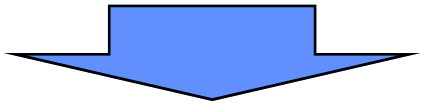
Démarche de développement OMT-UML/HOOD

Agusti CANALS
Cisi 13, rue Villet, 31400 Toulouse, France
Tél +33 (0)5 61 17 66 66, Fax +33 (0)5 61 54 13 39
E-mail : canalsa@toulouse.cisi.fr

Jean-Christophe LLORET
CNES, 18 avenue Edouard Belin, 31401 Toulouse Cedex
Tél. : +33 (0)5 61 28 29 61, Fax +33 (0)5 61 27 31 79
E-mail : lloret@cnes.fr

Introduction

- Situation actuelle:
  - OMT en analyse et conception
  - HOOD 3 en conception
- Situation à terme
  - OMT/UML
  - HOOD 4



L'étude de la transition OMT-UML/ HOOD 4 est une priorité

Sommaire

- Concepts OMT en HOOD,
- Traduction des associations OMT en HOOD,
- Patterns OMT en HOOD,
  - Singleton, Façade et Commande
- Processus de conception HOOD à partir d'une analyse OMT,
- Les critères de choix d'OMT/HOOD en conception,
- Conclusion

Concepts OMT en HOOD

OMT	HOOD
Classe, attribut, opération	Idem
Classe d'instance unique (singleton)	Objet HOOD ou Classe (si besoin de spécialisation): voir discussion sur le pattern singleton.
Sous-système	Objet HOOD père.
Classe dominante d'un sous-système	Définit l'interface exportée de l'objet père correspondant; nécessite éventuellement l'introduction d'un objet "op_control" où la reprise de la classe façade en HOOD (voir discussion sur le pattern façade).
Agrégation	Relation d'attribution (la classe composite a un attribut typé par la classe composant).
Association	Relation d'attribution éventuellement bidirectionnelle; le type des attributs dépend de l'arité de l'association (voir discussion ci-dessous).
Héritage	Idem
Diagramme de classes d'un sous-système	Vue structure de l'objet père HOOD correspondant (relations d'attribution et d'héritage).
Diagramme de sous-système	Vue structure + vue client/serveur de l'objet père HOOD (voir discussion sur le pattern façade).
Diagramme de flot d'événements entre classes d'un sous-système.	Vue client/serveur de l'objet père HOOD correspondant.
Scénario OMT d'un sous système	Entrée pour la stratégie informelle de la solution de l'objet père correspondant.
Diagramme d'état (statechart)	Comportement (Object State Transition Diagram: sous-ensemble des concepts des statecharts supportés).
Diagramme d'instance	Néant
Module (vue)	Pas de correspondance en général; 2 vues disponibles et obligatoires : vues structure et client/serveur.

Traduction des associations OMT en HOOD

Comment traduire en HOOD une association OMT qui reste pertinente pour la conception ?

- Le concept d'association n'existe pas en HOOD
- Pour le traduire il faut faire des choix de conception:
  - Choix d'orientation (sens de la relation d'attribution),
  - Choix d'implémentation des associations n-aires (choix d'un conteneur: liste ...)
- Implémentation des associations:
  - OMT: Règles de génération de code
  - HOOD: Formalisation dans la conception

Traduction des associations OMT en HOOD

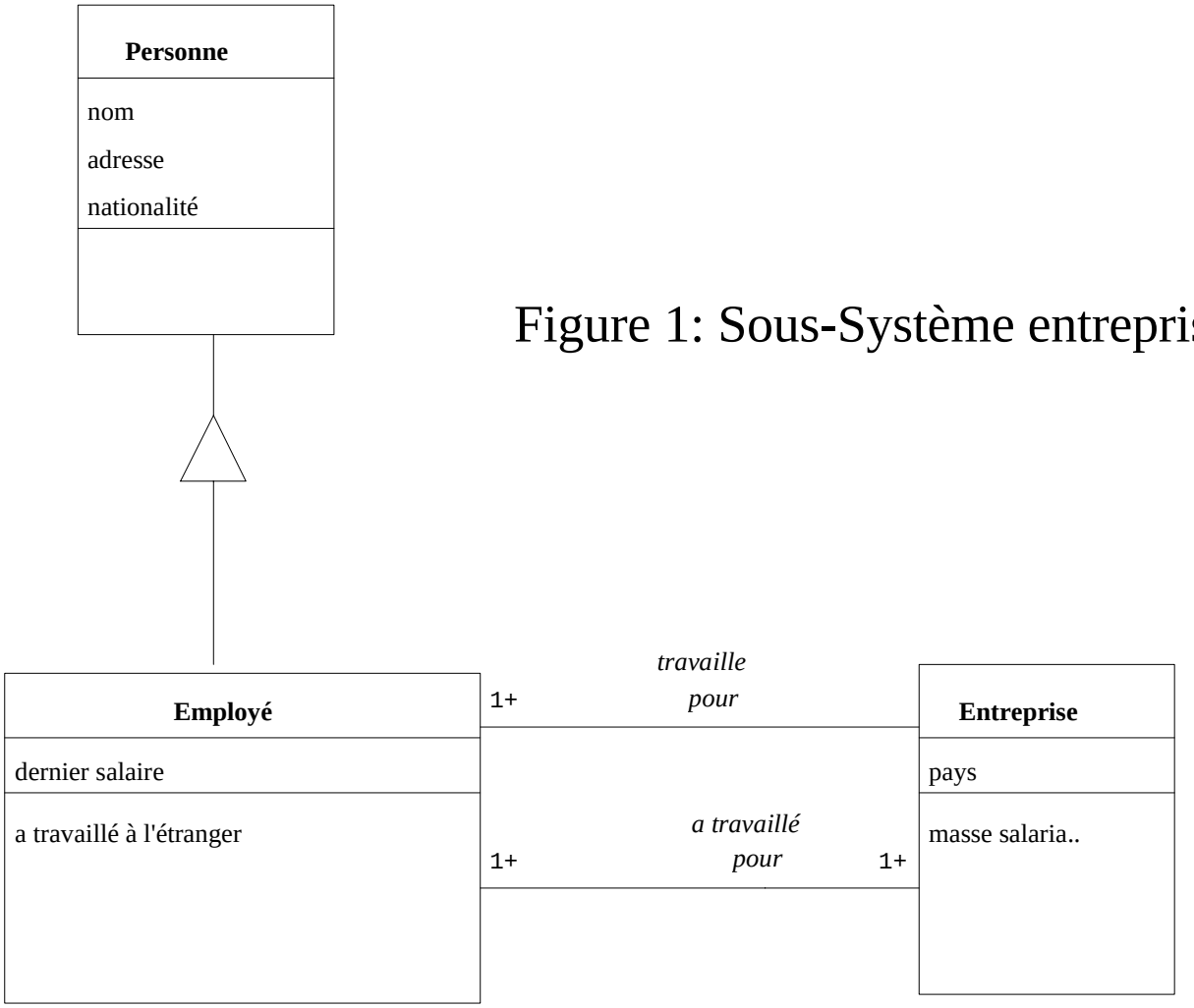


Figure 1: Sous-Système entreprises

Traduction des associations OMT en HOOD

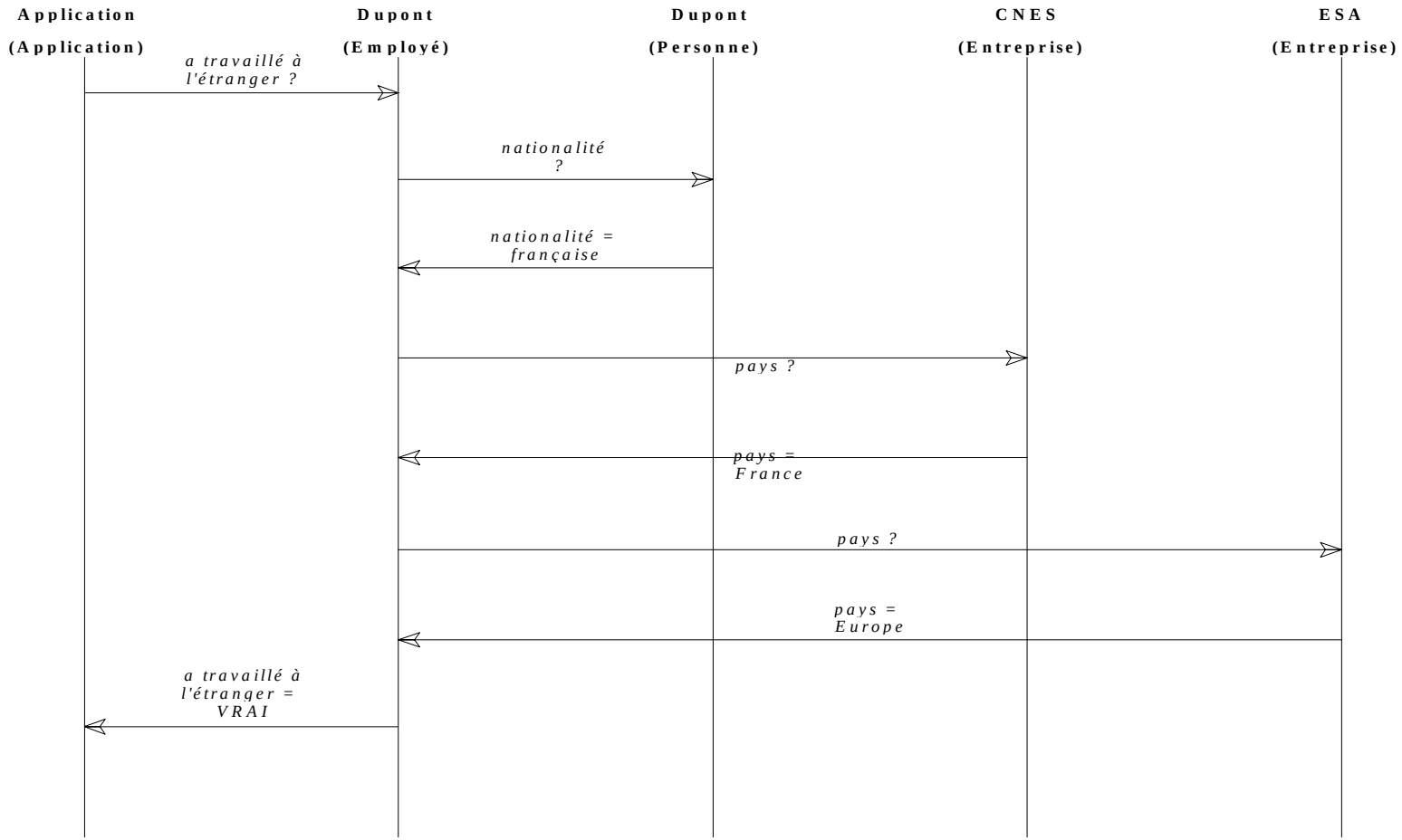


Figure 2: Scénario de fonctionnement

Traduction des associations OMT en HOOD

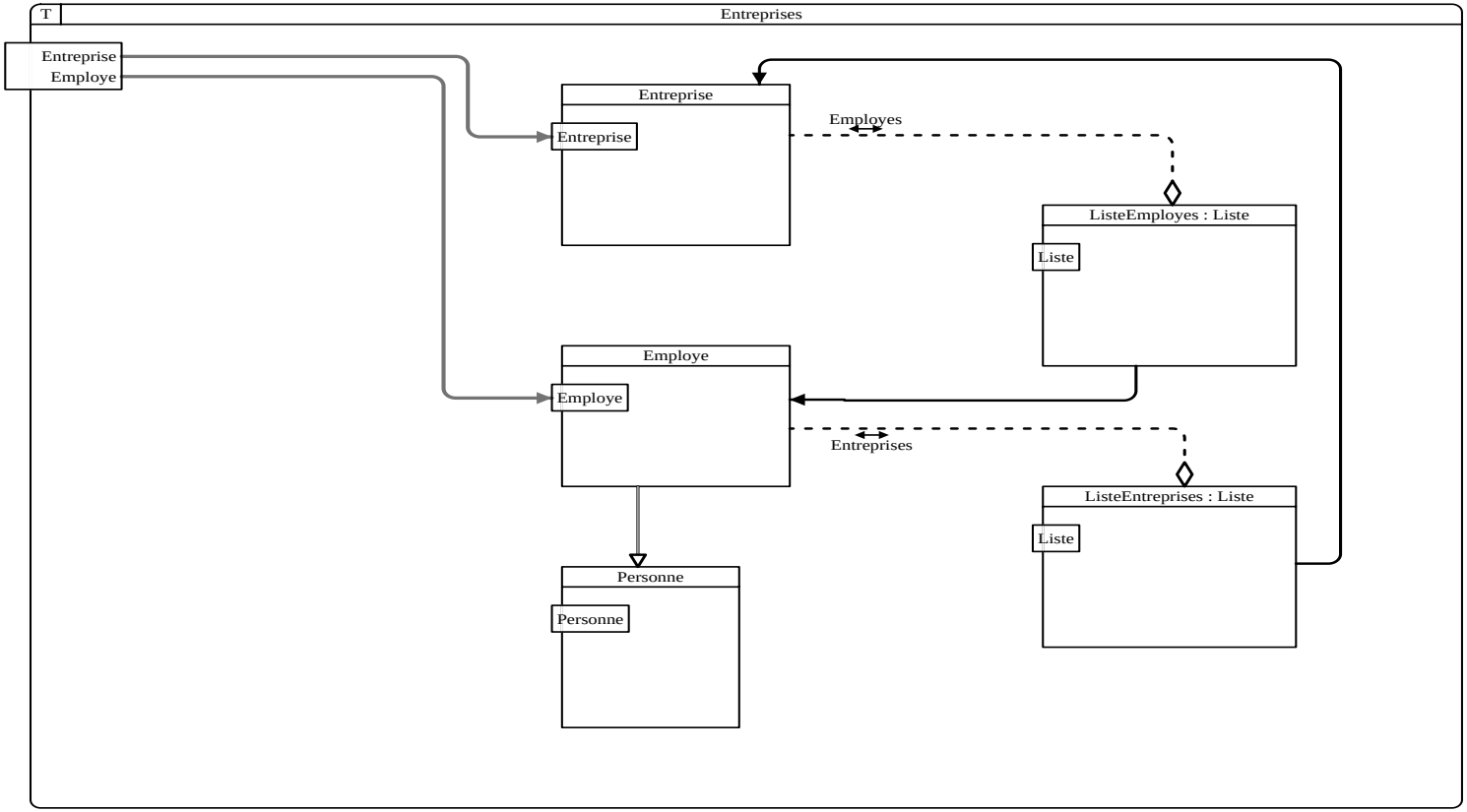


Figure 3: Vue Structure de l'objet Entreprises

Traduction des associations OMT en HOOD

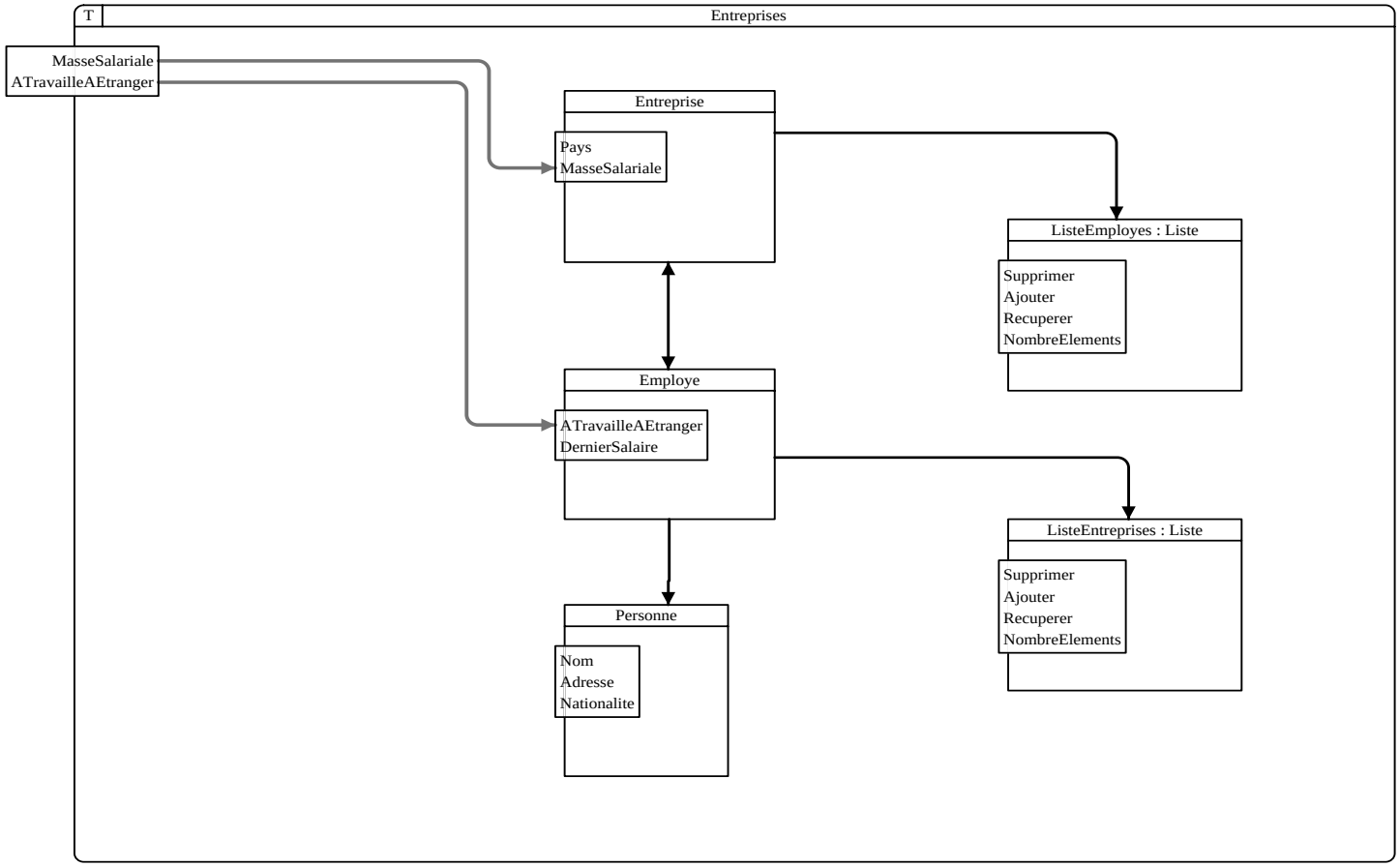


Figure 4: Vue Client/Serveur de l'objet Entreprises

Patterns OMT en HOOD

- Les «patterns» sont des schémas donnant une solution type à un problème de conception.

- Intêret pour la transition OMT/HOOD
  - des patterns sont utilisés aussi en analyse
  - certains patterns sont utiles pour formaliser le passage à la conception
  - les patterns peuvent être utilisés en conception HOOD (ils sont indépendants de la notation)

Patterns OMT en HOOD

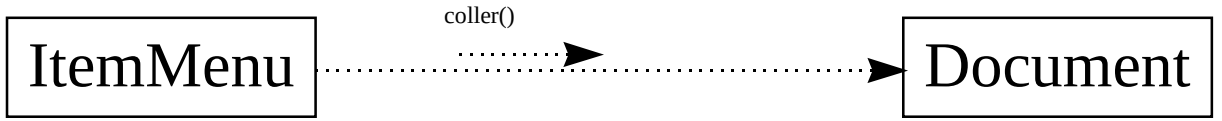
- Patterns étudiés:
  - Singleton
    - Pattern de création pour garantir l'unicité de l'instanciation d'une classe.
    - Pour traduire en HOOD des classes OMT à instance unique.
  - Façade
    - Pattern structurel pour donner un point d'accès unique à un Sous-Système.
    - Pour traduire en HOOD un sous système OMT.
  - Commande
    - Pattern de comportement.
    - Intêret en HOOD pour raffiner la conception par analyse des flots de contrôle.

Patterns OMT en HOOD

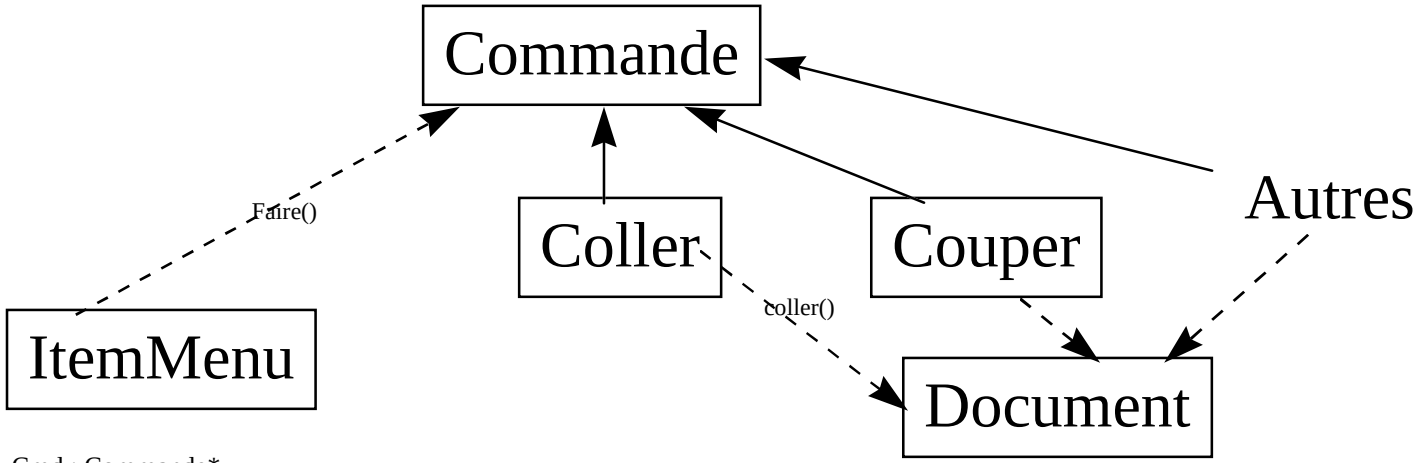
- Objectif du pattern Commande
  - Assurer l'indépendence entre une classe client et une classe serveur.
- Description
  - Hierarchie d'héritage de racine la classe abstraite «*Commande*» avec les méthodes (faire, défaire ...)
- Application en HOOD
  - Répartition des classes du Pattern dans la conception
    - Dans l'environnement: *Commande*, MacroCommande
    - Dans les sous-systèmes: Instances concrètes de *Commande*

Patterns OMT en HOOD

Modèle initial



Modèle après raffinement



Patterns OMT en HOOD

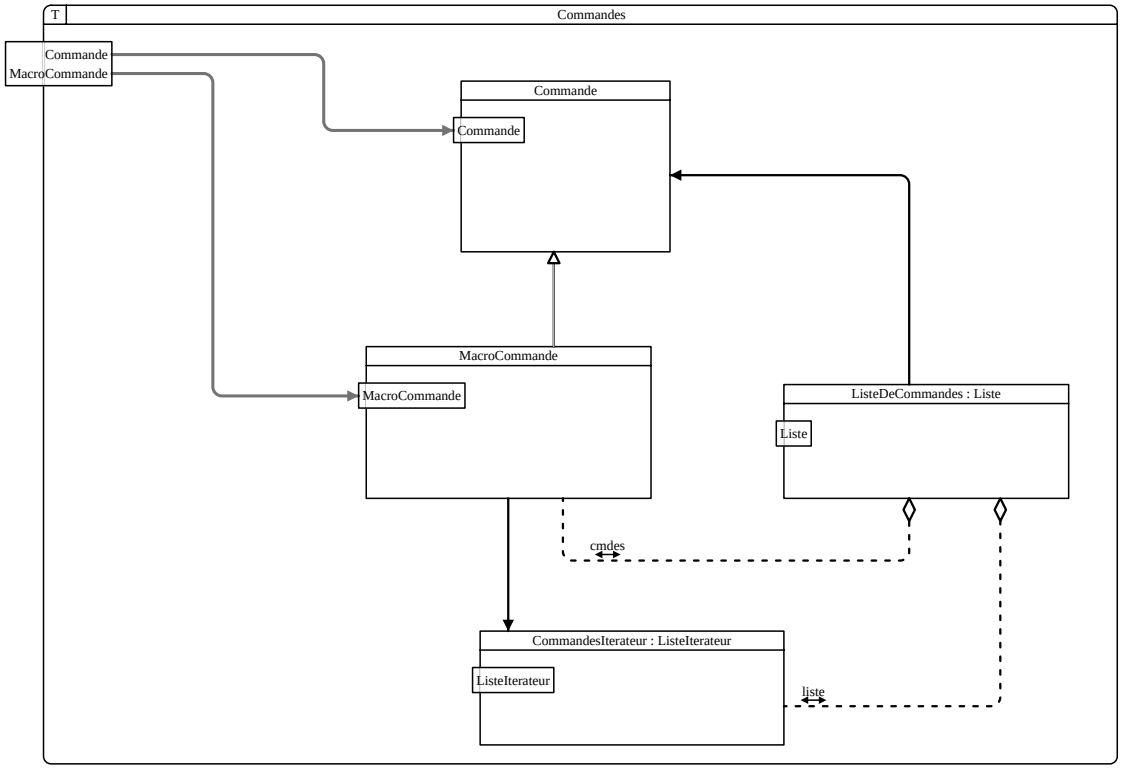


Figure 5: Vue Structure de l'objet Commandes

Patterns OMT en HOOD

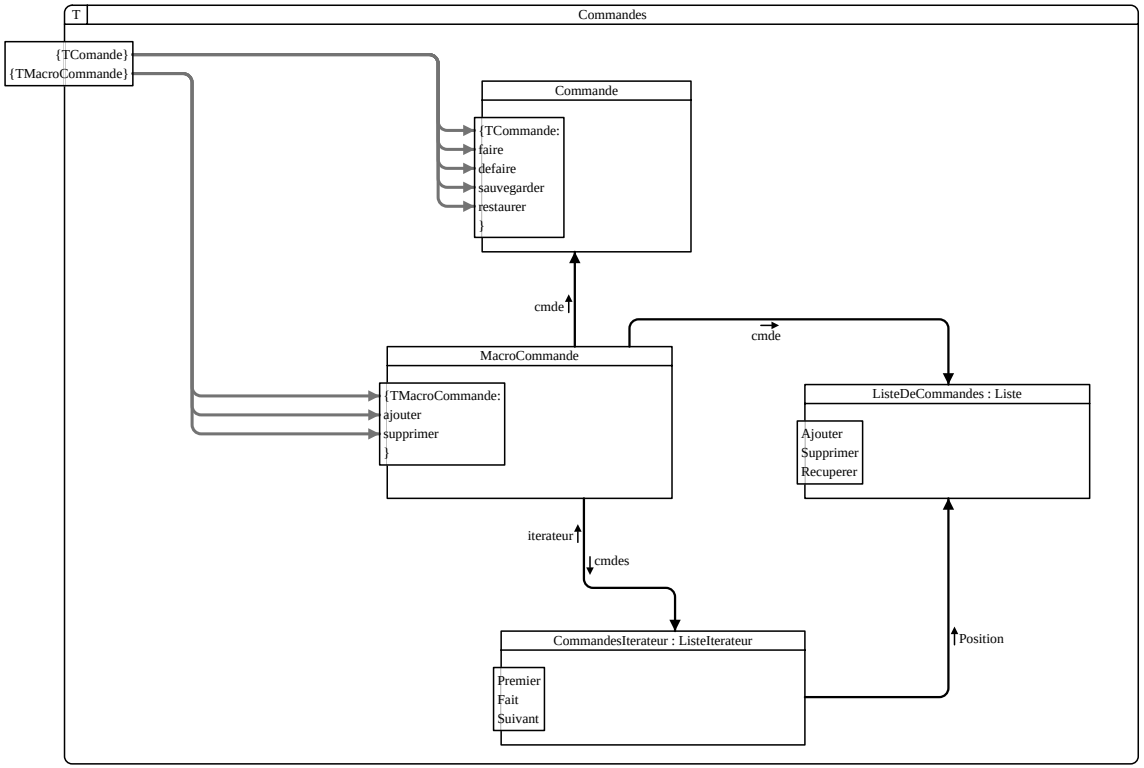


Figure 6: Vue Client/Serveur de l'objet Commandes

Processus de coception HOOD à partir d'une analyse OMT

- La démarche proposée permet:
  - d'utiliser au mieux le matériel issu de l'analyse,
  - d'assurer une continuité entre l'analyse et la conception
- La démarche proposée n'impose pas de solution pré-établie (remise en question du modèle d'analyse pour la conception)
- Point de départ de la conception: Les sous-systèmes OMT (Sous systèmes fonctionnels)
- Introduction de sous-systèmes de conception (services transversaux, IHM,...)

Les Critères de choix d'OMT/HOOD en Conception

- Langage de programmation
- Typologie du projet
  - Temps réel, système d'information ...
  - Solutions techniques utilisées
- Compétence des personnels
- Formalisation de la démarche
  - La notation HOOD impose une démarche
  - En OMT il faut formaliser des règles d'utilisation (cf MPM CNES)
- ...

Conclusion

- Approche sans rupture.
- Transition facilité par une analyse conduite suivant un processus rigoureux (utilisation des sous-systèmes).
- Intêret évident de l'utilisation des patterns en HOOD.
- Points Ouverts
  - Quels outils pour faciliter la transition ?
  - HOOD peut il correspondre à une particularisation de UML ?
  - Etude de la transposition d'autres patterns en HOOD ?