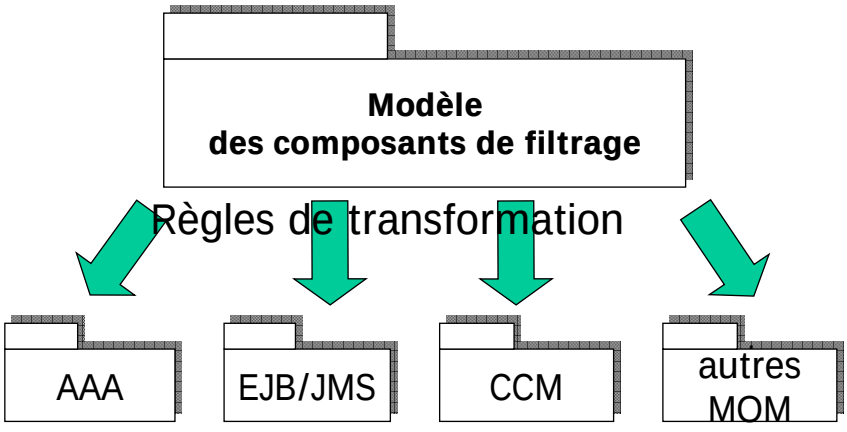
Utilisation du Profil UML pour les filtres de messages



- Ce profil UML utilise des concepts UML pour adhérer aux concepts du "Message Filtering Meta-Model" concepts quand ils s'y prêtent bien et sont facile à manipuler en UML.
- Vérification = contraintes du profil
- Génération = méthodes du profil …

18

Indépendance vis-à-vis de la technologie



•UML profil: Solution élégante mais « encore ?» liée atelier UML !

The Message Filtering PIM to CCM Transformation

19

```
eventtype Email {
  public string from, to, title;
  public sequence<string> contents;
};
component EmailFilterBase {
  consumes Email input;
  publishes Email accepted;
  publishes Email rejected;
};
component FromFilter : EmailFilterBase {
  attribute string from;
};
home FromFilterHome manages FromFilter{
  factory create_filter(in string from);
};
```

CCM = Corba Component Model

Etape suivante pour la définition d'un métamodele de "Message Filtering"

20

- Maintenant, nous avons seulement :
  - La définition des types de messages et de filtres
  - La génération des types OMG IDL types, leurs implementations…
- Il nous reste à :
  - Compléter le métamodèle pour décrire des instances de filtres et les connections entre eux
  - Puis automatiser la génération des descipteurs d'assemblages (XML) pour permettre un déploiement automatique.

I. Introduction

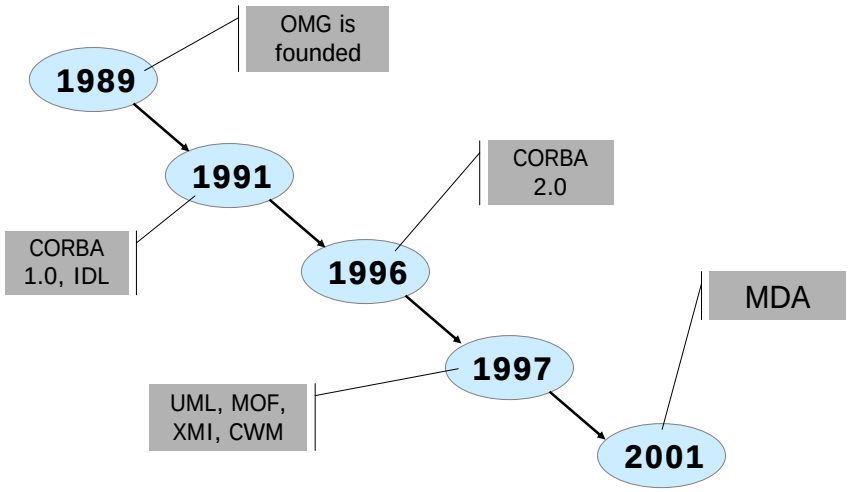
a) Pourquoi une « encore » nouvelle approche de la programmation?

Des exemples:

- Filtres
- Interactions

Vision de l'OMG : MDA est la partie standardisée du MDD par l'OMG.

OMG's Milestones



MDA: La nouvelle vision de l'OMG

23

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. November 27, 2000

OMG is a very important organization, but three years later, nobody still thinks that this is a local change to this organization.
It's much more important and will probably affect our whole profession in the next thirty years.

Mireill (appel à contribution)

Why modeling: master complexity

24

- 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.

Jeff Rothenberg.

Model Driven Architecture (MDA)

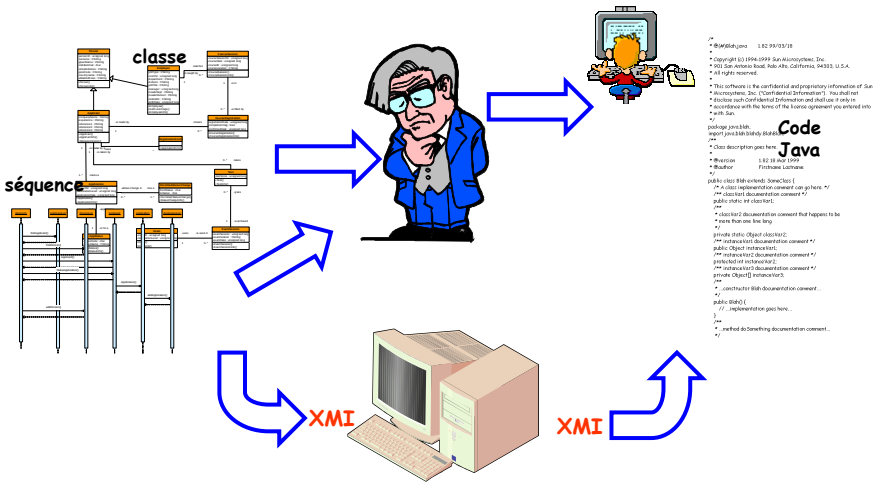
26

- Nouvelle orientation (fondation) des activités de l'OMG
  - successeur de l'OMA
- Focaliser sur les modèles de composants logiciels pour les métiers (Platform Independent Models (PIM))
  - via MOF, UML, profil UML, ASL, CWM, SPEM, …
  - sans les détails technologiques
- Travail sur les modèles « abstraits » de Plateforme (Platform Specific Models (PSM))
- Définir des transformations vers les middlewares existants
  - fixer les détails technologiques

➢ Préserver les modèles quand arriveront de nouveaux intergiciels !

Du contemplatif au productif

28

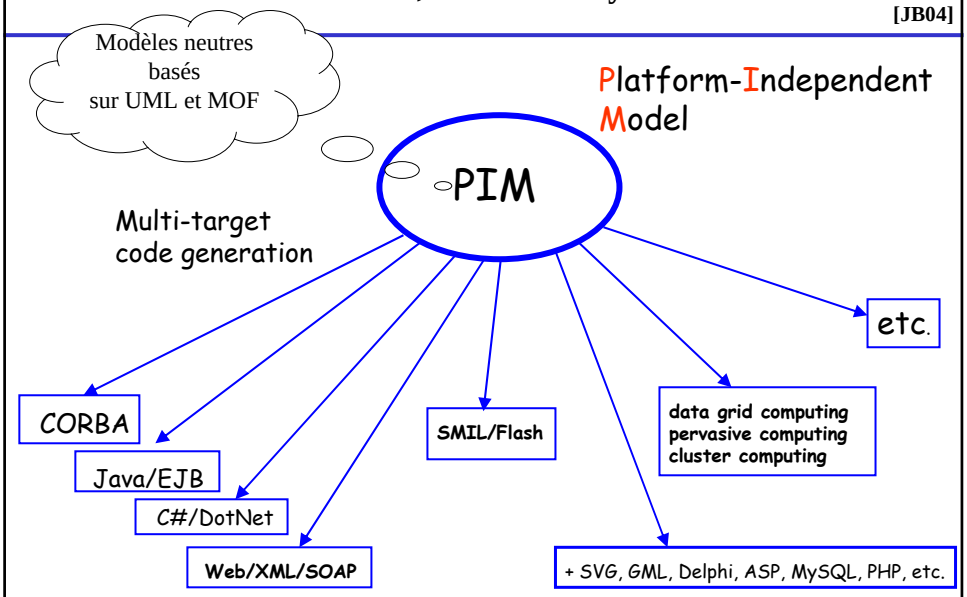


"from human-readable to computer-understandable"

29

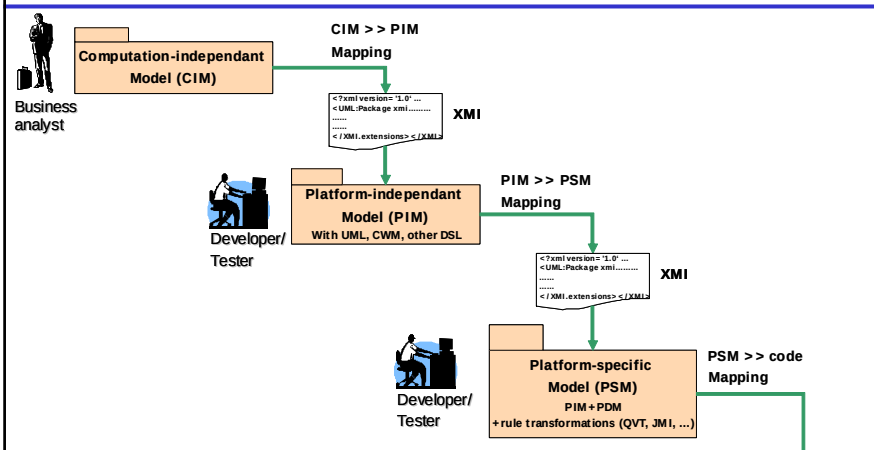
~~Write Once, Run Anywhere~~
Model Once, Generate Anywhere

[JB04]



30

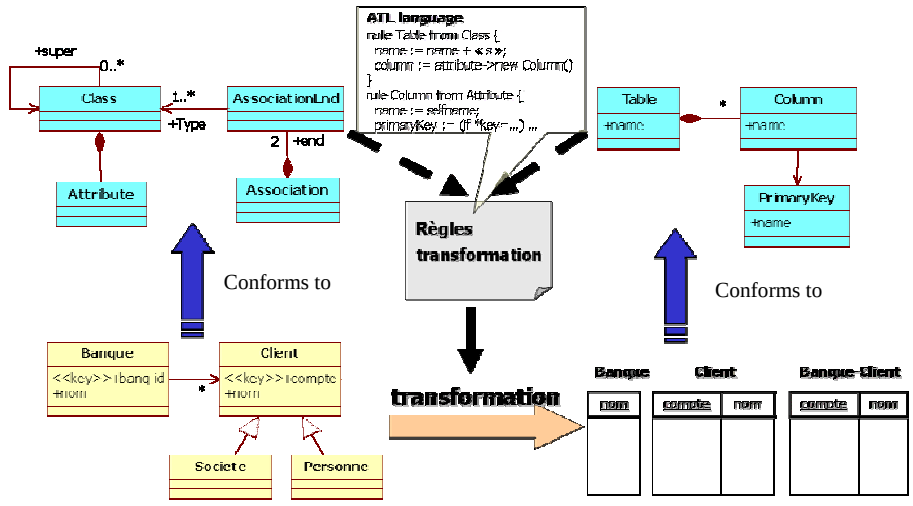
Etapes du processus MDA



CWM : Langage de modélisation pour les entrepôts de données, Common Warehouse Meta-model
XMI : XML Model Interchange, le standard des échanges entre modèles
PDM : Plateform Description Model

www.sqli.com/ressources/files/IBCom_mai2006_MDE**MDA**_ecourte.doc

Principes “généraux” MDA de génération de code



II. Principes généraux de l’ingénierie dirigée par les modèles

a) Modèle ?

b) Correspondances entre modèles

c) Transformations de modèles

II. Principes généraux de l'ingénierie dirigée par les modèles

a) Modèle ?
b) Correspondances entre modèles
c) Transformations de modèles

II. Principes généraux de l'ingénierie dirigée par les modèles

a) Modèle ?
b) Correspondances entre modèles
c) Transformations de modèles

37

III. Quelques ingrédients MDA

a) MOF
b) UML
c) XML

38

MDA ingrédients

- Meta Object Facility
  – Foundation for OMG Metadata and Modeling architecture
- Unified Modeling Language
  – The UML is a graphical language for specifying, visualizing, constructing, documenting the artifacts of software systems
- XML Metadata Interchange (XMI)
  – Use W3C Extensible Markup Language (XML) for the transfer syntax and interchange format for models
- Platforms and Mappings
  – CORBA, CORBA Component Model, .NET, J2EE

http://www.omg.org/news/meetings/workshops/UML2002-Manual/07-1_A_UML_Profile_for_CORBA_Components.pdf

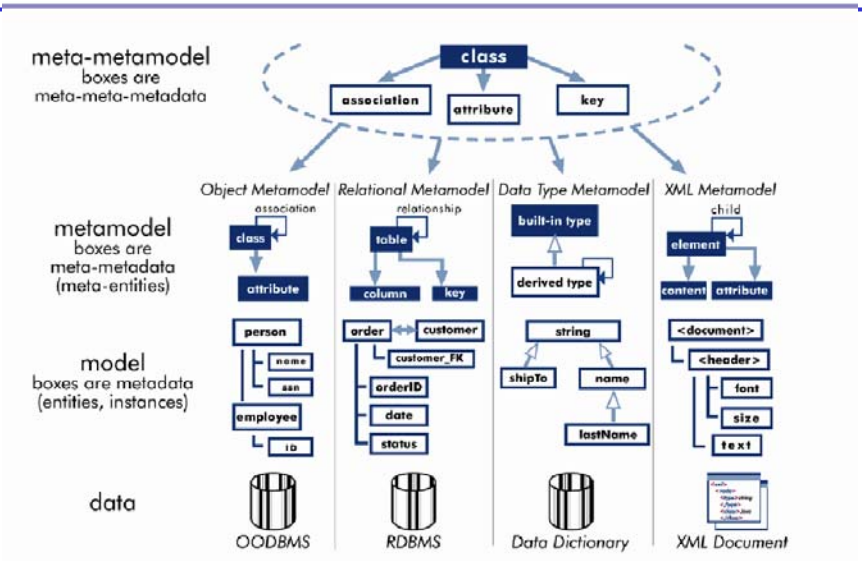
III. Quelques ingrédients MDA

a) MOF

b) UML

c) XML

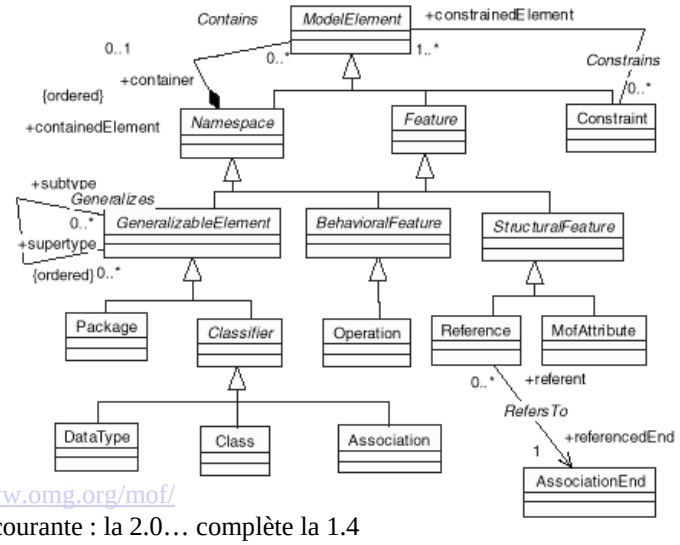
OMG's Meta- Object Facility (MOF)



http://www.omg.org/news/meetings/workshops/UML2002-Manual/03-3_Enterprise_Information_Integration_and_the_OMGs_MDA_and_MOF.pdf

41

Fragment du modèle MOF 1.1



http://www.omg.org/mof/

Version courante : la 2.0… complète la 1.4

42

III. Quelques ingrédients MDA

a) MOF

b) UML

c) XML

- UML 2.1.1 Infractruture (2007) : 220 pages
- UML 2.0 OCL (may 2006) : 236 pages
- UML 2.0 Diagram Interchange : 36 pages
- UML superstructure (2007) : 732 pages

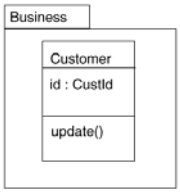
III. Quelques ingrédients MDA

a) MOF

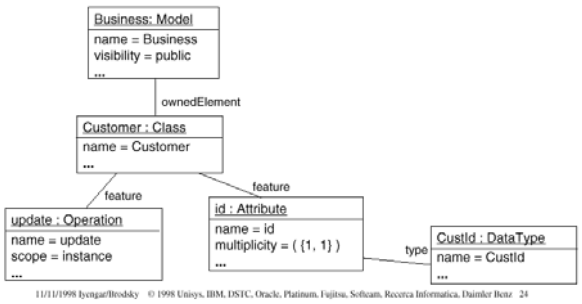
b) UML

c) XML

UML - MOF via XMI



UML




```
<Model>
  <name>Business</name>
  <visibility xmi.value="public"/>
  <Class>
    <name>Customer</name>
    <feature>
      <Attribute>
        <name>id</name>
        <multiplicity>
          <XMI.field>1</ XMI. field>
          < XMI. field>1</ XMI. field>
        </multiplicity>
      </Attribute>
    </feature>
  </Class>
</Model>
```

47

Productivité du MOF

DTD Généré

```
<!ENTITY % fixedDTD SYSTEM "XmiFixed.dtd">
% f i x e d D T D ;
<!ATTLIST XMI xmlns:RDB CDATA #IMPLIED >
<!-- PACKAGE: RDB:SimpleRDB -->
<!-- ***** RDB:tableHasColumn ******* -->
<!ELEMENT RDB:Table.column ( RDB:Column)* >
<!-- CLASS: RDB:Table -->
<!ELEMENT RDB:Table.name (#PCDATA|XMI.reference)*>
<!ENTITY % RDB:TableProperties '((RDB:Table.name)?)' >
<!ENTITY % RDB:TableCompositions '(RDB:Table.column*)' >
<!ENTITY % RDB:TableAttPropsList 'name CDATA #IMPLIED' >
<!ELEMENT RDB:Table ( %RDB:TableProperties;
                ,(XMI.extension* )
                , %RDB:TableCompositions; )?>
<!ATTLIST RDB:Table %RDB:TableAttPropsList; %XMI.element.att;
                % X M I . l i n k . a t t ; >
<!-- CLASS: RDB:Column -->
<!ELEMENT RDB:Column.name (#PCDATA|XMI.reference)*>
<!ELEMENT RDB:Column.type (#PCDATA|XMI.reference)*>
<!ENTITY % RDB:ColumnProperties '((RDB:Column.name)?
                ,(RDB:Column.type)?)' >
```

Interfaces générées

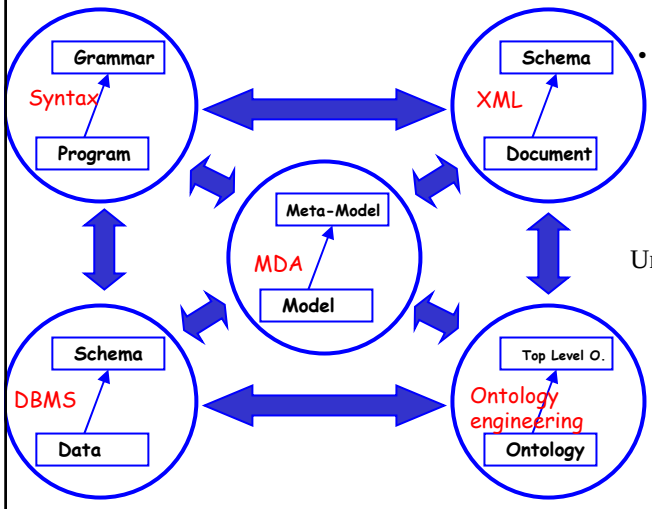
```
interface Table : TableClass
{
string name ()
raises (Reflective::MofError);
void set_name (in string new_value) raises (Reflective::MofError);
ColumnSet column () raises (Reflective::MofError);
void set_column (in ColumnSet new_value)
                raises (Reflective::MofError);
void add_column (in SimpleRDB::Column new_element)
                raises (Reflective::MofError);
void modify_column ( in SimpleRDB::Column old_element,
                in SimpleRDB::Column new_element)
                raises (Reflective::NotFound, Reflective::MofError);
void remove_column (in SimpleRDB::Column old_element)
                raises (Reflective::NotFound, Reflective::MofError);
}; // end of interface Table
```

48

IV. Au delà du MDA

La notion d'espace technologique

49



- Un espace technologique s'organise autour d'un ensemble de concepts
- Les espaces sont reliés par des "ponts" de transformations

MDA/OMG =
Un unique méta métamodèle

……

Est-ce réaliste ?

IDM se positionne différemment

Une leçon du passé

50

" Another lesson we should have learned from the recent past is that the development of "richer" or "more powerful": programming languages was a mistake in the sense that these monstrosities, these conglomerations of idiosyncrasies, are really unmanageable, both mechanically and mentally. I see a great future for very systematic and very modest programming languages. "

E.W. Dijkstra
The Humble Programmer
Communications of the ACM, Oct 1972

V. Développement dirigé par les domaines Programmation orientée domaine

Dave Thomas and Brian Barry, OOPSLA 2003

Langages spécifiques aux domaines

- *A domain-specific language (DSL) is a programming or executable specification language that offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain.*

Domain-Specific Language

54

En général

- Petit (micro langage)
- Souvent déclaratif
  - Langage de spécification et de programmation
  - Générateurs de code/composants (Yacc)/Documents(TEX)…
- Expressif à destination de l'utilisateur final (Excel macro-Language)

> A *domain-specific language* (DSL) is a programming language or executable specification language that offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain. [1]

Domain-Specific Languages

55

- Cobol, Fortran, Lisp
  -> gestion,calculs numériques, calculs symboliques
  Evolution vers des langages génériques

  Adaptation aux domaines
  - Bibliothèques de code
  - « Frameworks » orientés objets ou composants
  - Un langage spécifique

DSLs exemples

- Logiciels
  - Produits financiers (Riska)
  - Contrôles et coordination
  - Architectures logicielles (ADL)
  - Base de données
- Systèmes
  - Description et analyses d'arbres de syntaxes (Lex/Yacc),
  - Gestion de documents (LaTeX),
  - Gestion de video…
- Multi-médias
  - Interfaces web (Bigwig), manipulation d'images, animation et dessin 3D
- …….

Mais aussi

UML, QVT, SPEM (Software Process Engineering Meta-model), ASL (Action Semantics Language)…

DSL Avantages

+ Niveau Domaine : accessible à l'expert lui-même
+ Concision
+ Améliore: productivité, fiabilité, maintenance
+ Capture les connaissances d'un domaine et permet ainsi la conservation et la réutilisation de la connaissance.

☹ Coûts de modélisation, implémentation et maintien du DSL
☹ Coûts de l'apprentissage
☹ Difficultés de déterminer le champ d'application du langage

DSL Implementation

58

- Création de compilateur ou interpréteur dédié :
  - Optimisation à la fois dans l'expressivité et les choix de mises en œuvre
  - coût élevé
- Extension d'un langage donné :
  - Bénéficie de la puissance du langage sous-jacent

Librairie

- limitation dans l'expressivité

Pré-processing ou code génération

- simple mais détection des erreurs au niveau du langage cible ou seulement à l'exécution

Extension d'un compilateur existant (Tcl)

En conclusion, MDA, Une approche structurante pour construire les futurs modèles

59

- Caractériser le nouveau modèle abstrait
  - focaliser sur les concepts et non sur la syntaxe BNF
  - via MOF ou profil UML
- Décrire les transformations vers des modèles existants
  - via XSLT, QVT, ATL, ….
  - La transformation des PIMs en PSMs devra, **à long terme**, être entièrement automatique et réversible.
- Obtenir automatiquement
  - l'atelier pour spécifier les instances des modèles
  - le langage XMI associé = plus de BNF imparfaite !!!
  - le référentiel associé
  - les compilateurs et générateurs
  - les consoles d'administration
  - La transformation des PSMs devra, **à court terme**, être entièrement automatique, réversible et paramétrable.
- Enorme gain de productivité dans l'ingénierie des modèles de composants
  - CCM EJB .NET ~ 5 ans & Combien M$?

Outils orientés MDA (MDE)

60

- Peu d'outils "MOF compliant" pour expérimenter la création de nouveaux métamodèles! EMF/Eclipse joue un rôle de plus en plus important…
- Pas de portabilité des profils UML entre outils UML !!!
- Des formalismes pour exprimer les MDA transformations … lequel choisir?
  - QVT : 7/2007
- Peu de PSMs pour
  - Langages de programmation comme Java, C++, …
  - Component standard middleware
- Deux mondes : Java : MDA/UML && Microsoft : DSL/Software Factory

Pourquoi cette solution est bonne

61

- Rien ne permet d'affirmer à 100% que cette solution est la bonne !
- La pérennité des modèles est meilleure que la pérennité du code
  - la réutilisation du code ne marche pas en terme de coût
- Les techniques de génération de code commencent à être productives
  - parallèle avec C et assembleur
- Les techniques de modélisation deviennent industrielles
  - De nouvelles spécifications sont à attendre, suite aux projets Européen ModelWare[1] ou MODA-TEL[2] par exemple, autour de CIM ou de QVT, ….
- Une évolution des modes de développement est indispensable
  - Automatisation et d'optimisation de notre cycle de développement.

[1] http://www.modelware-ist.org/
[2] http://www.modatel.org/

62

Attention, cependant…

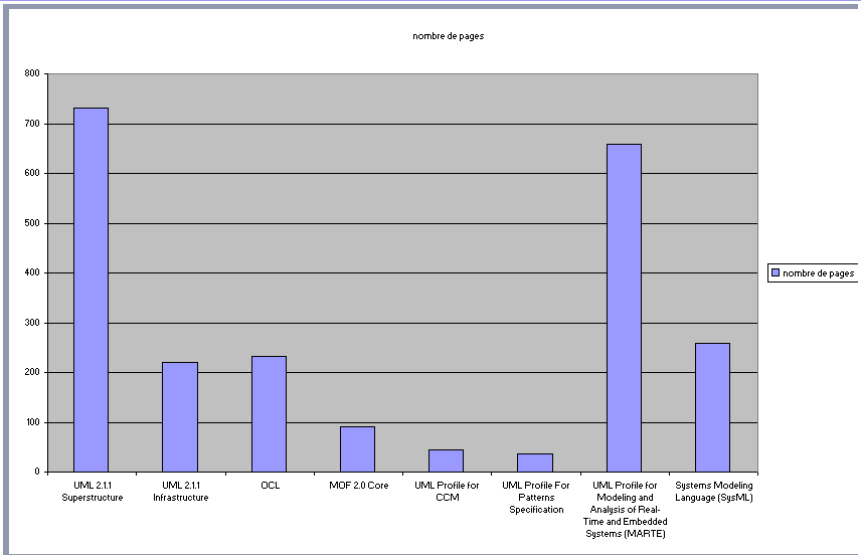
- Mais :
  - Complexité et « chants des sirènes » des Software Factories.
  - Personnalisation, gestion des modèles : coûteuse et difficile : peu de personnel formé à la modélisation, à la gestion des exigences, de la traçabilité et du changement.
  - La maîtrise des outils ne suffit pas. Les équipes projets doivent maîtriser de nombreux concepts, langages (UML, OCL, langage template, CIM, PIM, PDM, …)
  - Les expériences sont encore peu fréquentes (forte expansion dans les domaines critiques: avionique, automobile, ..), les solutions les plus complètes souvent propriétaires néanmoins la démarche est prometteuse.

- **Attention ne confondez pas le rôle du „métamodeleur“ de celui de l‘utilisateur.**

www.**sqli**.com/ressources/files/IBCom_**mai2006**_MDE**MDA**_ecourte.doc

63

Complexité des standards



Références
Voir sur le site web du module pour les plus récentes

64

- Site web de PlaneteMDE: http://www.planetmde.org/
- Site web de l'action IDM : http://www.actionidm.org/
- [IDM 06]FAVRE J.M.,ESTABLIER J., BLAY-FORNARINO M., Eds., Au delà du MDA : l'Ingénierie Dirigée par les Modèles, Hermès, 2006.
- http://web.univ-pau.fr/~ecariou/cours/idm.html
- Krzysztof Czarnecki and Simon Helsen, Classification of model transformation approaches,OOPSLA 2003 Workshop on Generative Techniques in the context of Model Driven Architecture, oct 2003.
- Domain-Specific Languages Arie van Deursen - Paul Klint - Joost Visser CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands, http://www.cwi.nl/ arie,paulk,jvisser/
- Cours UML en ligne Xavier Blanc
- Domain-Specific Languages - An Overview , IRISA/INRIA , en ligne
- Proposition d'action de recherche coopérative (ARC)
  SAM : Syntaxes Abstraites et Modèles ,
  http://www-sop.inria.fr/oasis/personnel/Didier.Parigot/SAM/
- Jean Bézivin, On the unification power of models, Journées Académiques Microsoft Research / 19-21 avril 2004, Chantilly, http://www.sciences.univ-nantes.fr/lina/atl/
- http://www.omg.org/docs/omg/03-06-01.pdf

Références

65

- Introduction and overview of MDA:
  - Model Driven Architecture: An introduction, R.M. Soley
  - Model Driven Architecture, Richard M.Soley (OMG CEO), November 2000
  - Exposé Model Driven Architecture, Fred Waskiewicz, OMG
- MDA documentation:
  - http://www.omg.org/mda
- http://sciences.univ-nantes.fr/info/perso/permanents/bezivin/UML2003
- Contracts, Patterns and Aspects within MDA, ***Prof. Jean-Marc Jézéquel,*** http://www.irisa.fr/prive/jezequel

66

Autres exemples de transformation de modèles & MDE

Atelier IHMs
Atelier Génie Logiciel
Intégration de Services

67

LES PROJETS

http://anubis.polytech.unice.fr/cours/2008_2009:si5:idm:start