De MDA à MDE
Modélisation par contextes fonctionnels

Olivier Caron

Laboratoire d'Informatique Fondamentale de Lille,
Université des Sciences et Technologies de Lille,
59655 Villeneuve d'Ascq cedex - France

http://www.lifl.fr/GOAL/cocoa

Plan

Merci à....

- L'équipe COCOA de GOAL
- Les transparents d'Alexis Muller

Ingénierie des modèles

- Pourquoi des modèles ?
  - Certes, MDA : PIM, PSM, transformations,...
  - Mais à l'origine, les modèles servent :
    - à offrir une représentation d'un système avec le minimum d'ambiguïtés
    - à construire un système
    - à exprimer simplement et rapidement un système (en tout cas, plus simple que la cible PSM)

Complexité des systèmes

- La construction de systèmes d'information reste une tâche difficile
- Doivent répondre à un grand nombre de préoccupations :
  - métiers :marketing, fabrication, ventes,...
  - non fonctionnelles :sécurité, persistance, répartition,...
- Taille des systèmes de plus en plus importante
- Délais de conception toujours plus courts
- Technologies non pérennes

Structuration

- La structuration doit permettre une meilleure maîtrise de cette complexité
- Deux axes de structuration peuvent être utilisés :
  - Selon les fonctions : structuration horizontale
  - Selon des niveaux d'abstraction : structuration verticale
- Différents paradigmes pour la structuration horizontale : préoccupations, sujets, aspects, vues,...
- Structuration verticale mise en avant par l'approche MDA

Structuration

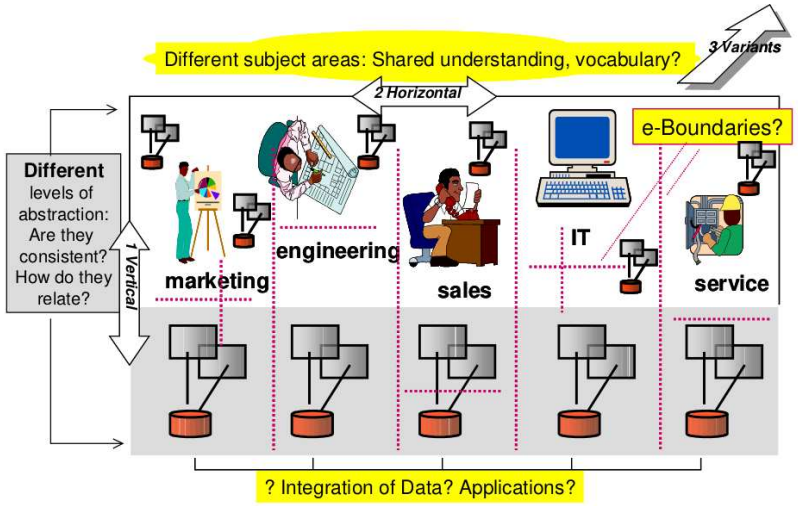


FIG.: Dimensions de structuration d'un système [MDAI]

Réutilisation

- La réutilisation doit permettre de faire face aux délais de conception toujours plus courts
- Peut être utilisée à différents niveaux d'abstraction
- Peut être de différentes granularités :
  - par l'utilisation d'objets
  - par l'utilisation de frameworks
- L'utilisation de composants complexes doit permettre un meilleur rendement
- Mais leur intégration en est encore plus complexe

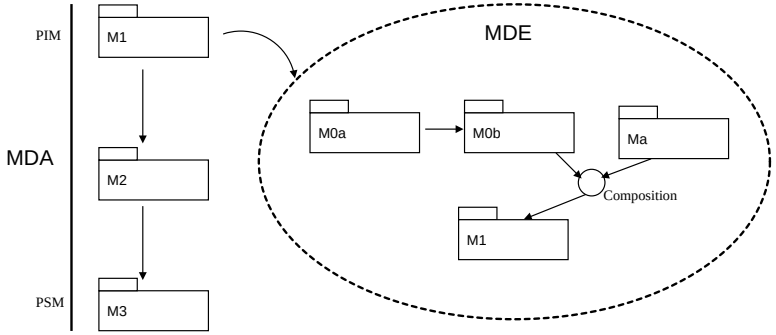
De l'architecture dirigée par les modèles (MDA)

- Placer les modèles au cœur des processus
- Séparation des spécifications fonctionnelles des spécifications techniques
- Modèles métiers : PIM
- Modèles spécifiques : PSM
- Basée sur les langages standards : UML, MOF, QVT
- Idée séduisante mais . . .
- Vers l'ingénierie dirigée par les modèles

À l'ingénierie dirigée par les modèles (MDE)[Ken02]

- Généralise l'idée du MDA
- Propose de structurer l'espace des modèles dans toutes ses dimensions
- Problème de la mise en relation des différents modèles
- Vise à faire des modèles des artefacts manipulables

MDA et MDE

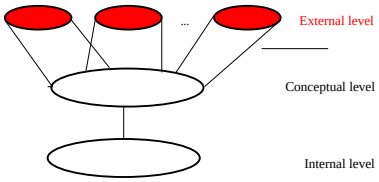


Verrous scientifiques

- Comment faire des modèles des composants réutilisables ?
- Comment les rendre génériques ?
- Comment exprimer et valider la composition de modèles ?
- Peut-on composer les modèles génériques entre eux ?
- La structuration introduite au niveau conception peut-elle être conservée à l'exploitation ?
- Comment faciliter la prise en compte des besoins architecturaux ?

Origine des Vues

- Créé dans le cadre des bases de données.
- Architecture normalisée ANSI/SPARC :



Vues et bases de données

- Indépendance logique des données.
- La vue n'est pas enregistrée physiquement.
- A l'exécution d'une requête portant sur une vue, une opération de reconstitution de la vue est effectuée.
- Tables virtuelles issues de une ou plusieurs tables :
  - Modification du schéma et/ou
  - Modification des instances participantes à la vue.
- Une vue est considérée comme une table pour les utilisateurs.
- Mise à jour des vues difficile et/ou impossible.

Concepts de base des vues

- Notion de schéma-vue : ensemble cohérent et structuré d'entités qui répondent à une fonction du sytème.
- Notion d'entité conceptuelle : entité de base et vues forment une même identité (cycle de vie, . . .)

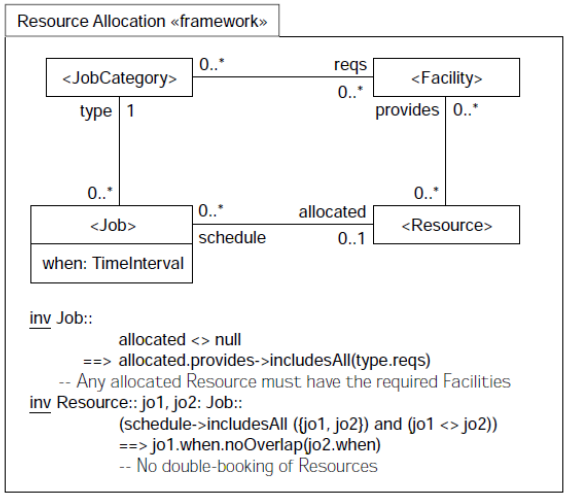
Qualités des Vues

- Construction itérative, fonction par fonction.
- Aboutit à des schémas plus simples et lisibles.
- Favorise l'évolution du système.

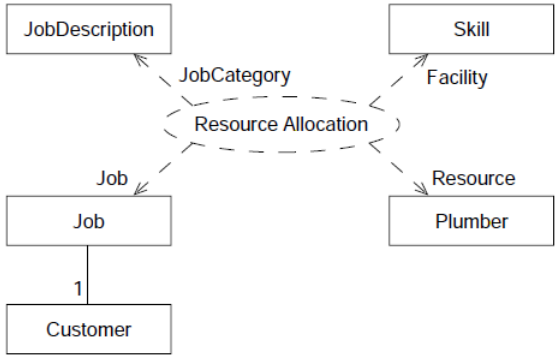
Vues et UML

- La notion de vue existe dans UML mais n'a pas la même signification.
- Les Vues UML (classes, séquences, états, . . .) :
  - Offre une représentation multiple
  - Ne permet pas de gérer l'identité conceptuelle
- Même constat pour le concept d'UML d'interface.

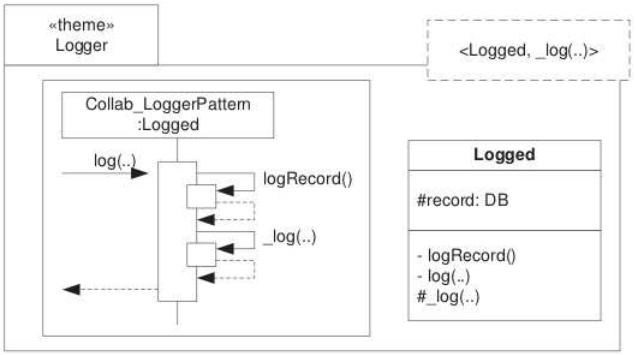
Catalysis[DW99] (1/2)



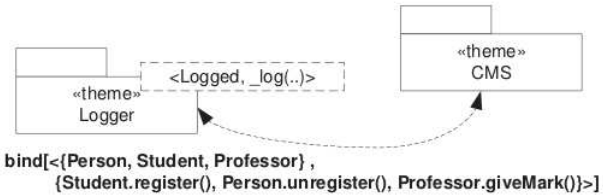
Catalysis[DW99] (2/2)



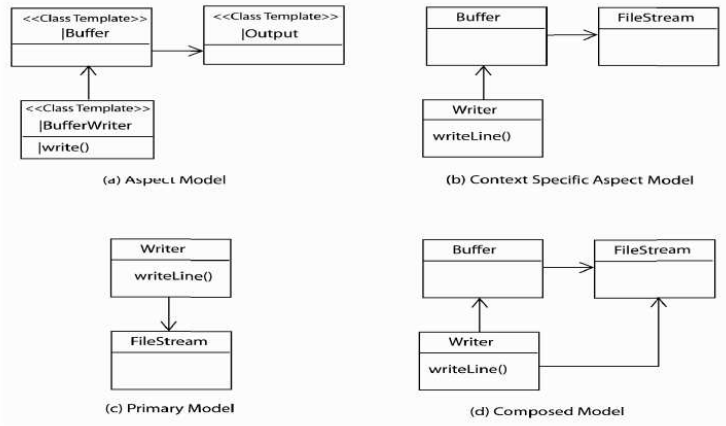
SOD/theme[BC04] (1/2)



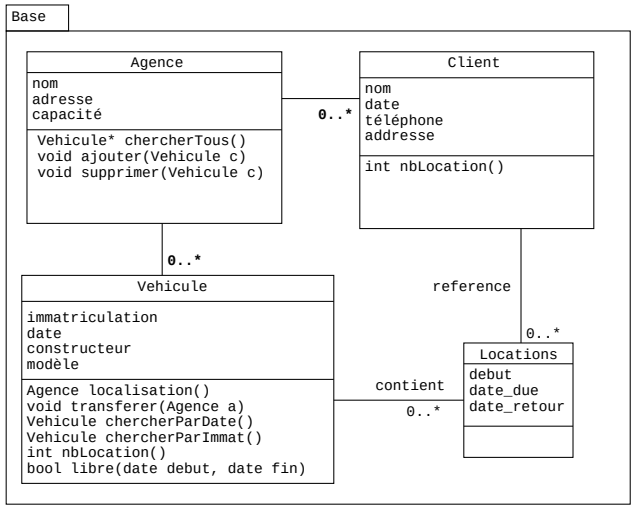
SOD/theme[BC04] (2/2)



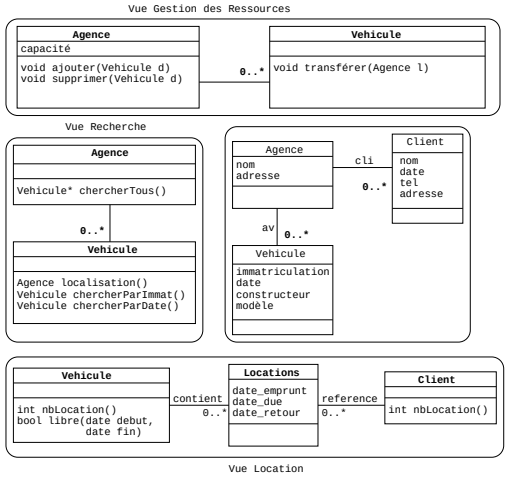
AOM (R. France[SGS04])



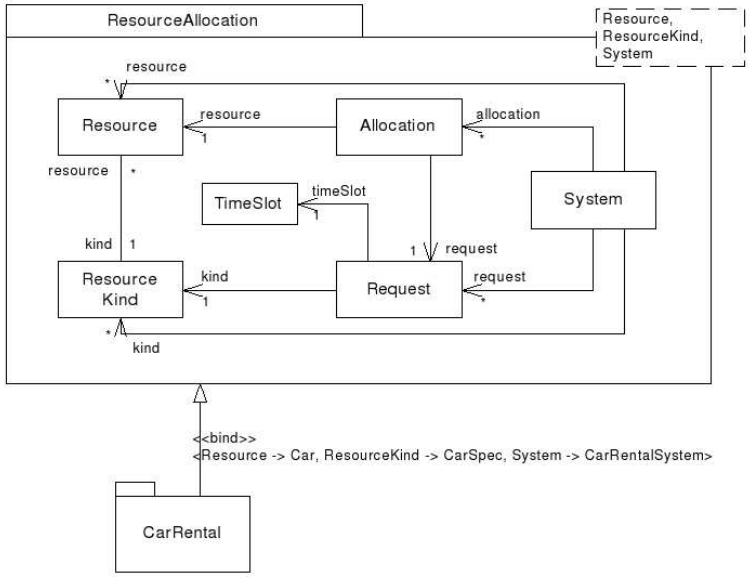
CROME (1/2)



CROME[Van99] (2/2)



Qu'en est-il des standards ? [UML05b]



Bilan (1/2)

	Séparation des préoccupations	Artefacts manipulables	Granularité des paramètres
Catalysis Framewoks	Paquetages	non, doivent être adaptés	éléments unitaires
Theme/UML	Paquetages	oui, méta-modèle	classes
R. France	Modèles	oui, méta-modèle	éléments unitaires
CROME	Vues	non	N/A
Templates UML	N/A	oui, méta-modèle	éléments unitaires

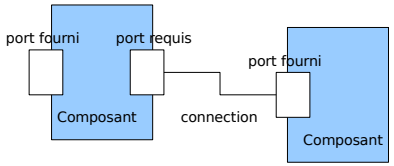
Bilan (2/2)

	Modes de composition	Validation de la composition
Catalysis Framewoks	construction d'un framework à partir d'un autre	vérification a posteriori
Theme/UML	tissage	vérification au niveau analyse
R. France	fusion et directives	vérification a posteriori
CROME	N/A	N/A
Templates UML	N/A	N/A

Vue d'ensemble

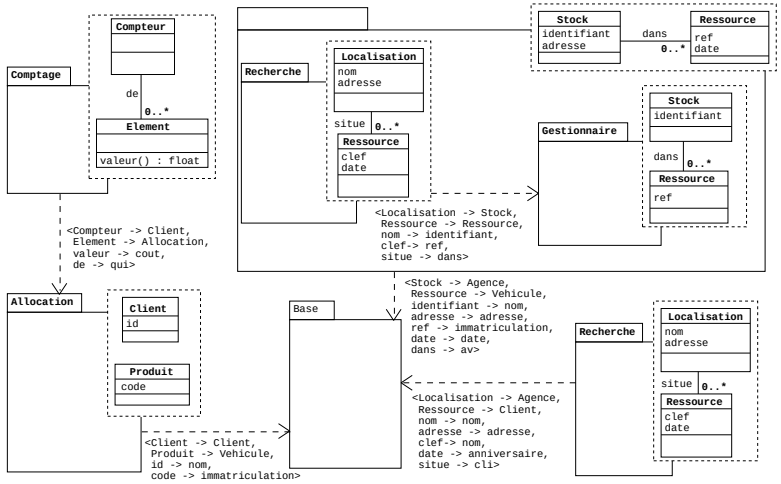
- Vers des *composants de modèles*
- Exprimer une partie requise complexe sous forme d'un modèle de système
- Un opérateur d'application pour l'expression d'assemblages complexes
- Permettre l'expression de modèles génériques manipulables
- Rapprochement avec les standards

Le paradigme composant

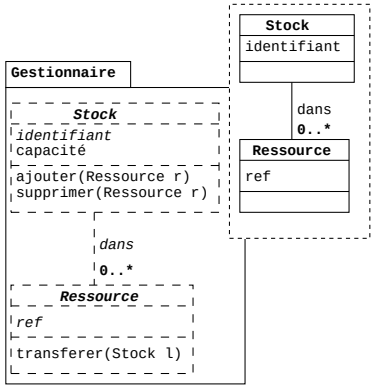


	composant logiciel	composant de modèle
composant	module (jar, . . .)	modèle
Port	interface	modèle

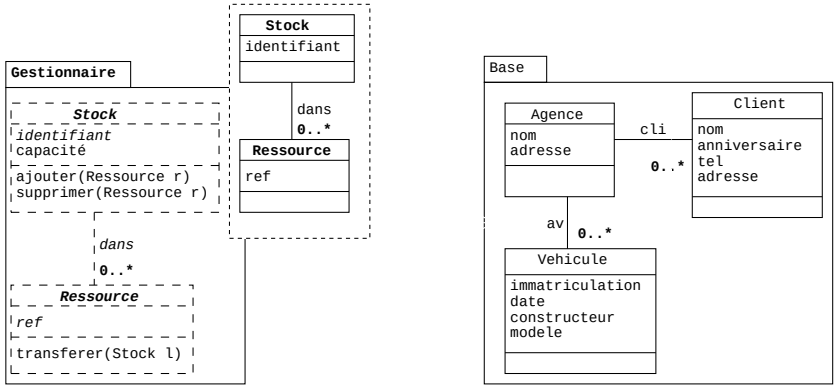
- Vérification de l'assemblage beaucoup plus complexe !



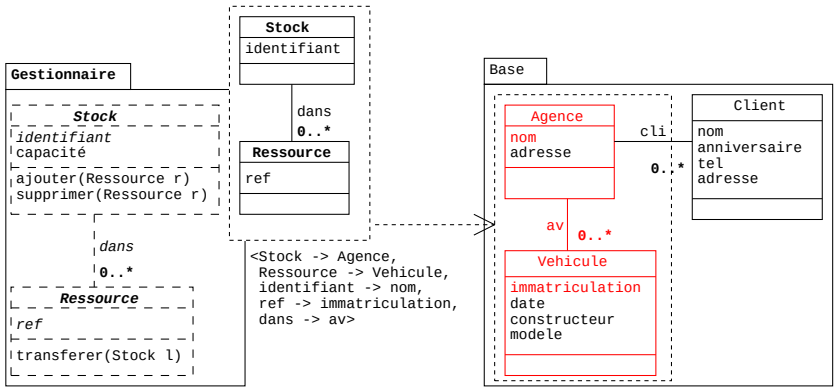
Modèle paramètre



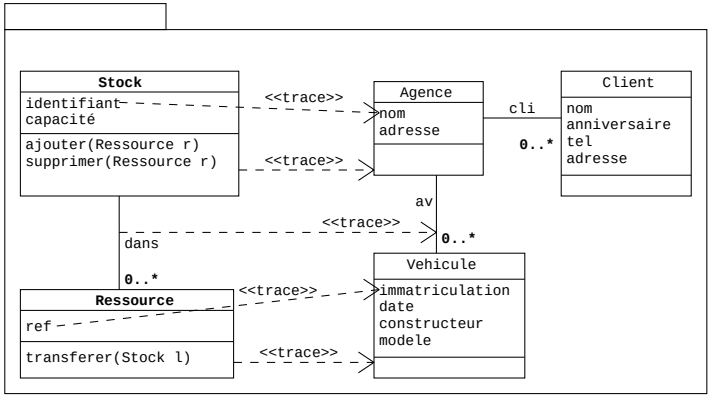
Application d'une fonctionnalité à un système (1/2)



Application d'une fonctionnalité à un système (2/2)



Expression du résultat



Problèmes d'ordre

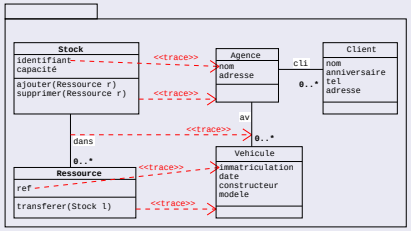
- La conception d'un système nécessite l'utilisation d'un ensemble de composants de modèles
- Les composants de modèles doivent pouvoir être appliqués de façon cohérente
- Le résultat doit être indépendant de l'ordre d'évaluation des applications
- Est-il possible de trouver des séquences d'applications équivalentes ?

Définitions [MMCV05]

Définition 1

Un modèle A est défini par un triplet (E_A, P_A, V_A). E_A est un ensemble d'éléments de modèle. $P_A \subset E_A$ est un ensemble de paramètres. V_A est un ensemble de relations de correspondance définies dans ($E_A \times E_A$).

Exemple



Définitions [MMCV05]

Définition 2

Deux modèles sont égaux si et seulement si, ils contiennent le même ensemble d'éléments, ils ont le même ensemble de paramètres et ils ont le même ensemble de relations de correspondance.

Définitions [MMCV05]

Definition 3

On note $R = B \underset{s}{\rightarrow} A$ l'application d'un modèle paramétré B à un modèle A conformément à l'ensemble de correspondances s.
On note FP_s l'ensemble des paramètres formels et EP_s l'ensemble des paramètres effectifs de s.
Le modèle résultat R est construit conformément aux règles suivantes :

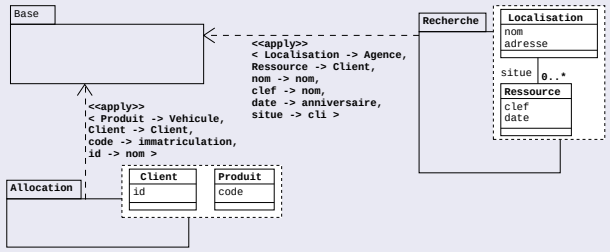
$$R = B \underset{s}{\rightarrow} A \Rightarrow R = \begin{cases} V_R = V_B \cup s \cup V_A \\ E_R = E_B \cup E_A \\ P_R = (P_B \setminus FP_s) \cup P_A \end{cases}$$

Propriétés [MMCV05]

Propriété d'ordre

L'ordre d'application de deux modèles à un troisième n'influence pas le résultat.
Soit deux ensembles de correspondances s, s' tel que $EP_s \subseteq E_A$ et $EP_{s'} \subseteq E_A$. Alors $B \underset{s}{\rightarrow} (C \underset{s'}{\rightarrow} A) = C \underset{s'}{\rightarrow} (B \underset{s}{\rightarrow} A)$

Exemple



Démonstration

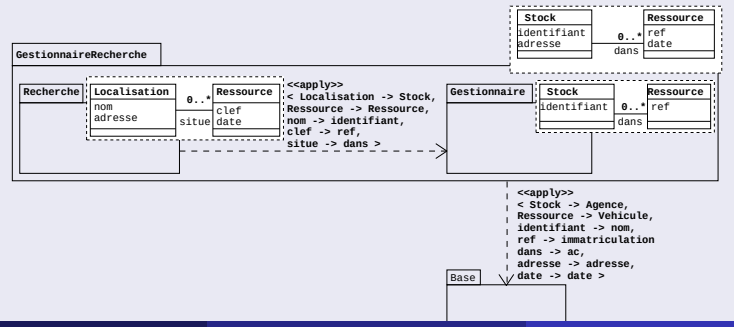
Dans la these.

Propriétés [MMCV05]

Séquences d'applications équivalentes

Pour toute séquence $B \underset{s_1}{\rightarrow} (C \underset{s'_1}{\rightarrow} A)$, il existe une séquence $(B \underset{s_2}{\rightarrow} C) \underset{s'_2}{\rightarrow} A$, qui produit le même résultat, tel que
$s_2 = s_1 \setminus ((E_A \times \mathscr{E}) \cap s_1)$ et $s'_2 = s'_1 \cup ((E_A \times \mathscr{E}) \cap s_1)$

Exemple

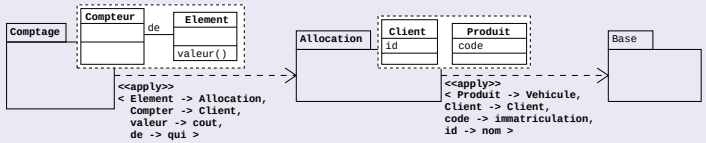


Propriétés [MMCV05]

Propriété d'ordre des chaînes d'applications

Soit $B \underset{s}{\longrightarrow} C \underset{s'}{\longrightarrow} A$ une chaîne d'application telle que $EP_s \subseteq E_C$, elle peut être évaluée aussi bien comme $B \underset{s}{\longrightarrow} (C \underset{s'}{\longrightarrow} A)$, que comme $(B \underset{s}{\longrightarrow} C) \underset{s'}{\longrightarrow} A$.

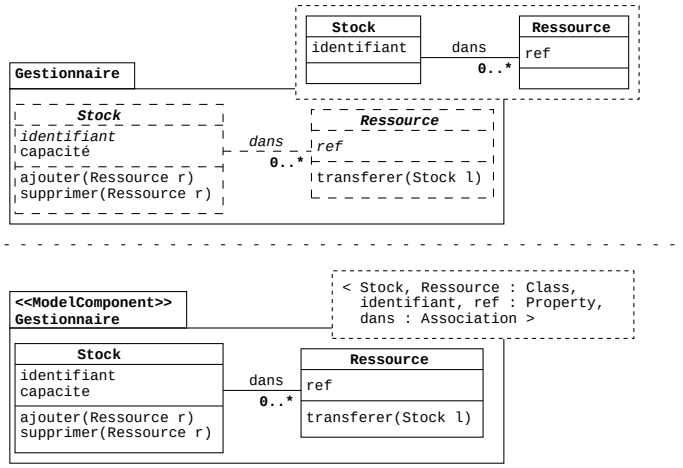
Exemple



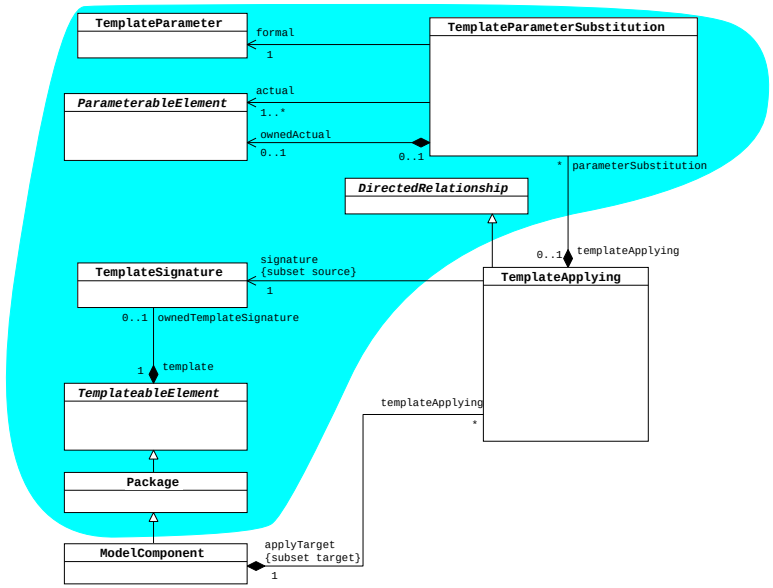
Bénéfices

- Permet de garantir l'unicité du résultat quelque soit l'ordre d'évaluation des applications
- Autorise l'évolution incrémentale des systèmes
- Permet la construction de fonctionnalités complexes à partir de fonctionnalité plus simples
- N'impose pas une forme particulière du système produit

Paquetage template [CCMV04]



Méta-modèle



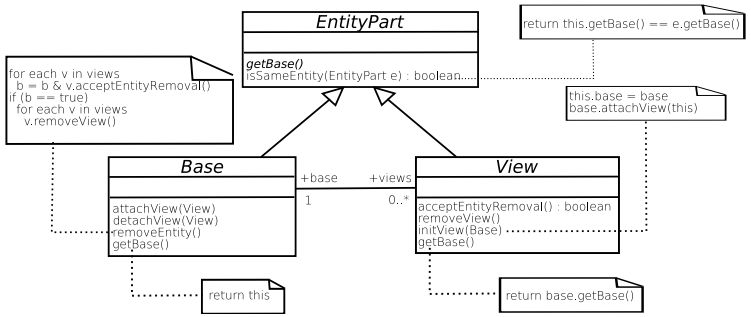
Niveau architectural

- Possibilité d'exprimer le résultat selon différents modes, notamment à l'aide de vues
- Avantages des vues : expression claire de la double structuration entités/fonction, traçabilité des fonctionnalités
- Prise en compte des besoins architecturaux
- Problème de la préservation des vues jusqu'à l'exploitation

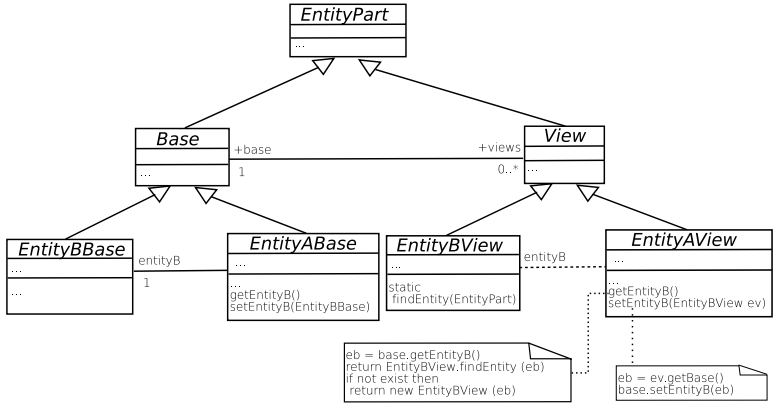
Architecture à base de patrons

- Permettre la mise en œuvre éclatée d'entités [CCMV03]
- Permettre l'adaptation de fonctionnalités génériques (réutilisation de composants binaires) [CCMV05]
- Permettre la gestion automatique des fragments [CCMV05]

Patron de représentation éclatée



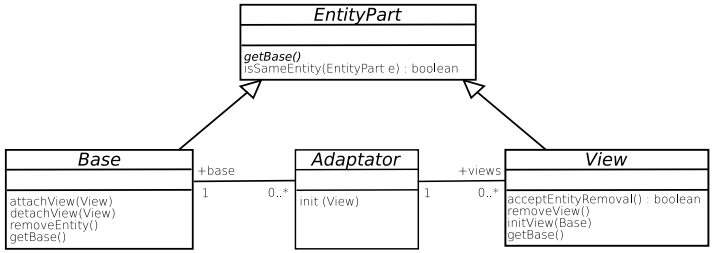
Gestion des vues associations



Architecture à base de patrons

- Permettre la mise en œuvre éclatée d'entités [CCMV03]
- Permettre l'adaptation de fonctionnalités génériques (réutilisation de composants binaires) [CCMV05]
- Permettre la gestion automatique des fragments [CCMV05]

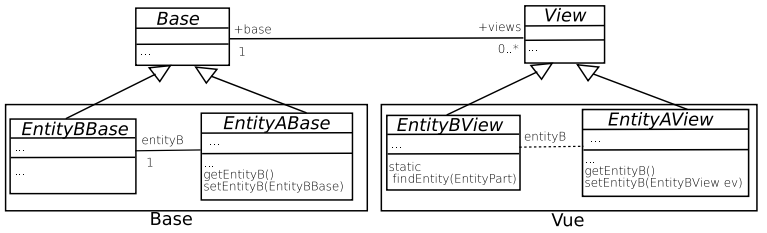
Introduction du patron Adaptateur



Architecture à base de patrons

- Permettre la mise en œuvre éclatée d'entités [CCMV03]
- Permettre l'adaptation de fonctionnalités génériques (réutilisation de composants binaires) [CCMV05]
- Permettre la gestion automatique des fragments [CCMV05]

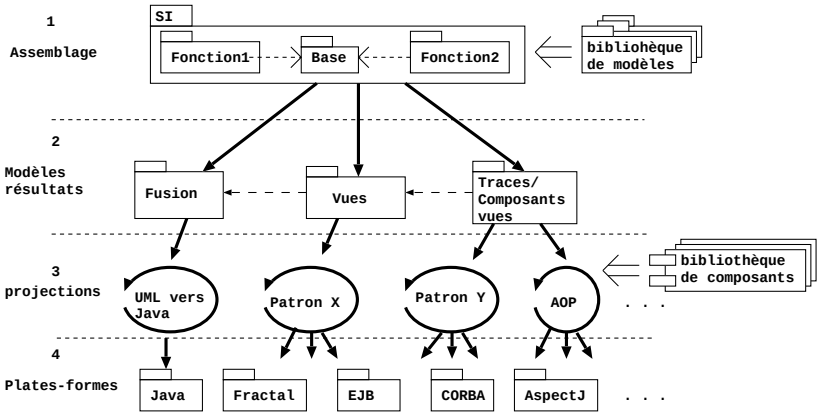
Gestion des vues



Réalisations

- Architecture expérimentée sur différentes plates-formes :
  - EJB [CCMV03]
  - Fractal [BMP05]
  - CORBA
- Un atelier de composition de modèles :
  - permet la conception de composants de modèles
  - permet la réalisation de modèles d'assemblage
  - permet le calcul des modèles fusionnés
  - permet la projection vers la plate-forme CORBA

Chaîne de production



Conclusion

- Expression à un niveau modèle de fonctionnalités génériques
- Expression de parties requises complexes sous forme de modèles
- Bonnes propriétés d'ordre
- Indépendance de la forme du système à produire
- Définition stricte à l'aide d'un méta-modèle
- Possibilité de préserver la structuration du système jusqu'à son exploitation
- Autorise une démarche à base de transformations paramétrées [BCGM04]

Perspectives

- Intégration dans les processus et ateliers de conception
  - Opérationalisation à l'aide de langages de transformation
  - Gestion de bibliothèques de composants
- Notion de conformité de modèles [SJ05]
  - Prise en compte de l'héritage
  - Des cardinalités
- Composition de modèles dynamiques [BC04]
  - Composition de diagrammes d'états ou de séquences
  - Qu'en est-il des propriétés d'ordre ?