UNIVERSITÉ PARIS DESCARTES

MEMBRE DE

USPC Université Sorbonne Paris Cité

Bases de la conception « orientée objet »

DUT Informatique
Semestre 2

Mourad Ouziri
mourad.ouziri@parisdescartes.fr

IUT de Paris Descartes

Conception objet :

« Modélisation de classes avec UML »

Objectif : savoir analyser une situation réelle simple et identifier de manière pertinente les classes de l'application objet

Conception objet
Qualité

- Rappel sur la qualité
  - **Fonctionnalité** : réponse aux spécifications fonctionnelles
  - **Maintenabilité** : coût de corrections et d'évolutions
  - Performance : ressources requises pour fonctionner
  - Robustesse : capacité à fonctionner en cas d'anomalies (manque de ressources)
  - …
- Phases de développement
  - Phase d'analyse : analyse **fonctionnelle** de l'application
  - Phase de conception : mise en place de la structure (architecture) de l'application (pour une meilleure **maintenabilité** *approfondie en semestre 3*)

Conception objet

Modèle de classes

- DCU
  - Exprime les services fonctionnels que doit offrir le logiciel à ses utilisateurs
  - Ne donne aucune indication du comment les implémenter
- Modèle de classes – définition
  - Exprime la structure des éléments (objets) constituant l'application
  - Exprime les classes (ou types) d'objets de l'application
  - Indique les classes, attributs et (signature des) méthodes constituant le logiciel
  - Structure statique : n'indique pas comment les méthodes sont implémentées

Conception objet
Définitions

☞ Application objet

– Une application objet est une société d'objets (ensemble d'objets communicants)

– Chaque objet réalise des tâches élémentaires qui relèvent de ses compétences (**rôle**)

– Les objets de l'application collaborent (**interagissent**) pour réaliser les fonctions de l'application

Conception objet
Définitions

- Conception objet
  - Structurer/décomposer une situation réelle en objets
  - Modélisation (par des schémas visuels) d'une situation réelle avec des objets
  - Une formule simple, un objet du monde réel est représenté par un objet informatique (résidant dans la mémoire vivre de l'application)
- Abstraction : un élément fondamental dans la conception
  - Faire face à la complexité de la situation réelle
  - Identifier les **propriétés externes** des objets les distinguant les uns des autres
  - Regrouper les objets ayant les mêmes propriétés en classes

Conception objet
Notion d'objet

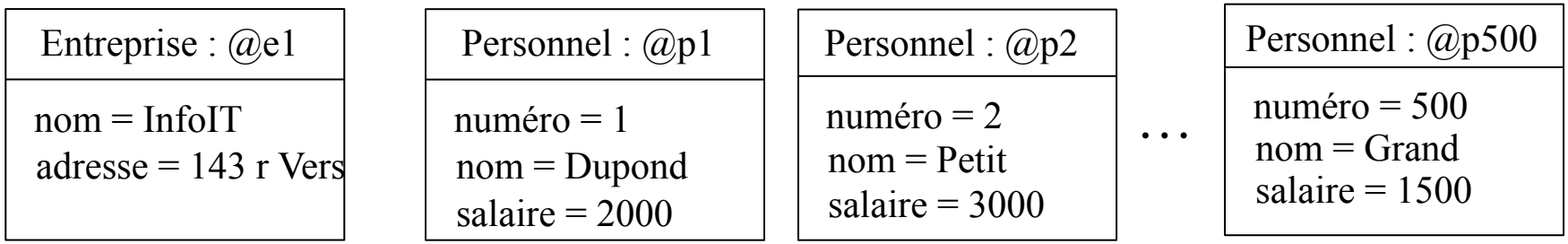
- Objet modélisé
  - Un objet modélisé peut représenter toute entité (matérielle ou immatrérielle) ayant un rôle dans l'organisation modélisée
- Objet informatique
  - Entité en mémoire vive caractérisé par : Identité, attributs, comportement
  - Un objet forme un tout
- Identité d'un objet
  - Identifiant de l'objet (peut être son adresse mémoire !)
  - Permet de s'adresser à un objet
  - Elle n'est pas définie par le concepteur
  - Différente de l'identifiant (clé) de gestion, définie par le concepteur

Conception objet

Notion d'objet

- Attributs d'objet
  - Caractéristiques (non visibles) décrivant les objets
  - Possèdent des valeurs propres (privées) à chaque objet
  - Déterminent l'état d'un objet
  - Peuvent évoluer dans le temps (ou pas)
- Exemple :
  - L'entreprise *InfoIT* sise au *143 r Versailles* emploie 500 personnels décrits par leurs numéro, nom et salaire

@x : adresse de l'objet



Conception objet
Notion d'objet

- Comportement d'un objet
  - Opérations (traitements) que peut accomplir un objet
  - Rôles (responsabilités) attribués à l'objet : actions qu'il peut accomplir
  - Part de contribution de l'objet dans l'accomplissement des fonctions de l'application
  - Agit, généralement, sur les attributs de l'objet et il en dépend
- Exemple (suite) : l'entreprise *IfoITSA*
  - Rôles attribués à un objet Personnel :

    Conserver le numéro, nom et salaire du personnel qu'il représente

    Fournir le numéro, nom et salaire du personnel qu'il représente

    Modifier le salaire du personnel

Conception objet

Notion d'objet

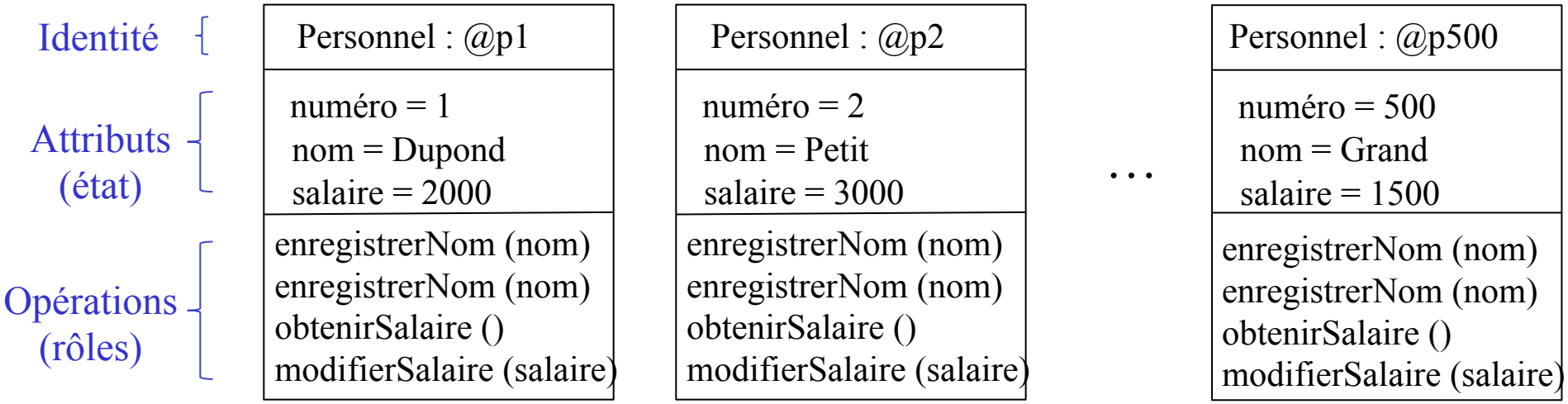
- Exemple : comportement d'un objet *Personnel*
  - Quel est le salaire de *Dupond* ?
  - A quel(s) objet(s) s'adresser ? A qui a-t-on attribué cette responsabilité (rôle) !
  - A celui qui détient l'information !
  - C'est l'objet *Personnel* identifié par *@p1*
- Exemple : comportement des objets *Personnel*
  - Quel est le salaire d'un personnel ?
  - S'adresser à l'objet *Personnel* qui représente le personnel en question
  - C'est l'objet *Personnel* identifié par *@p1*
- Tous les objets *Personnel* possèdent les mêmes responsabilités

Conception objet

Notion d'objet

☞ Exemple (suite) :

– Tous les objets Personnel possèdent les mêmes opérations (rôles) :



Conception objet

Notion d'objet

☞ Exemple (suite) : l'entreprise *IfoITSA* …

– L'entreprise est organisée en 3 services, chaque service est décrit par son nom

Service : @s1
nom = Administratif

Service : @s2
nom = Technique

Service : @s3
nom = Informatique

– Rôles attribués à un objet Service :

Conserver le nom du service qu'il représente

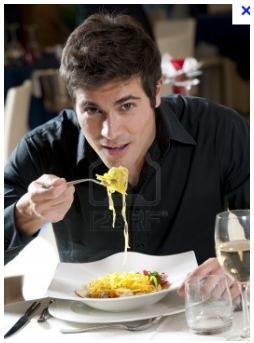
Fournir le nom du service information à la demande

Modifier le nom du service représenté

Conception objet

Notion d'objet

☞ Exemple : quels attributs et comportements ?!



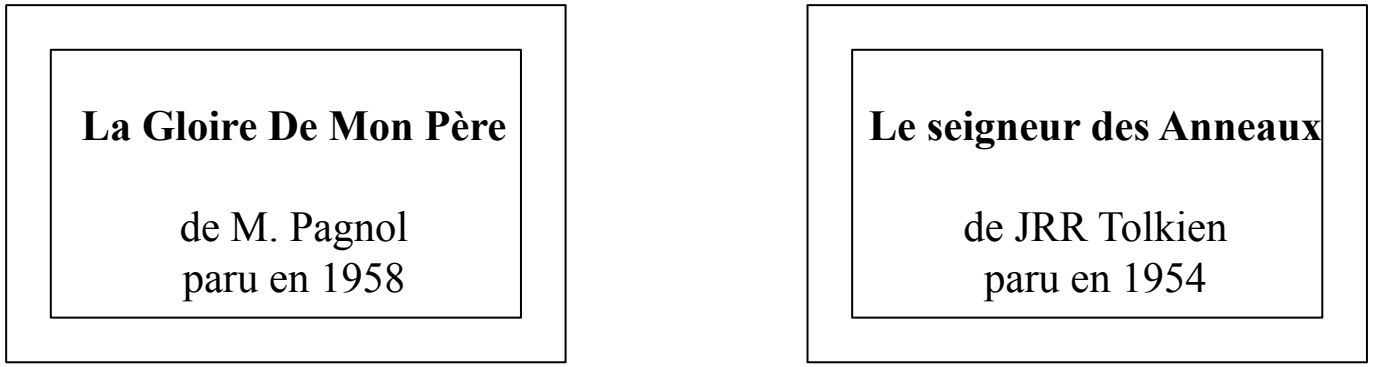
Conception objet

Notion de classe

- Une classe : représentation de plusieurs objets d'une même famille, ayant des propriétés similaires (attributs et comportement)

  Les personnels Dupond, Petit, … sont décrits par leurs numéro, nom et salaire

  Les livres :



  sont caractérisés par les mêmes attributs : titre, auteur et date de parution

  Le résultat de conception sera le même pour l'ensemble de ces objets

- Conception : étude des **propriétés** des objets en faisant **abstraction** des valeurs de leurs attributs les différenciant

Conception objet

Notion de classe

- Classe
  - Structure (abstrait) décrivant les propriétés des objets d'une même famille
  - Type d'objets composé d'attributs et d'opérations
  - Permet de réduire la complexité de la situation étudiée
  - Moule d'objets : permet de créer des objets similaires (instance de classe)
- Attributs
  - Caractéristiques décrivant les objets de la classe
  - Définissent l'état d'un objet de la classe
- Opérations : compétences/rôles
  - Fonctions (traitements) pouvant être réalisées par les objets de la classe
  - Tous les objets de la classes fournissent (peuvent exécuter) ces opérations
- Propriétés d'une classe d'objet : définies par attributs et opérations

Conception objet

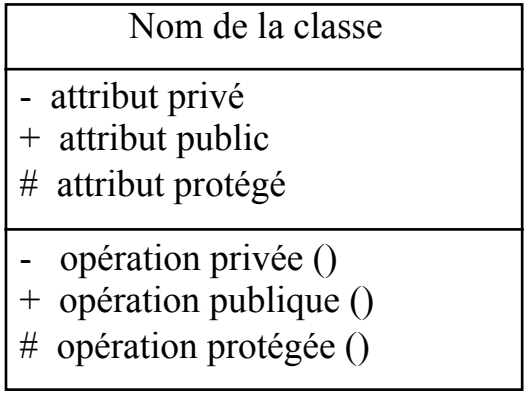
Modélisation de classe avec UML

☞ Eléments de modélisation en UML

- Classes : attributs, méthodes (opérations)
- Relations : association, agrégation, composition, multiplicités et rôles
- Héritage
- Utilisation (dépendance)
- Classes abstraites et Interface
- Réalisation/implémentation

Conception objet

Modélisation de classe avec UML

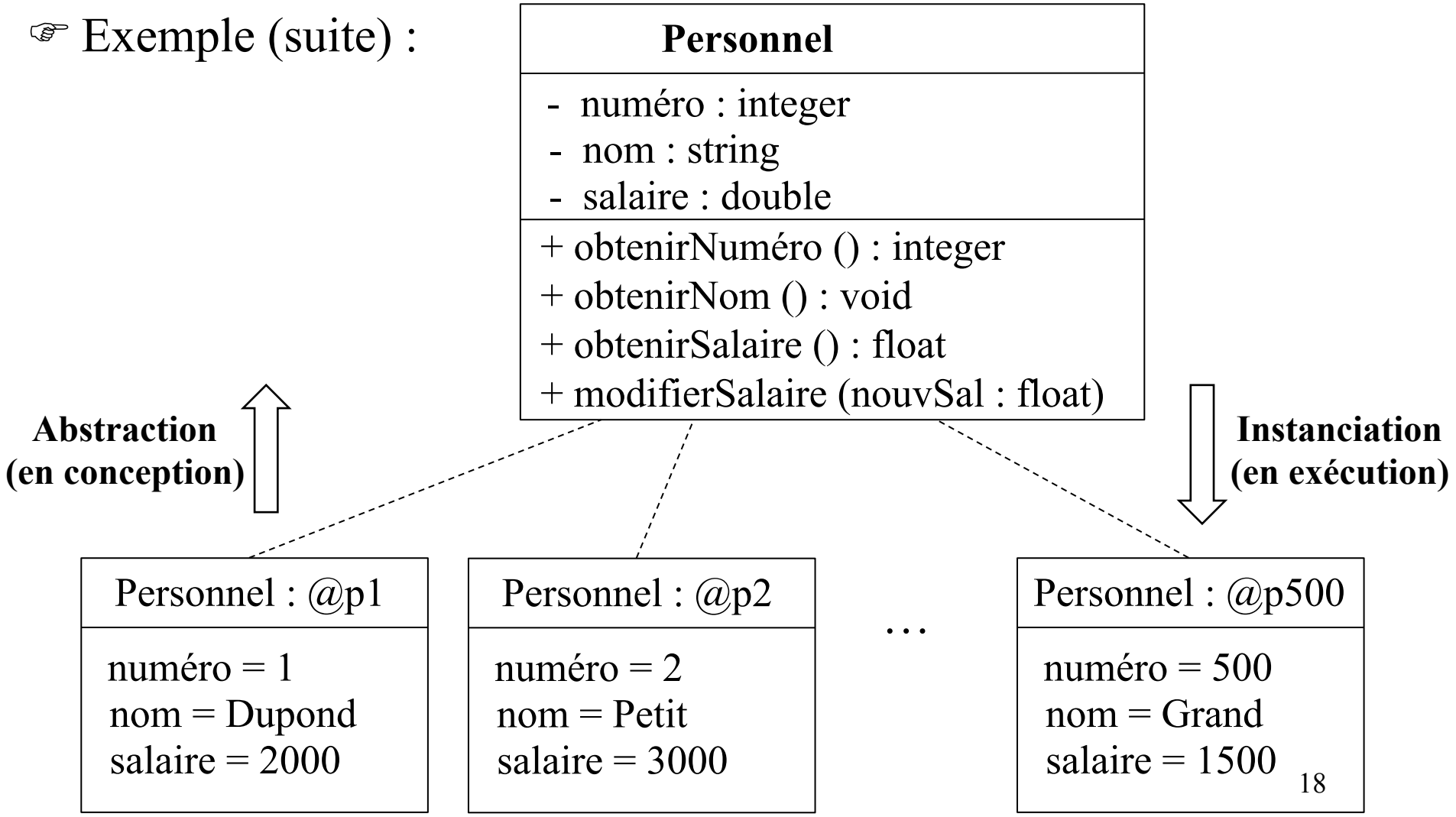


- Classe
  - Structure commune à plusieurs objets
- Encapsulation
  - Les attributs définissent l'état interne (non visible) des objets de la classe
  - Accessibles uniquement par les opérations de l'objet, pas d'ailleurs !
- Objet : instance d'une classe
  - Classe (moule d'objets) : peut servir à créer des objets
  - Objet : obtenu par instanciation (attribuer des valeurs aux attributs) de sa classe
  - Identifié par un *oid* (Object Identifier)

Conception objet

Modélisation de classe avec UML

☞ Exemple (suite) :

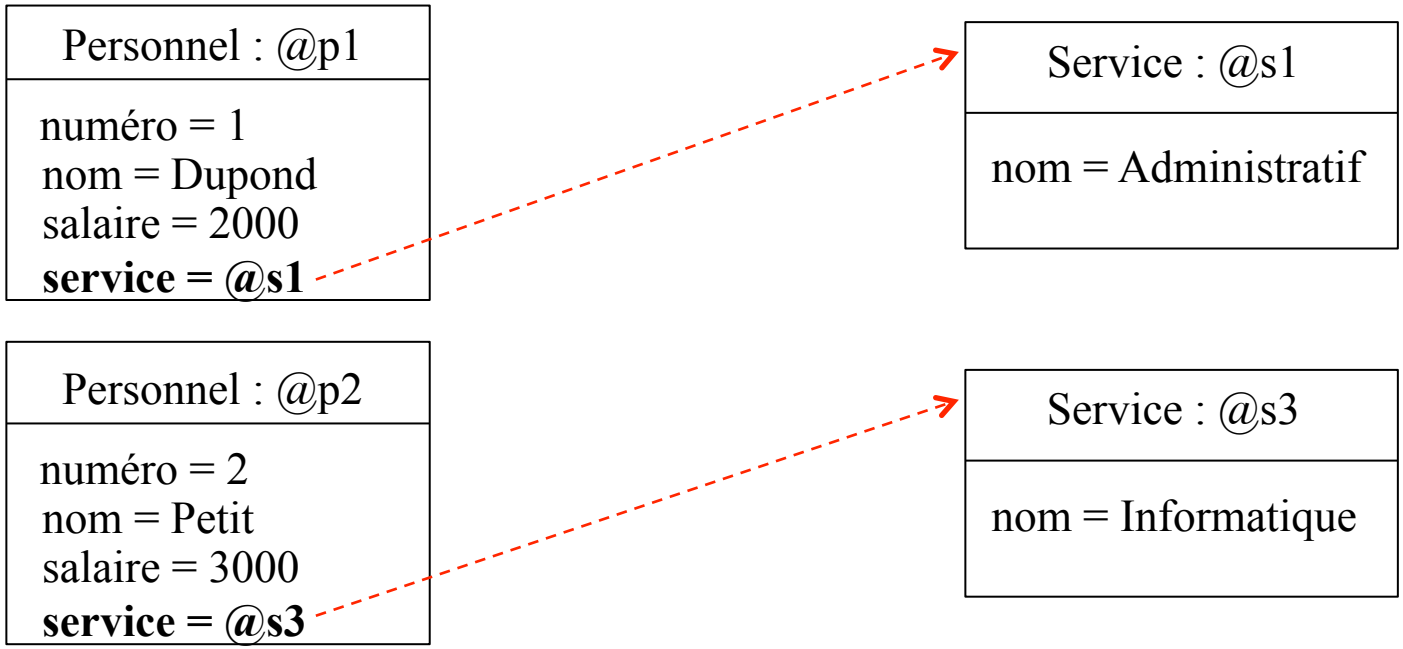


Conception objet

Relations entre objets

☞ Exemple (suite) : un personnel est affecté à un service

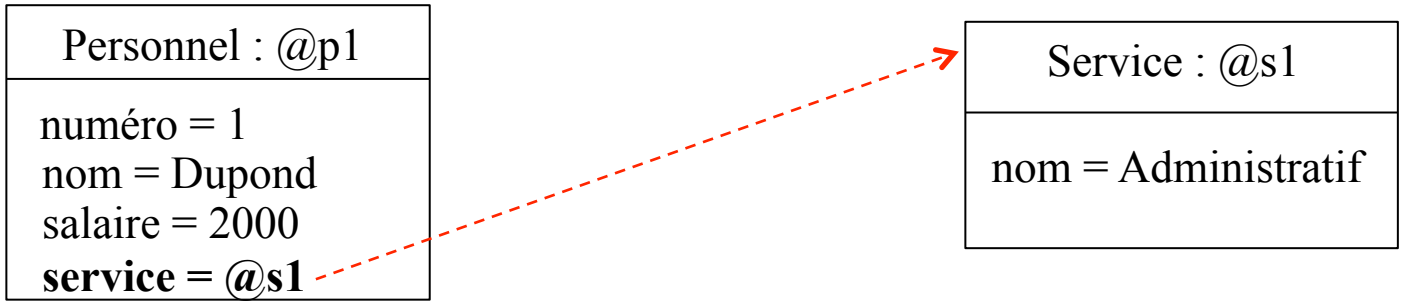
- On voudrait savoir à quel service est affecté un personne
- Un objet personnel garde (stocke) un lien vers le service auquel il appartient
- Exemple : le personnel *Dupond* est affecté au service *Administratif* et le personnel *Petit* au service *Informatique*



Conception objet

Relations entre objets

- Comportement lié aux relations
  - A quel service est affecté le personnel *Dupont* ?
  - A qui s'adresser ?
  - A l'objet qui détient l'information, l'objet Personnel *Dupond* !
  - L'objet *Dupond* répond : c'est le service *@s1*
  - Mais on voulait savoir le nom du service !
  - Il suffit donc de demander au service *@s1* son nom !

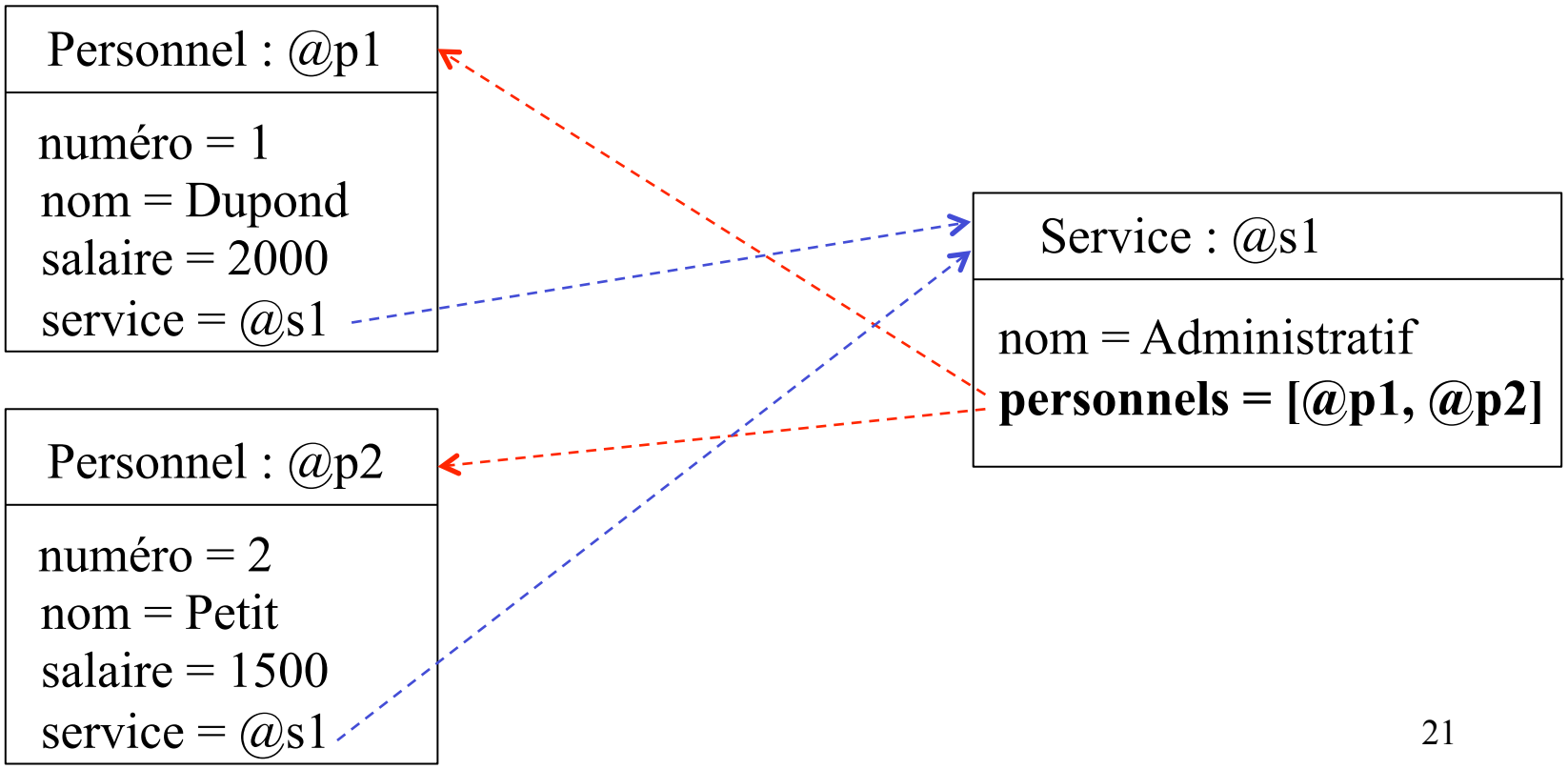


Conception objet

Relations entre objets

☞ Exemple (suite) : un service comprend plusieurs personnels

– Le service *Administratif* comprend les personnels *Dupond* et *Petit*



Conception objet
Relations entre objets

☞ Comportement lié aux relations

– Quels sont les personnels du service *Administratif* ?

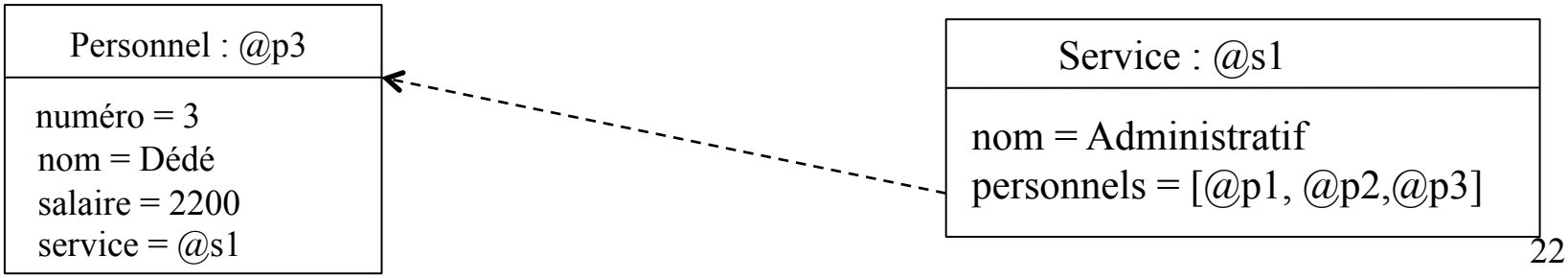
Demander à l'objet Service *@s1* la liste de ses personnels

Il renvoie la liste [@p1, @p2]

– Embaucher un nouveau personnel (3, Dédé, 2200) au service *Administratif*

Créer un nouveau objet Personnel (son identité est *@p3*)

Demander à l'objet @s1 de l'ajouter à la liste de ses personnels



Conception objet

Modélisation de classe avec UML

☞ Exemple : Comportement lié aux relations

Personnel
- numéro : integer - nom : string - salaire : double - **serviceAffecté : Service**
+ obtenirNuméro () : integer + … + **affecterAService (Service s) : void** + **obtenirSonService () : Service**

Service
- nom : string - **personnels : Personnel []**
+ obtenirNom () : string + … + **affecterPersonnel (Personnel p) : void** + **désaffecterPersonnel (Personnel p)** + **obtenirPersonnels () : Personnel []** **…**

Conception objet

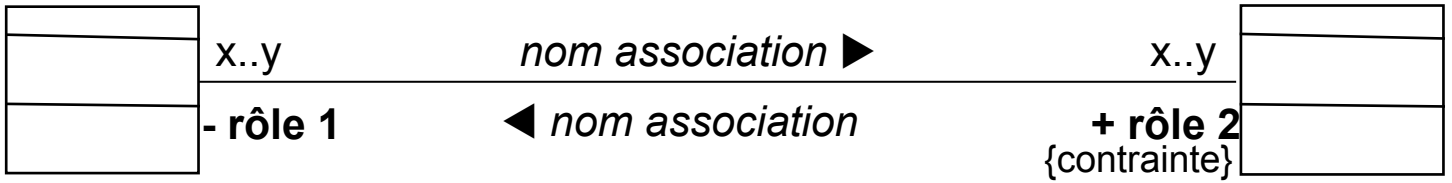
Modélisation de classe avec UML

- Associations entre classes
  - Exprime une relation sémantique bidirectionnelle entre classes
  - Abstraction des liens entre objets de l'application
  - Permettent les interaction entre objets
  - Deux catégories de relations : structurelles et non structurelles
- Association structurelle
  - Exprime une relation sémantique uni ou bidirectionnelle entre classes
  - Lien informatif, permettant de stocker une information d'un objet
  - Exemple : à quel service est affecté un personnel ?
- Association non structurelle (dépendance entre classes)
  - Tout lien ne servant pas à stocker un lien d'objet
  - Exemple : un objet appelle une méthode d'un autre objet passé en paramètre

Diagramme de classes

Relations entre classes

- Association structurelle
  - Exprime une relation sémantique (uni ou bidirectionnelle) entre les objets des classes associées
  - Possède un nom, deux rôles, deux multiplicités, des contraintes, …



  - *Nom de l'association* : forme verbale, sens de lecture avec flèche
  - *Rôles* : forme nominale, identification d'une extrémité de l'association
  - *Multiplicités (x..y)* : 0..1, 1..1 (ou 1), 0..* (ou *), 1..*, M..N

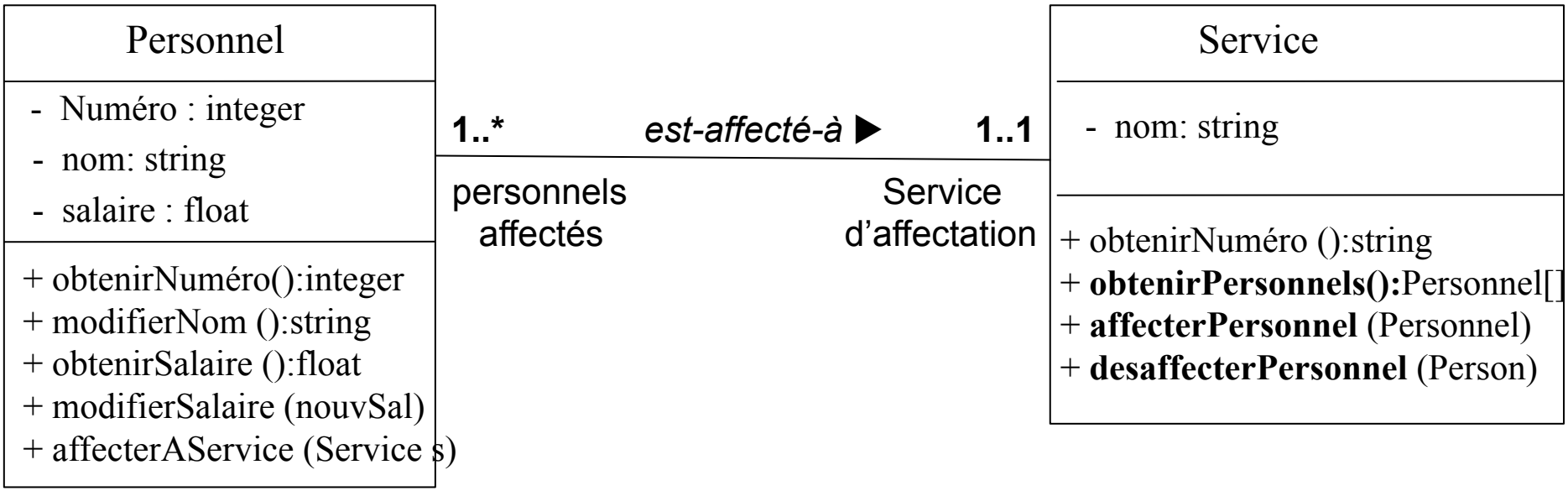
Diagramme de classes

Relations entre classes

☞ Exemple :

– Un personnel est affecté à un seul et unique service

– Un service comprend un ou plusieurs personnels



Conception objet

Analyse des fonctions de l'application

- Une application à objets est une société d'objets
  - Les objets interagissent pour réaliser les fonctions de l'application (*cf.* DCU)
  - Chaque objet contribue en réalisant une partie d'une fonction de l'application
- Conception d'une fonction de l'application
  - Déterminer les entrées et sortie de la fonction → opération principale
  - Décrire les macro-étapes de la fonction
  - Trouver l'objet (classe) à qui attribuer la responsabilité de la fonction (celui à qui confier l'opération)
  - Déterminer les objets (classes) participants à la réalisation de la fonction et le rôle de chacun → opérations intermédiaires

Conception objet

Analyse des fonctions de l'application

☞ Exemple (suite) : embauche d'un personnel (1)

– Embaucher le nouveau personnel ***(501, Dupuis, 2300 eur)*** au service *Administratif* (géré par l'objet ***@s1***)

1. Demander à l'objet *@s1* d'embaucher le nouveau personnel (*501, Dupuis, 2300 eur)*. opération *embaucherPersonnel (numéro, nom, adresse)*

2. L'objet *@s1* crée (instancie) un nouvel objet Personnel *@p501*

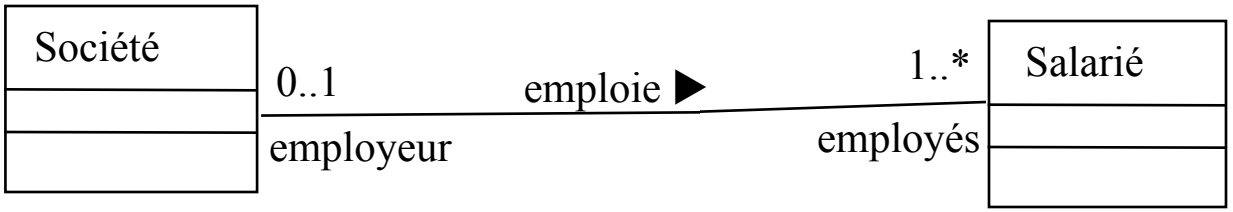
3. L'objet *@s1* demande à l'objet personnel *@p501* de stocker les renseignements (*501, Dupuis, 2300 eur)*

4. L'objet *@s1* ajoute l'objet Personnel *@p501* à sa liste de personnels

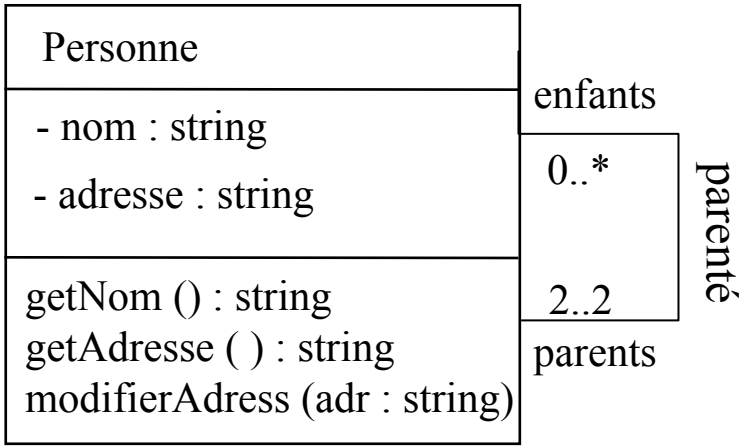
Diagramme de classes

Relations entre classes

☞ Rôle



– Les rôles sont nécessaires pour la compréhension des associations réflexives



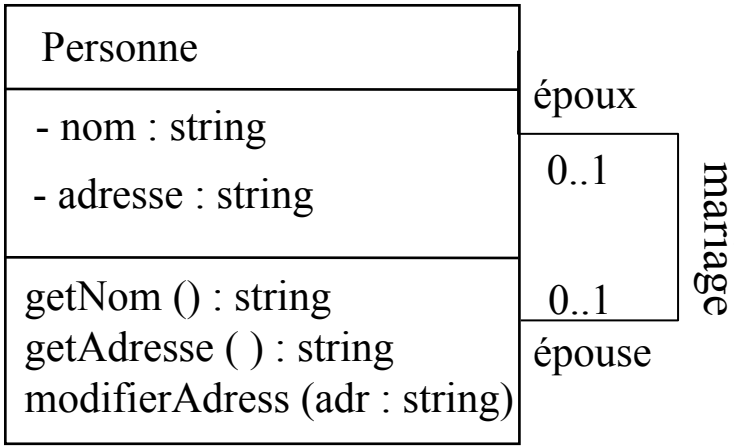
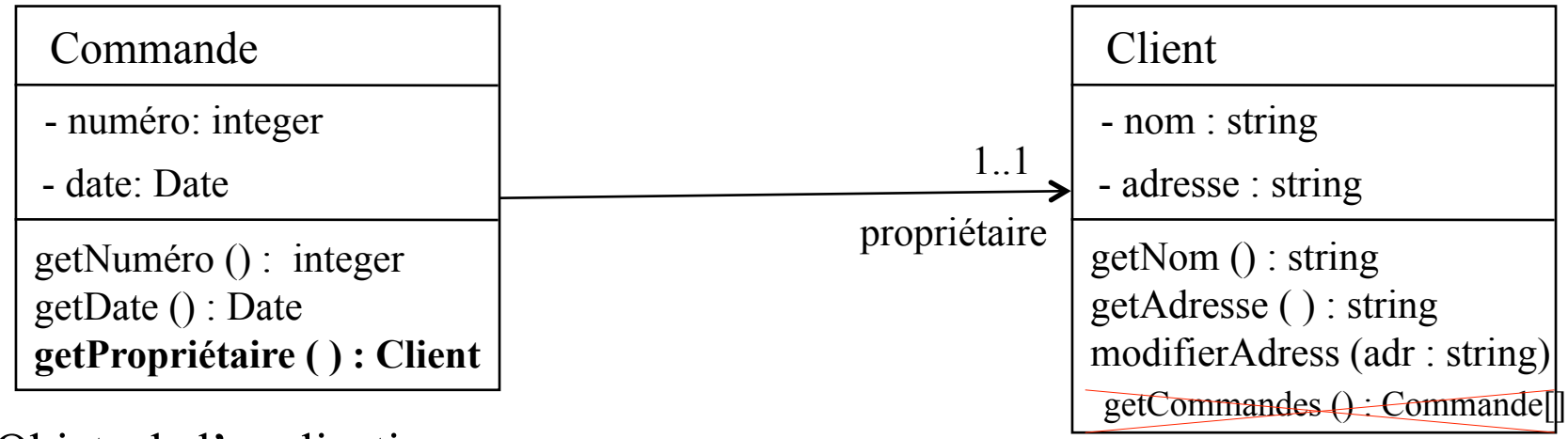


Diagramme de classes

Relations entre classes

☞ Association unidirectionnelle

– Le lien n'est stocké que dans une seule classe !



– Objets de l'application

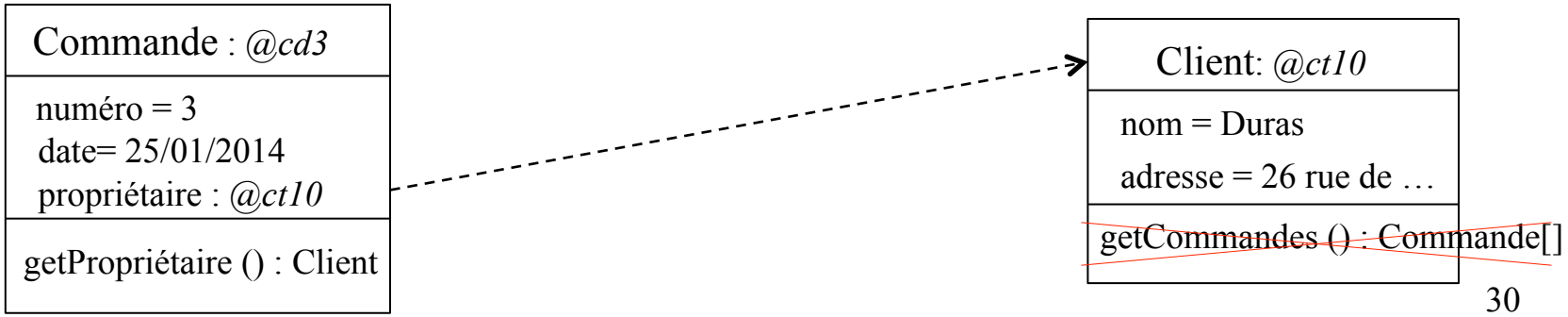


Diagramme de classes

Relations entre classes

- Agrégation
  - Association exprimant le lien de composition d'objets : *composite (agrégat)* et *composant (agrégé)*
  - Non symétrique : une des extrémités joue un rôle prédominant par rapport à l'autre
  - Association simple dont le nom pourrait être *est-partie-de, fait-partie-de, est-composé-de, est-constitué-de, ...*
  - Un objet *composant* peut être utilisé pour composer un ou plusieurs objets *composites*

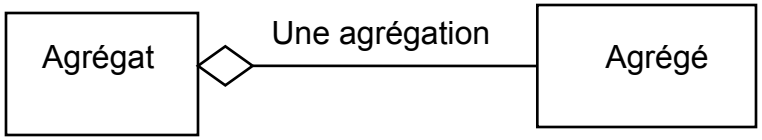
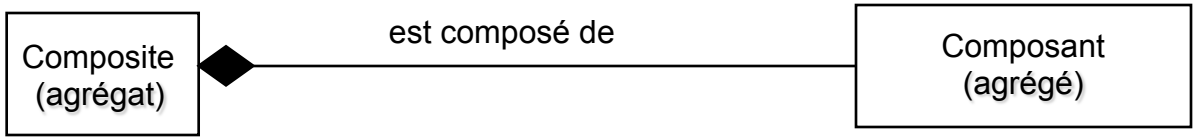


Diagramme de classes

Relations entre classes

- Composition
  - Cas particulier de l'agrégation. C'est une agrégation forte
  - La durée de vie de l'objet *composant* est limitée à celle de l'objet *composite*
  - La destruction de l'objet *composite* implique celle de tous ses *composants*
  - Un objet *composant* ne peut composer qu'un seul objet *composite*



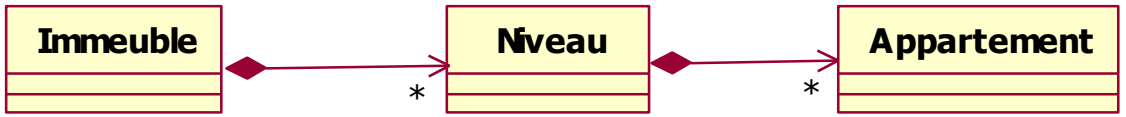
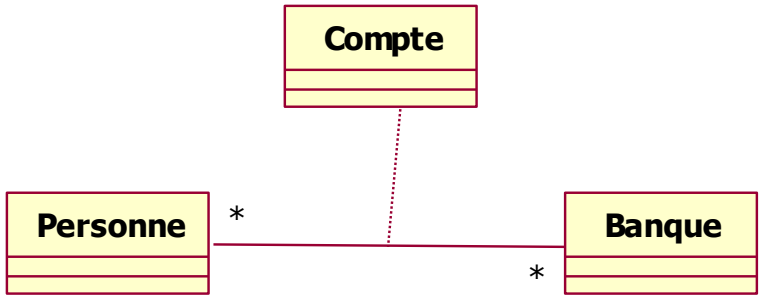


Diagramme de classes

Relations entre classes

- Classe d'association
  - Une association peut être amenée à porter des données
  - Une donnée portée par une association dépend (fonctionnellement) à la fois de tous les objets participants à cette association
- Exemple

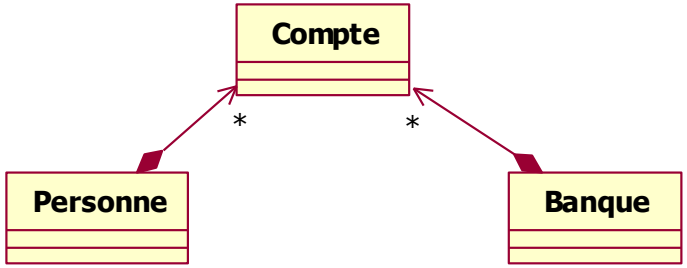


- Contraintes implicites
  - L'existence d'un objet de la classe d'association dépend de l'existence des objets associés
  - Une association ne peut lier qu'une seule fois les mêmes objets

Diagramme de classes

Relations entre classes

- Transformation de la classe d'association
  - Contrainte implicite : l'association ne peut lier qu'une seule fois les mêmes objets
  - Comment se passer de cette contrainte ?
- 1ère possibilité



- 2ème possibilité : contrainte {bag}

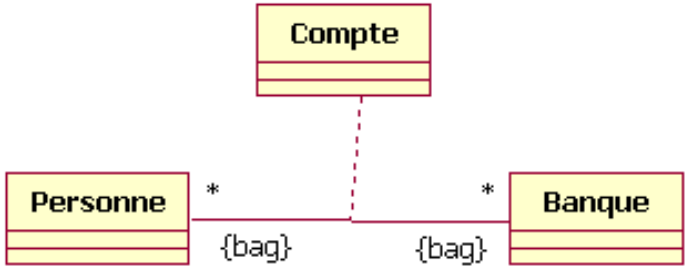
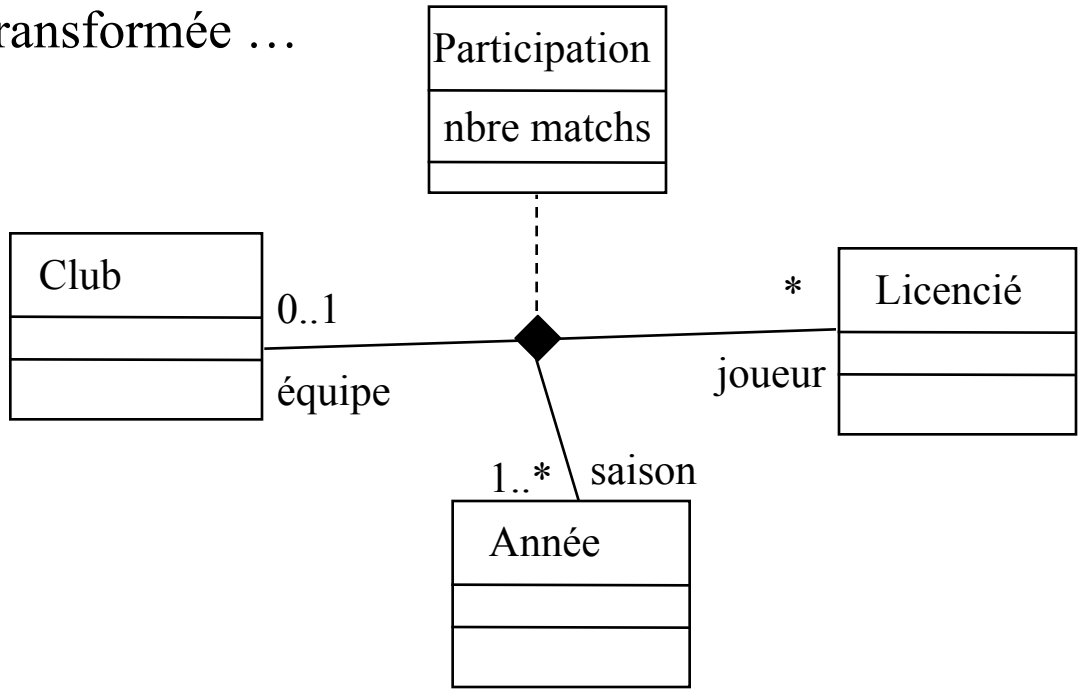


Diagramme de classes

Relations entre classes

- Association *n-aire*
  - Association de trois objets ou plus
  - Sémantique ambigüe
  - Souvent transformée …
- Exemple



Conception objet

Modélisation de classes

- Reste à voir
  - Héritage et polymorphisme
  - Interface, implémentation et polymorphisme d'interface
- A suivre …