CHAPITRE 5 : LA VUE STATIQUE DU SYSTÈME

Le Diagramme de Classes (DCL)

Le Diagramme d’Objets (DOB)

Un bon modèle n’est pas un modèle où l’on ne peut plus rien ajouter, mais un modèle où on ne peut plus rien enlever

A. de St-Exupéry

Things must be as simple as possible, but no simpler»

. A.

Partie I : le diagramme de classes

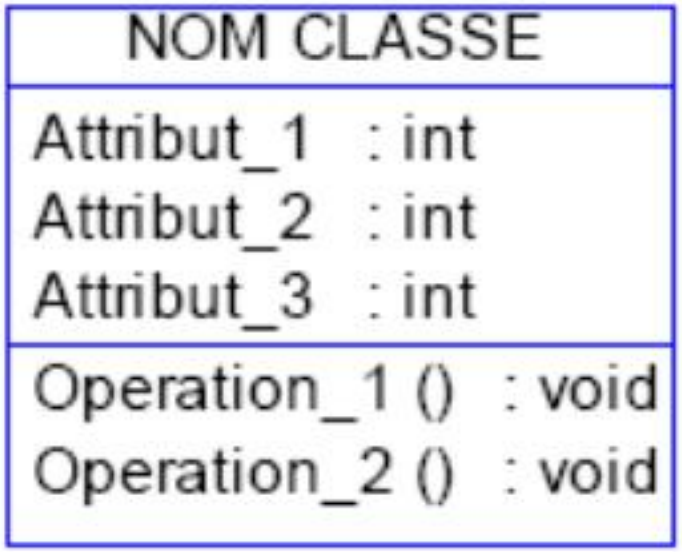
Définition d'un DCL

- Un modèle de représentation statique de l'ensemble des informations finalisées (gérées par le domaine)
- Exprime la structure interne du système en termes de
  - **Classes : entités du système (Informations structurées, regroupées dans des classes)**
  - **et de relations entre ces classes.**
- comporte 6 concepts : classe,

Classe – Définition

- **Type de données abstrait (modèle)** caractérisé par des propriétés (attributs et opérations) communes à tout un ensemble d'objets, qui sont

ce



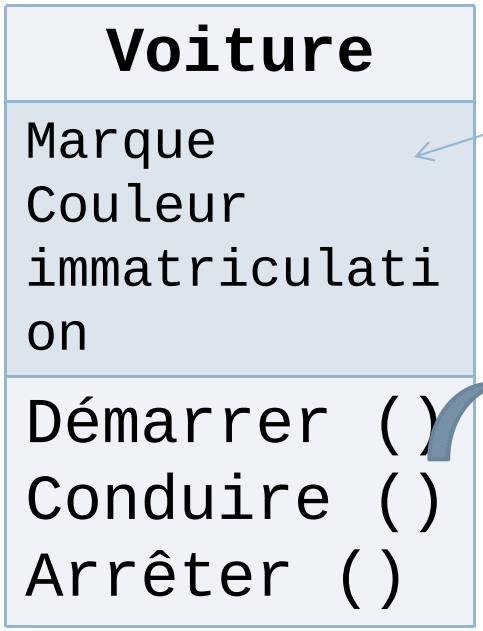
FACTURE
+N° facture : int +Date : date +Montant : double +/ Montant TVA : double
Editer () : void

Classe - Objet

- Unité atomique formée de l'union d'un état et d'un comportement et se caractérise par :
  - **identité:** unique pour se distinguer des autres objets, indépendamment de son état (ex. code du produit)
  - **ensemble d'attributs décrivant son état :** variables de données stockant des informations sur l'état de l'objet.
  - **ensemble d'opérations définissant**

Classe – Objet

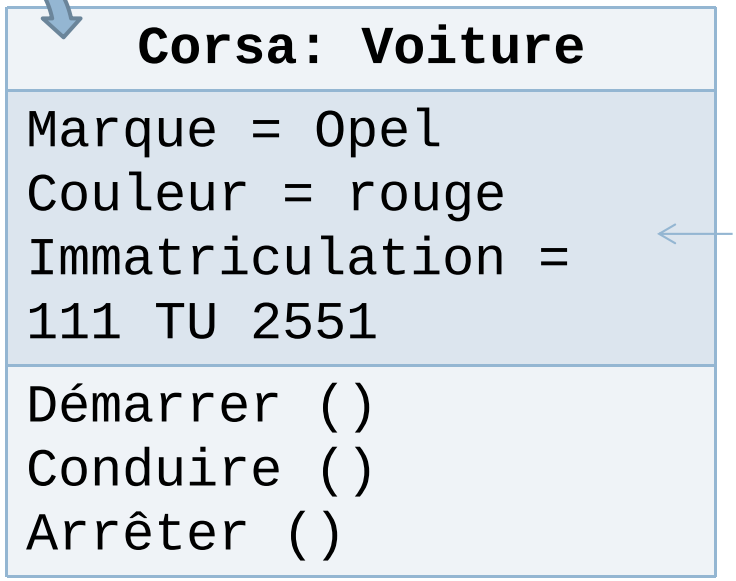
Classe = nom +attributs + opérations



Instanciation en Objets

Regroupement de données et de traitement dans une classe

Instanciation

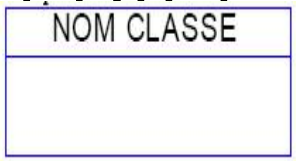


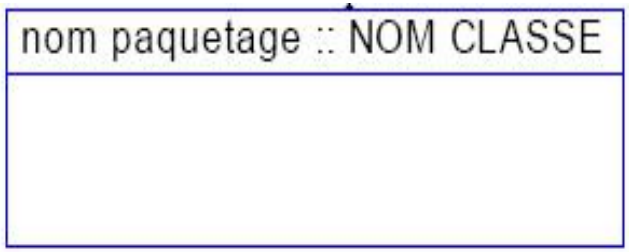
Un objet est une instance de classe (une occurrence d'un type abstrait).

Formalisme – Nom de la classe

1. simple <NomClasse>
2. ou complet

[« stéréotype »]
[<NomPackage1>:: … ::<NomPackageN>::]
<NomClasse>[{[abstract],[aute
[état] ..}]





FigureGéométrique
{abstract,
auteur:Ilhem,
état:validée}

Personne

Java::util::Vector

« Controleur »
Legardien

moula

Classe – Encapsulation

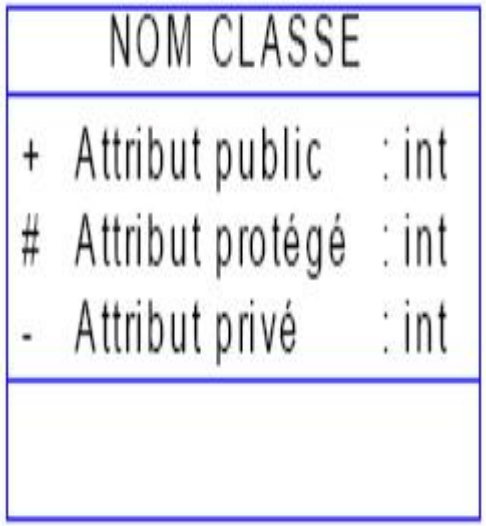
- **Masque les détails** d'implémentation d'un objet, en définissant **une interface (**vue externe d'un objet, définissant les services accessibles aux utilisateurs de l'objet).
- **Facilite l'évolution d'une application** car elle stabilise l'utilisation des objets => On peut modifier l'implémentation des attributs d'un objet ou la façon dont l'objet est utilisé sans modifier son interface

Formalisme – Attributs

- Propriété nommée d'une classe, décrit l'ensemble de valeurs que les instances de cette propriété peuvent prendre.
- Une classe peut avoir 0 à plusieurs attributs.
- La forme la plus complète :

<visibilité> [/]<nom_attribut> : <NomClasse> ['[' multiplicité ']' : <type> [= valeur initiale]

Visibilité et portée des attributs

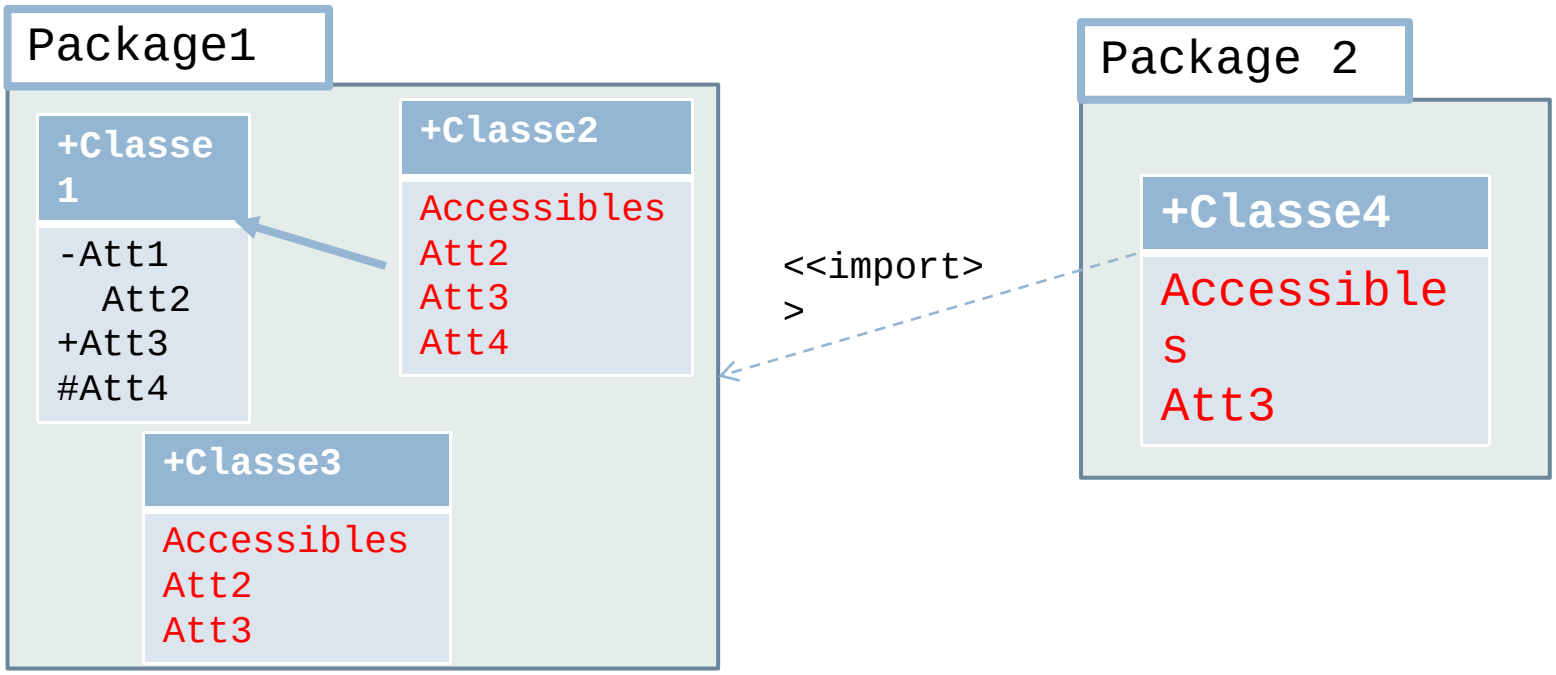


1- publique (+) : l'attribut peut être utilisé par plusieurs classes

2- protégé (#) : l'accessibilité à l'attribut est limitée à la classe et aux sous-classes

3- privé (-) : l'attribut n'est

Visibilité des propriétés d'une classe



L’identifiant

- Attribut particulier, qui permet de repérer de façon unique chaque objet, instance de la classe.

FACTURE
+N° facture : int +Date : date +Montant : double **+/ Montant TVA : double**
Editer () : void

Définitions ….

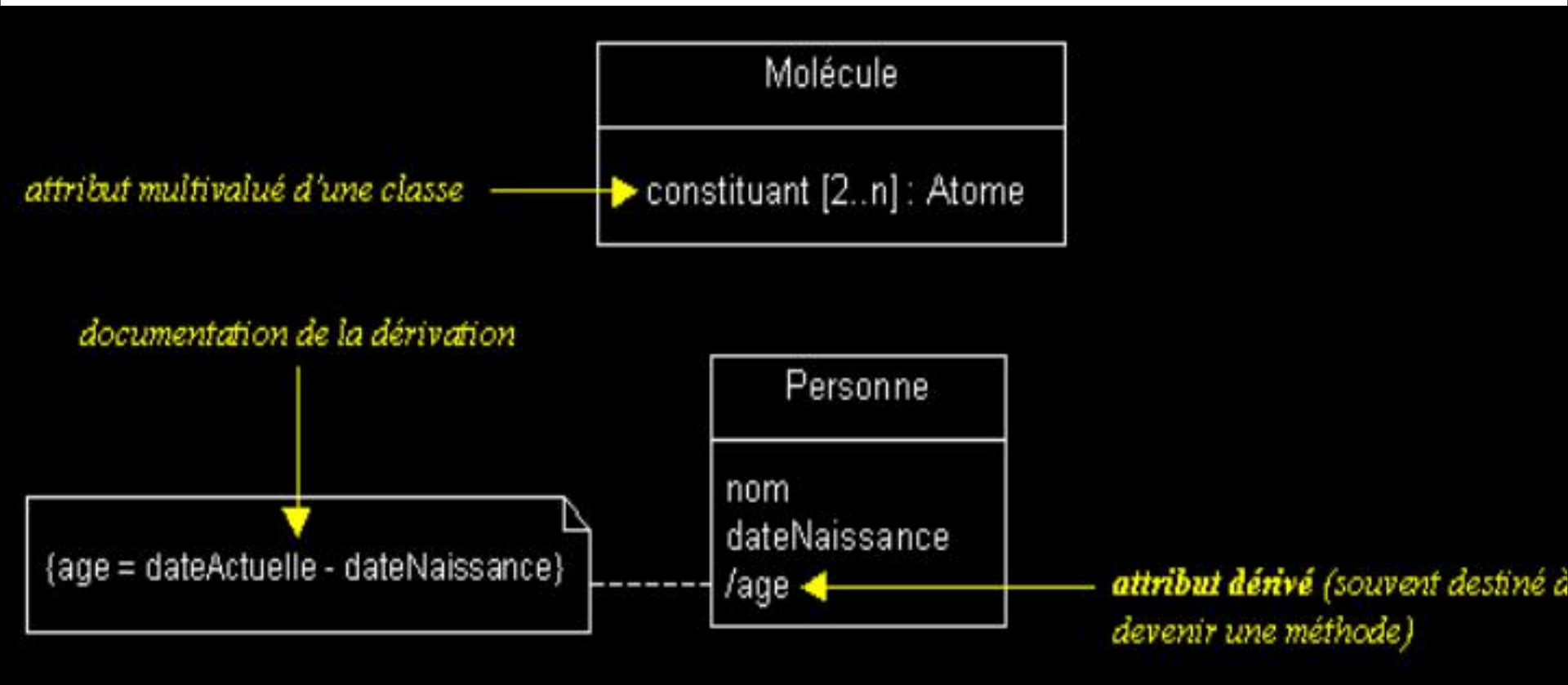
- **Multiplicité** = nombre d'instances qu'un attribut peut avoir => attribut multivalué
  - Ex : *adresse [1..3]* On peut avoir au maximum trois adresses (domicile, bureau, vacances)

- **Type** : un attribut peut être de type simple (chaine de caractères, entier, booléen, etc.) ou complexe (enregistrement, classe, etc.)

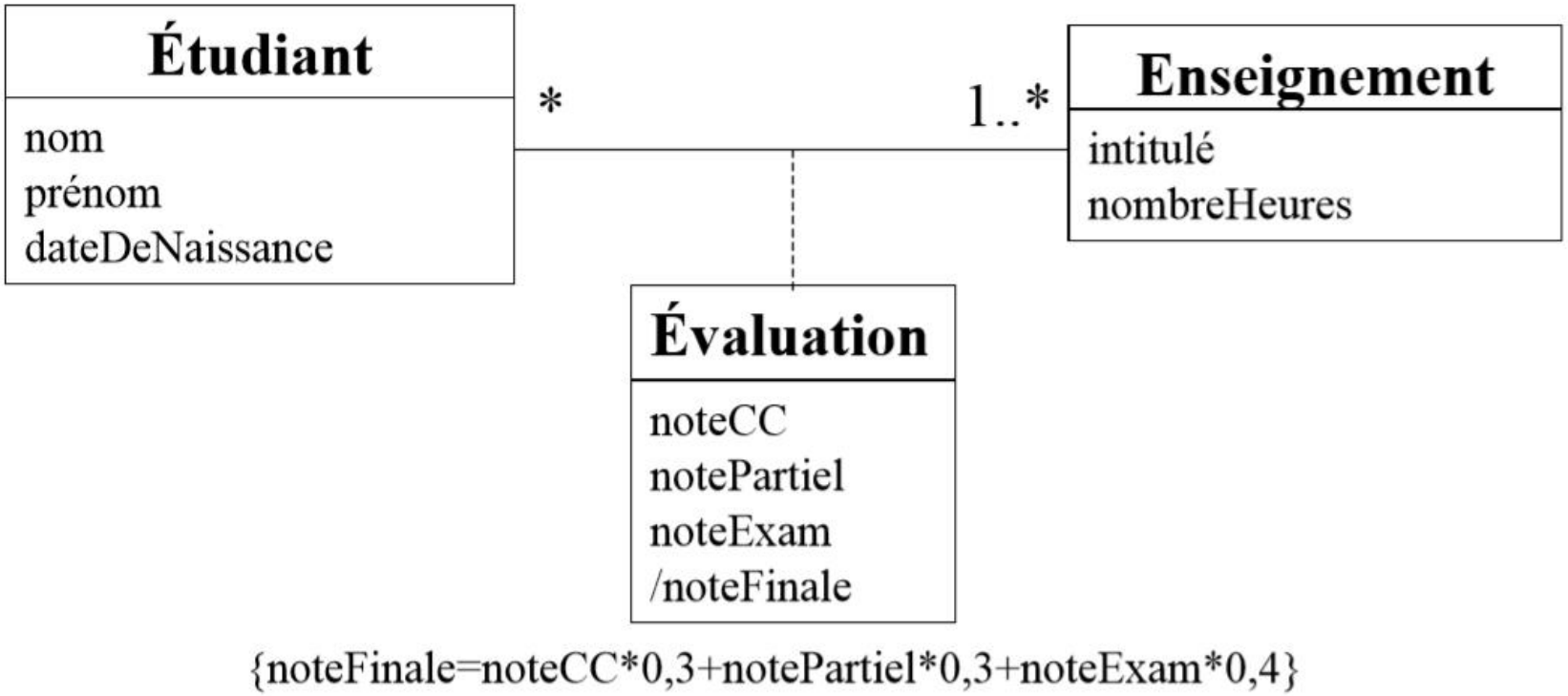
- **Valeur initiale** : valeur prise par défaut
  - Ex : *taille : entier = 0*

Attribut Dérivé

- Valeur calculée à partir d'autres attributs de la classe. On place alors un « / » devant son nom.



Exemple


Étudiant
nom
prénom
dateDeNaissance
*
1..*
Enseignement
intitulé
nombreHeures
Évaluation
noteCC
notePartiel
noteExam
/noteFinale
{noteFinale=noteCC*0,3+notePartiel*0,3+noteExam*0,4}

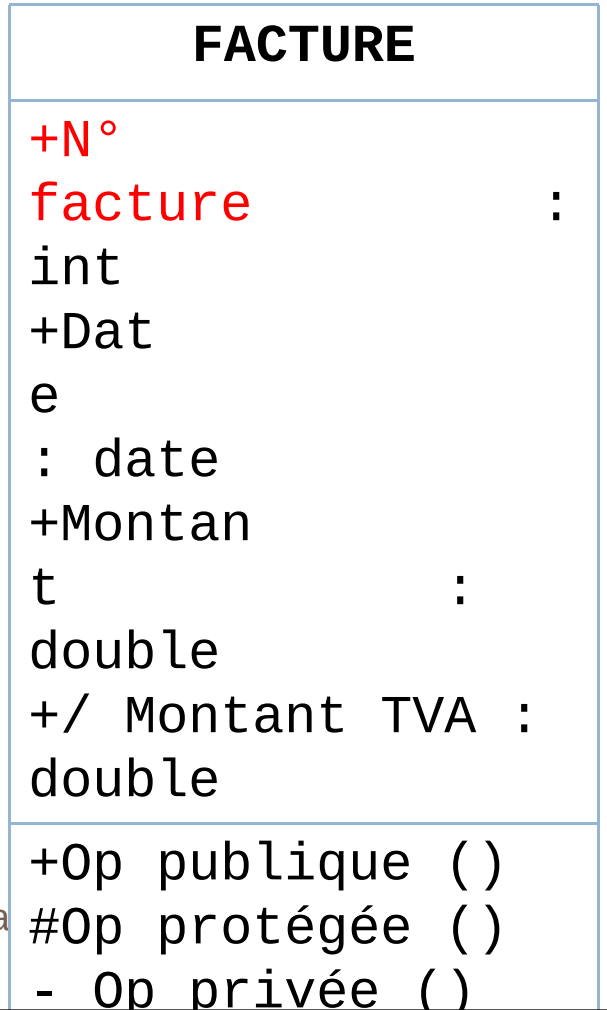
Les opérations

- UML différencie l'opération de la méthode.
  - Une méthode est l'implémentation de l'opération, algorithme exécutable, désigné dans un langage de programmation ou du texte structuré.
  - Une opération est une fonctionnalité assurée par une classe, représente un élément de comportement des objets de la classe.

Visibilité Nom_opération (liste_param): type_résultat

Visibilité

- 3 niveaux de visibilité:
  1. **publique (+)** : l'opération est visible pour tous les objets de la classe
  2. **protégée (#)** : l'opération est visible pour les sous-classes de la classe
  3. **privée (-)** : l'opération n'est visible que par les objets de la classe dans laquelle elle est déclarée



Paramétrage

- L'opération peut avoir 0 ou plusieurs paramètres
  - Chaque paramètre a la syntaxe suivante :

    direction nom_param: type = valeur initiale

    - ***Direction : in, out, inout***

    +créer ()

    +créer Agent ()

    +créer (nom: string)

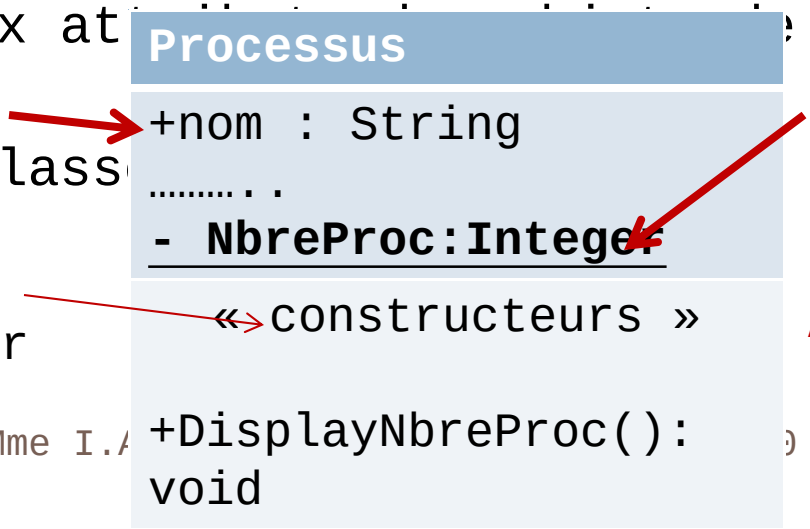
    *+créer (nom : string = «***** »)*

    +créer (nom : string) : personne

Opération de classe

- Opération propriété de la classe et non de ses instances
- Une opération qui ne dépend pas des valeurs propres de chaque objet, mais porte sur les attributs de la classe ou sur les paramètres de l'opération.
  - Pas d'accès aux at… la classe
  - Opération de class…

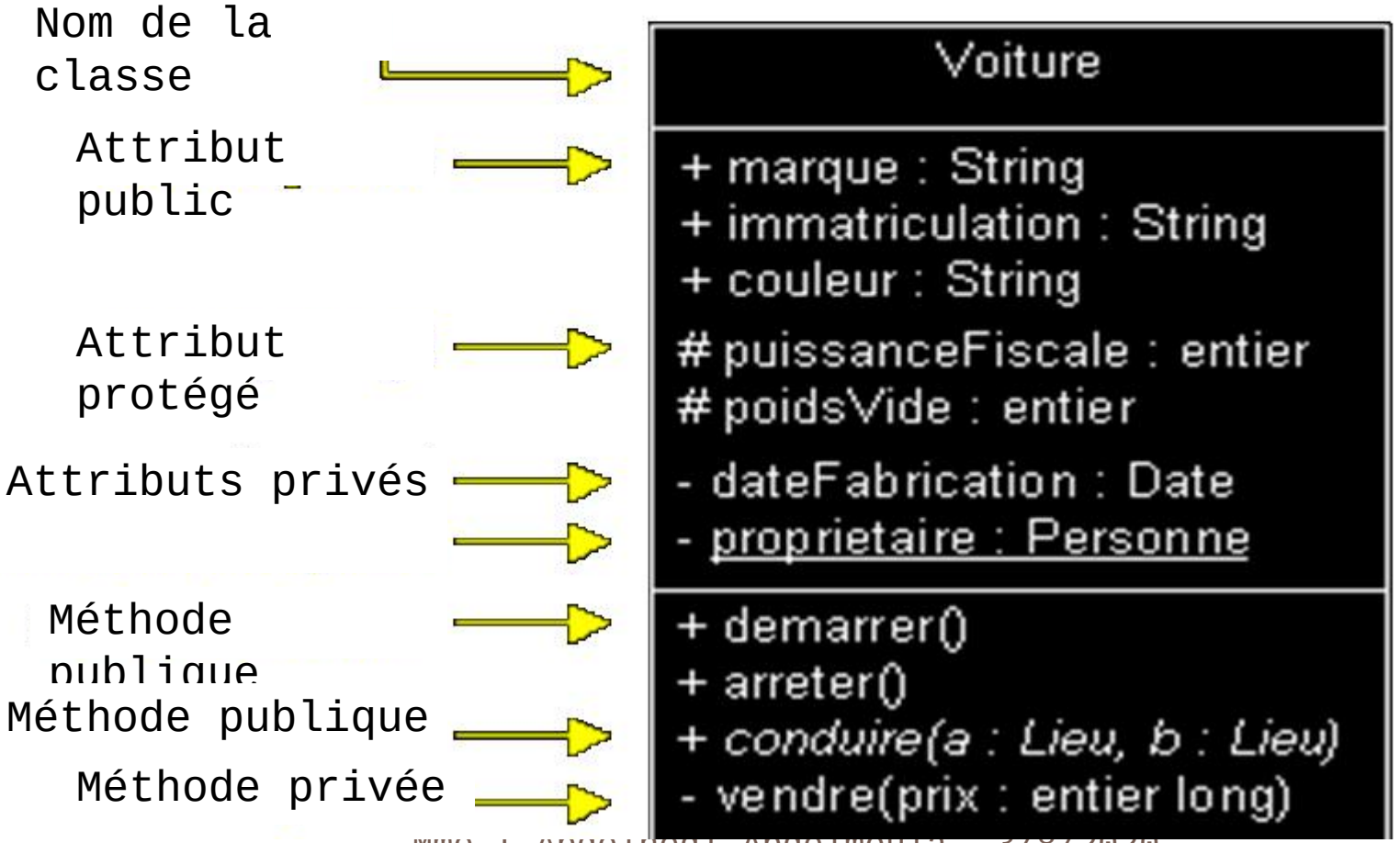
Attribut qui sera associé à chaque processus



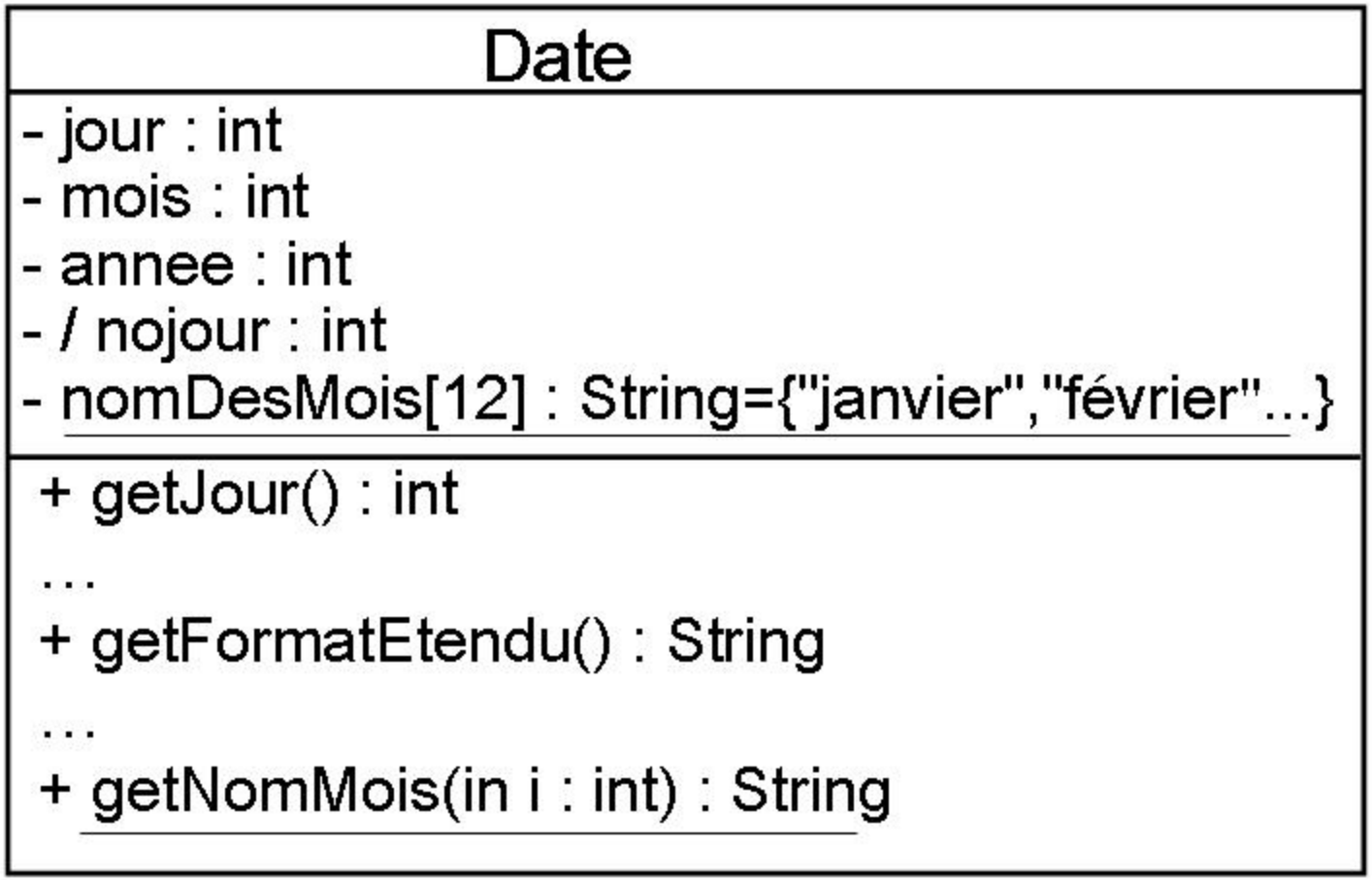
NbreProc n'est disponible qu'en un seul exemplaire **Attribut de classe** associé uniquement à la classe

Le stéréotype type de l'opération utilisé pour simplifier la notation ou lorsque les opérations ne sont pas

Résumé : Exemple de



Exemple


Date
- jour : int
- mois : int
- annee : int
- / nojour : int
- nomDesMois[12] : String={"janvier","février"...}
+ getJour() : int
...
+ getFormatEtendu() : String
...
+ getNomMois(in i : int) : String

Compartiments complémentaires d'une classe

- Les responsabilités, les contraintes générales, les exceptions, etc..

ConnexionInformatique
+ip : Byte[4]
Respecter la charte Posséder une adresse IP
+ SeConnecter(…)
« exceptions » Connexion impossible

Exercice 1

- Proposez une modélisation de la classe Personne spécifiée comme suit :
  - Une personne est caractérisée par son nom, son prénom, son sexe et son age. Ces attributs sont privés.
  - Elle comporte 3 opérations : le calcul de l'âge, le calcul du revenu et le paiement des charges.
    - 2 types de revenus sont envisagés: le salaire et toutes les sources de revenus autres que le salaire
    - Les charges sont calculés en appliquant un coefficient fixe de 15% sur les salaires et un coefficient de 20% sur les autres revenus.
  - Un objet de la classe personne peut être créé, en particulier à partir du nom, de son sexe et de la date de naissance. Il est possible de changer son prénom. Par ailleurs le calcul des charges ne se fait pas de la même manière lorsque la personne décède.

Solution : exercice 1

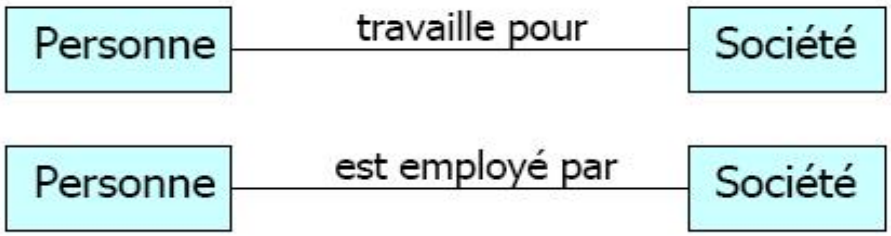
Personne
- Nom: string -Prénom : String -Sexe:string ={‘homme’, ‘femme’} -DateNaissance: Date -/age : integer -Salaire:double -Autrerevenus:double -/Revenu: double -/charge: double -Etat:string={‘vivant’, ‘décédé’}
calculerAge (d: Date, e: string):integer ModifierPrenom (p:string): String calculrevenu(a,s:double):double calculcharge(a,s:double, e:string):double

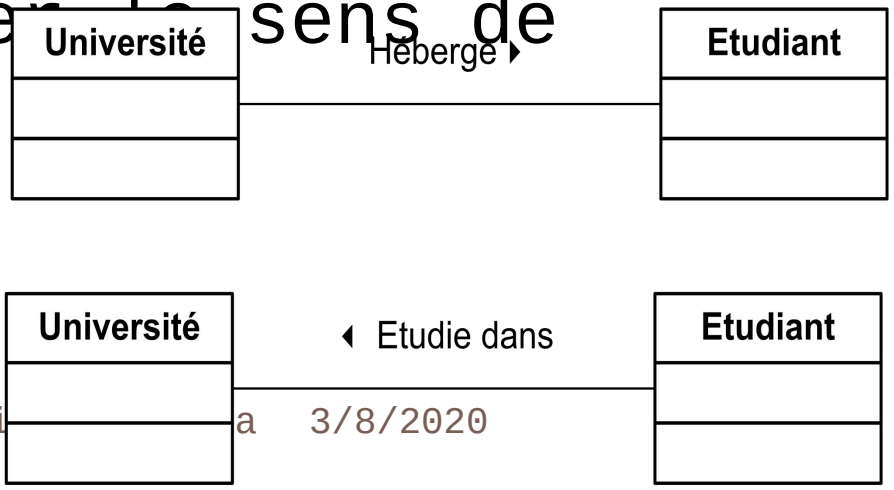
Association – Définitions

- Une relation statique relie 2 à plusieurs classes entre elles.
- Exprime une connexion sémantique bidirectionnelle entre deux classes.
  - Existe entre les classes et non entre les instances de ces classes: montrer une structure et non des échanges de données.
  - Autorise la navigation d'un objet d'une classe à un autre objet de

Nom de l'association

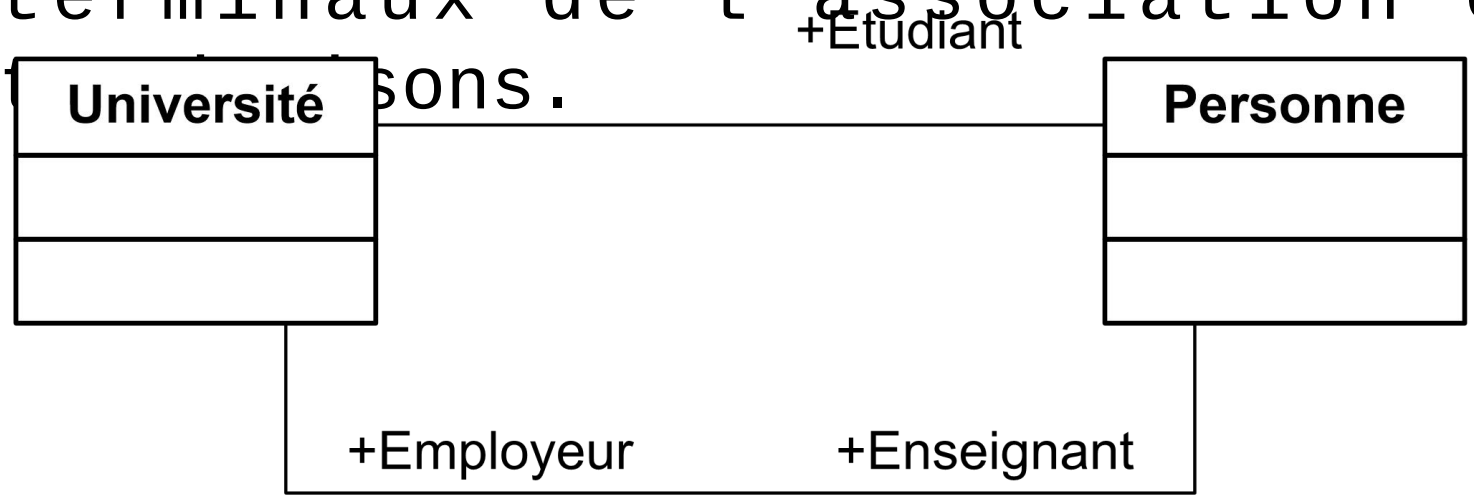
- **Nom en forme verbale active**, sert à décrire la nature de la relation.
- Par défaut le sens du lecture du nom d'une association est de gauche à droite
  - On peut ajouter l'un des signes < ou > pour indiquer le sens de





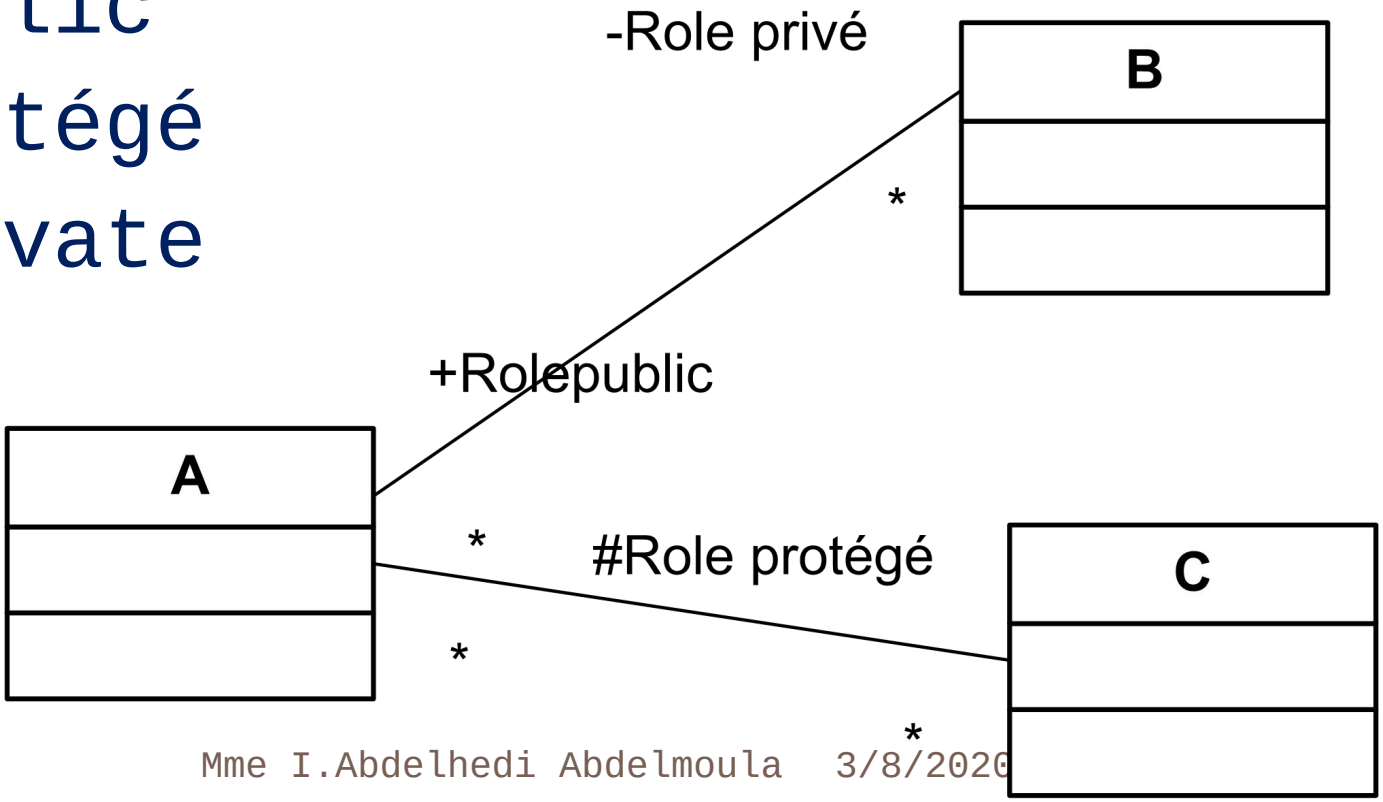
Rôle de l’association

- Chaque classe qui participe à l’association joue un rôle donné. Une association n-aire possède n rôles qui sont les points terminaux de l’association ou [illegible]sons.



Visibilité des rôles de l'association

1. Public
2. Protégé
3. Private



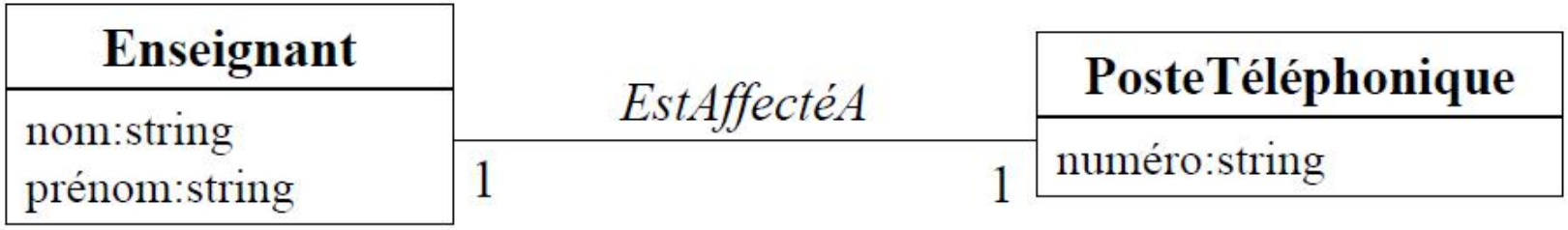
Multiplicité de l'association

- Définit le nombre (entier ou un intervalle de valeurs) d'instances d'une classe qui participent à l'association pour une seule instance de la [illegible] multiplicité est no[illegible]

1	Exactement une instance
0..1	Une instance au plus
1..*	Une instance à plusieurs (1 à n)
*****	Plusieurs instances (0 à n)
9	Nombre fixe d'instances
2-5	Intervalle fixe de nombre d'instances
1,3,	Cardinalité spécifiée

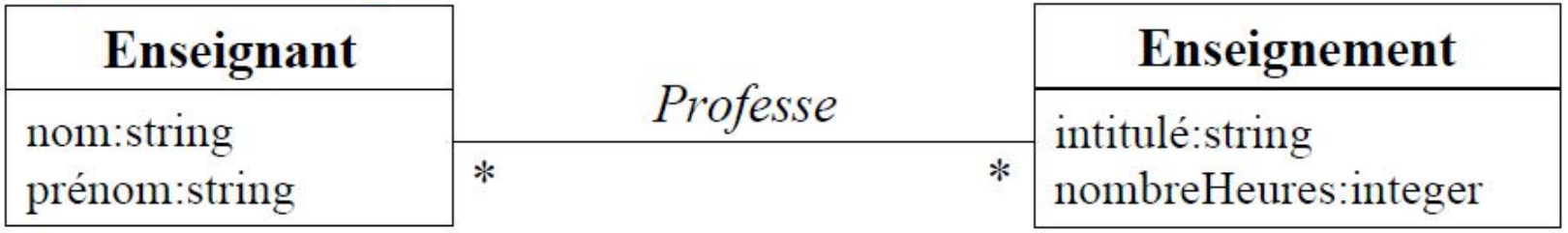
Exemple

- « un-à-un »



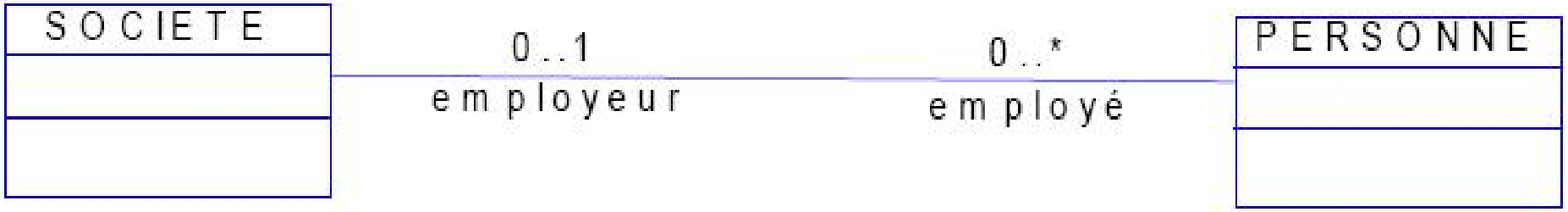
- « zéro-à-un »

- « plusieurs-à-plusieurs »



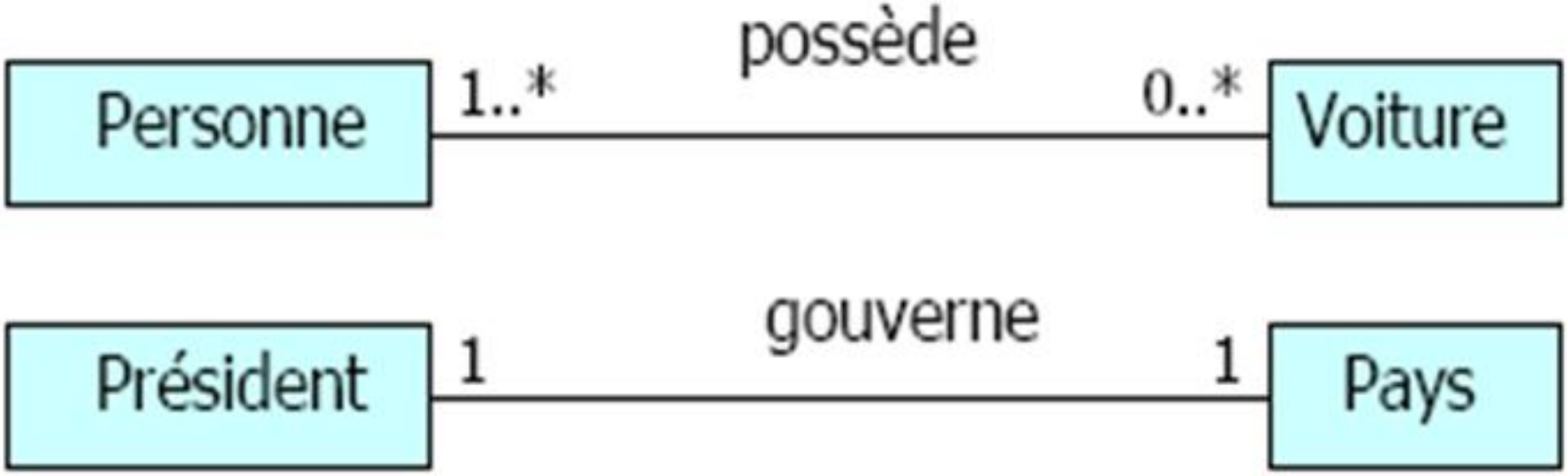
Formalisme

- Les noms des rôles et leur multiplicité



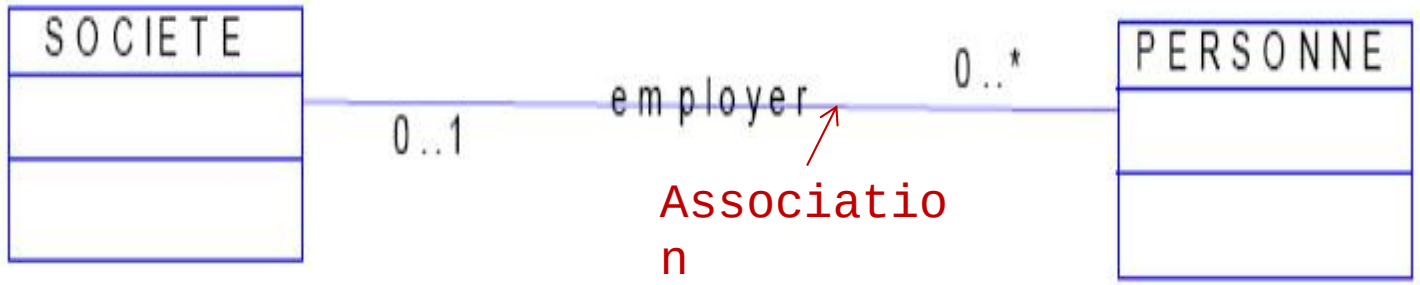
- Le nom de l'association et sa

Exemple



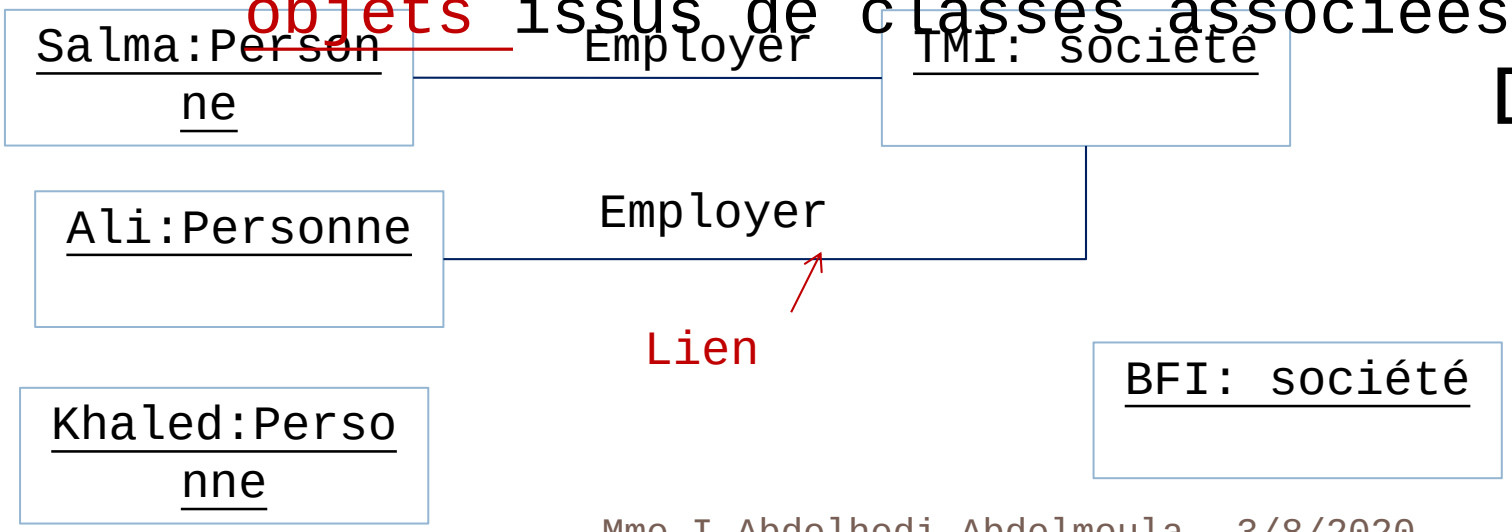
DCL => Diagramme d'objets

DCL



Une association est instanciable dans un diagramme d'objets, sous forme de liens entre objets issus de classes associées.

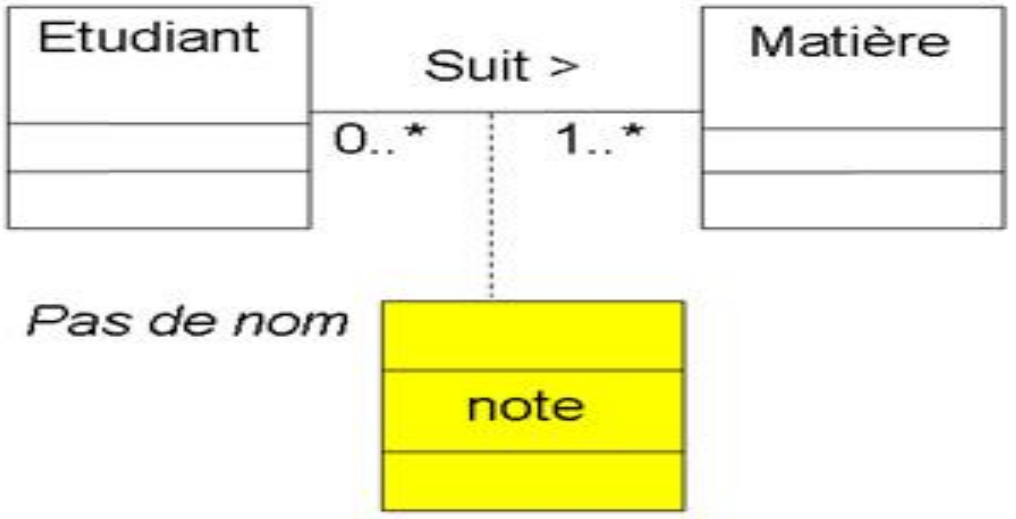
DOBJ



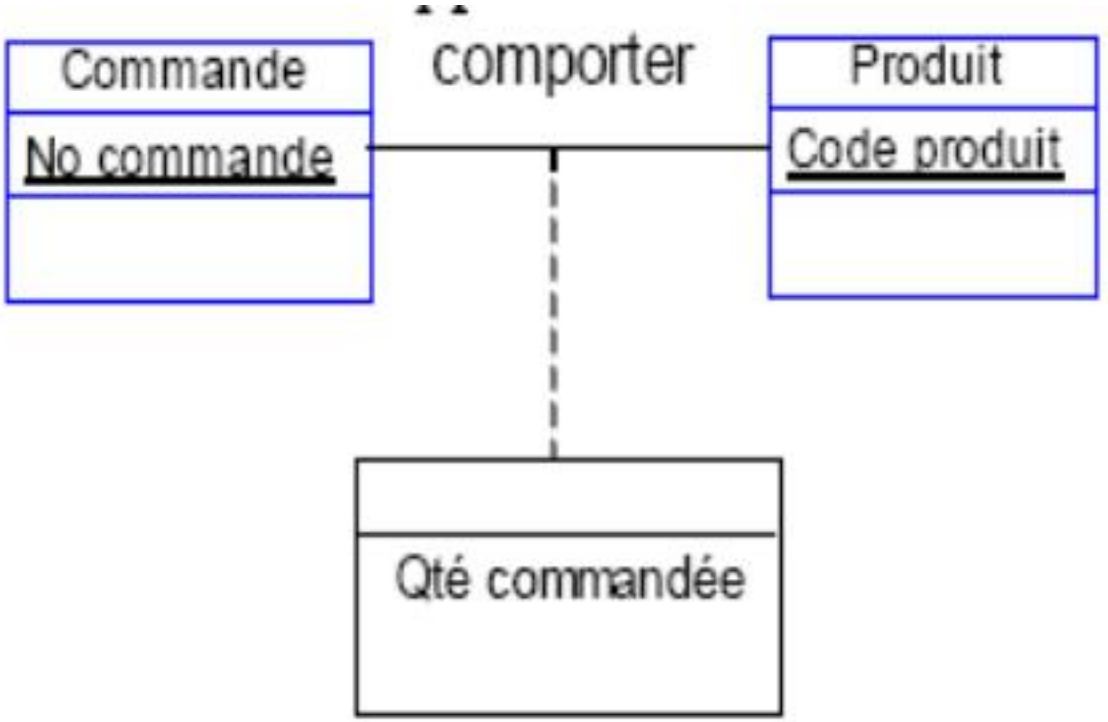
Classe Association

- Les attributs d'une classe dépendent fonctionnellement de l'identifiant de la classe.
- Lorsqu'un attribut dépend fonctionnellement de 2 identifiants, appartenant à 2 classes différentes

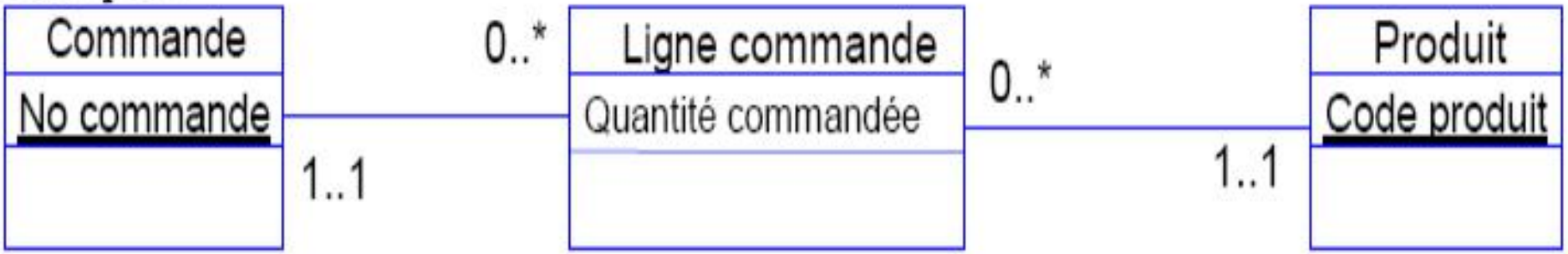
⇒Association », ou classe-asso



Exemple

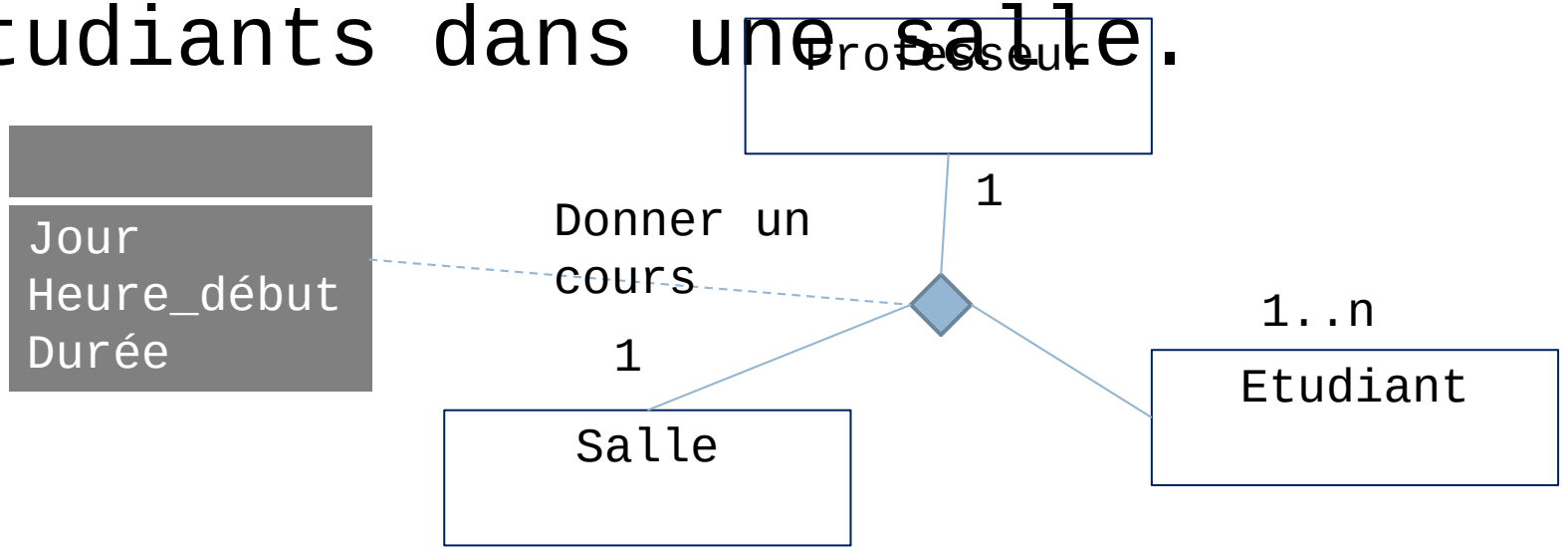


Exemple :

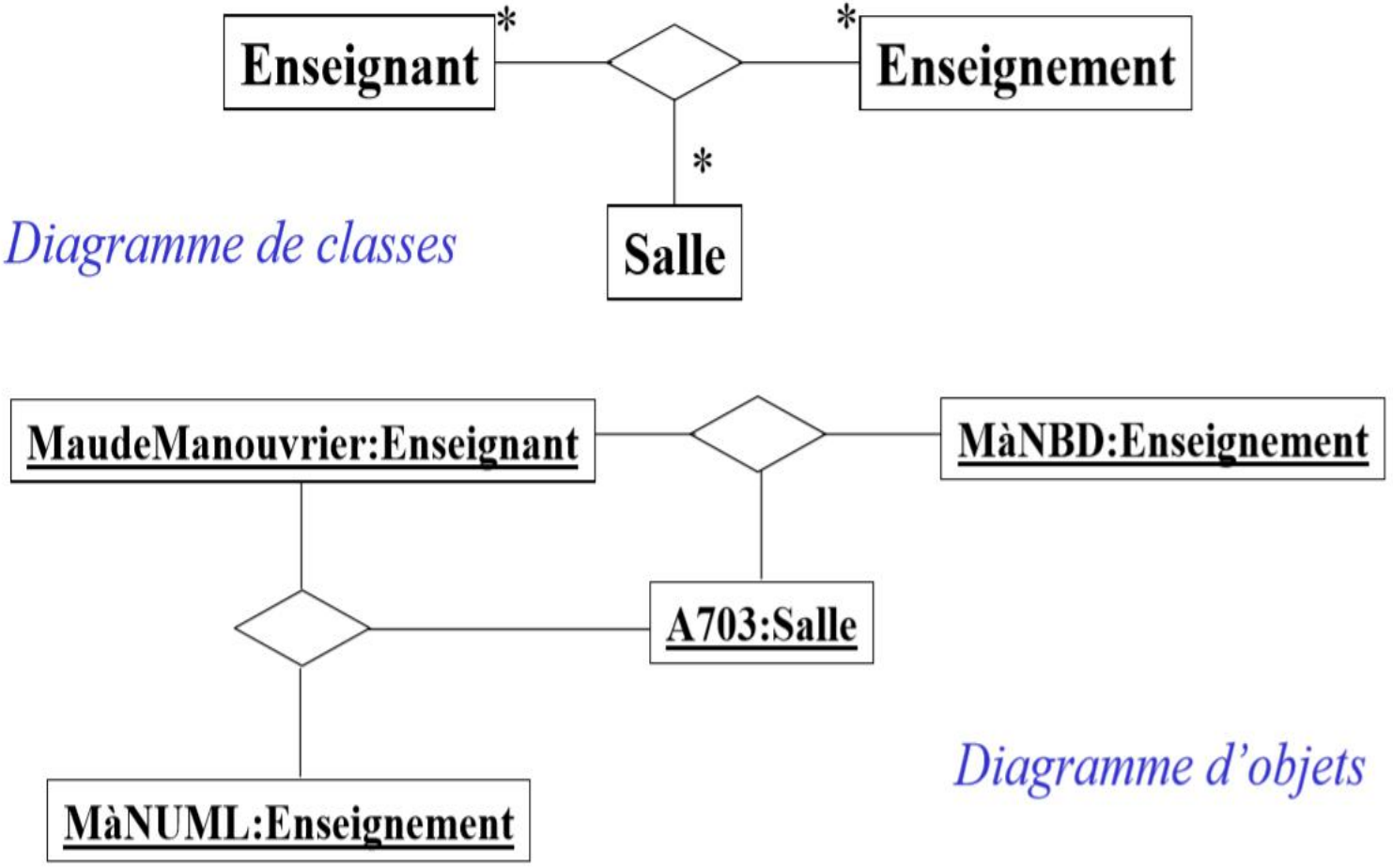


Association ternaire

- Trois classes peuvent être associées ensemble dans une seule relation ternaire.
- Un professeur enseigne plusieurs étudiants dans une salle.



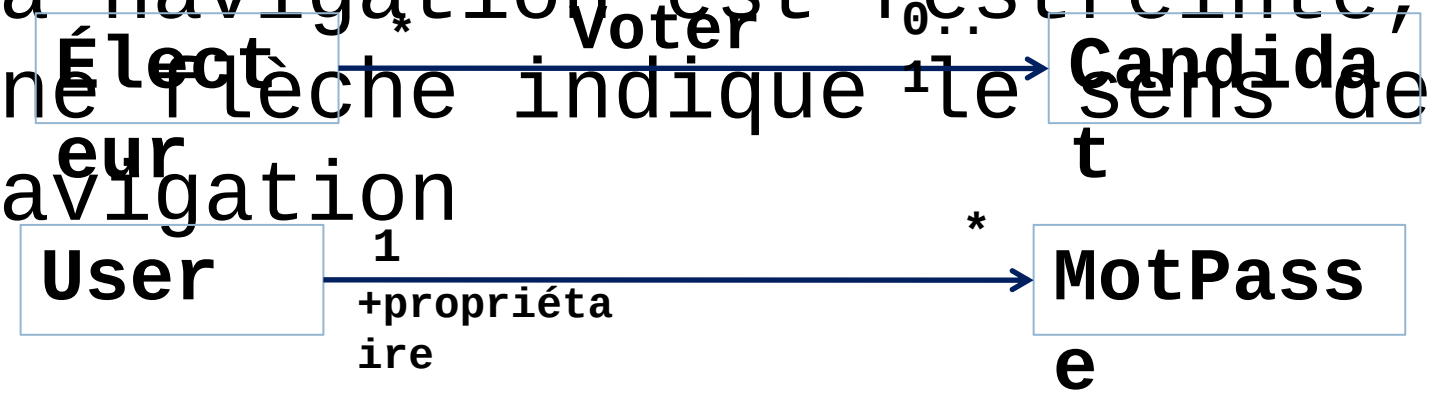
Association n-aire



Association à navigation restreinte

- Bien que les associations soit bi-directionnelles, on peut limiter la navigation à un seul sens

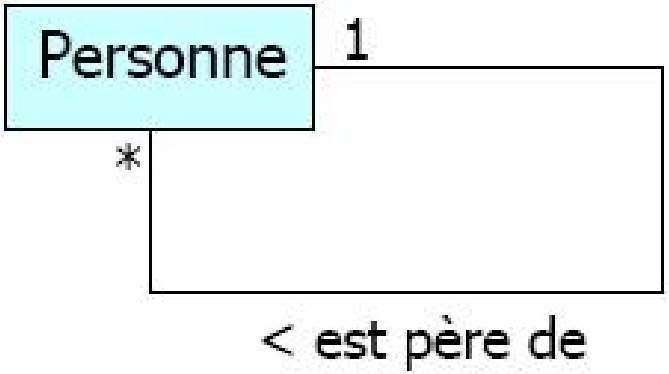
  la navigation est restreinte, une flèche indique le sens de navigation

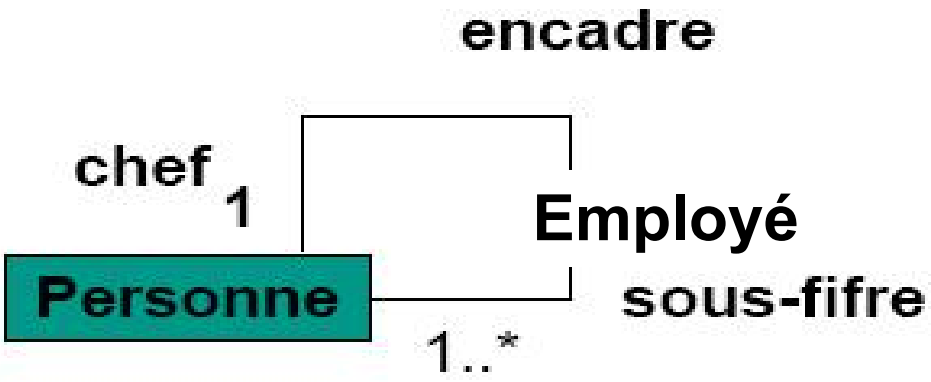


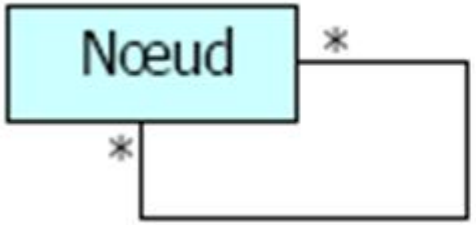
Etant donné un mot de passe, on ne souhaite pas pouvoir accéder à l'utilisateur correspondant

Association réflexive

- lie des objets de même classe







Un réseau informatique est composé de nœuds inter-connectés

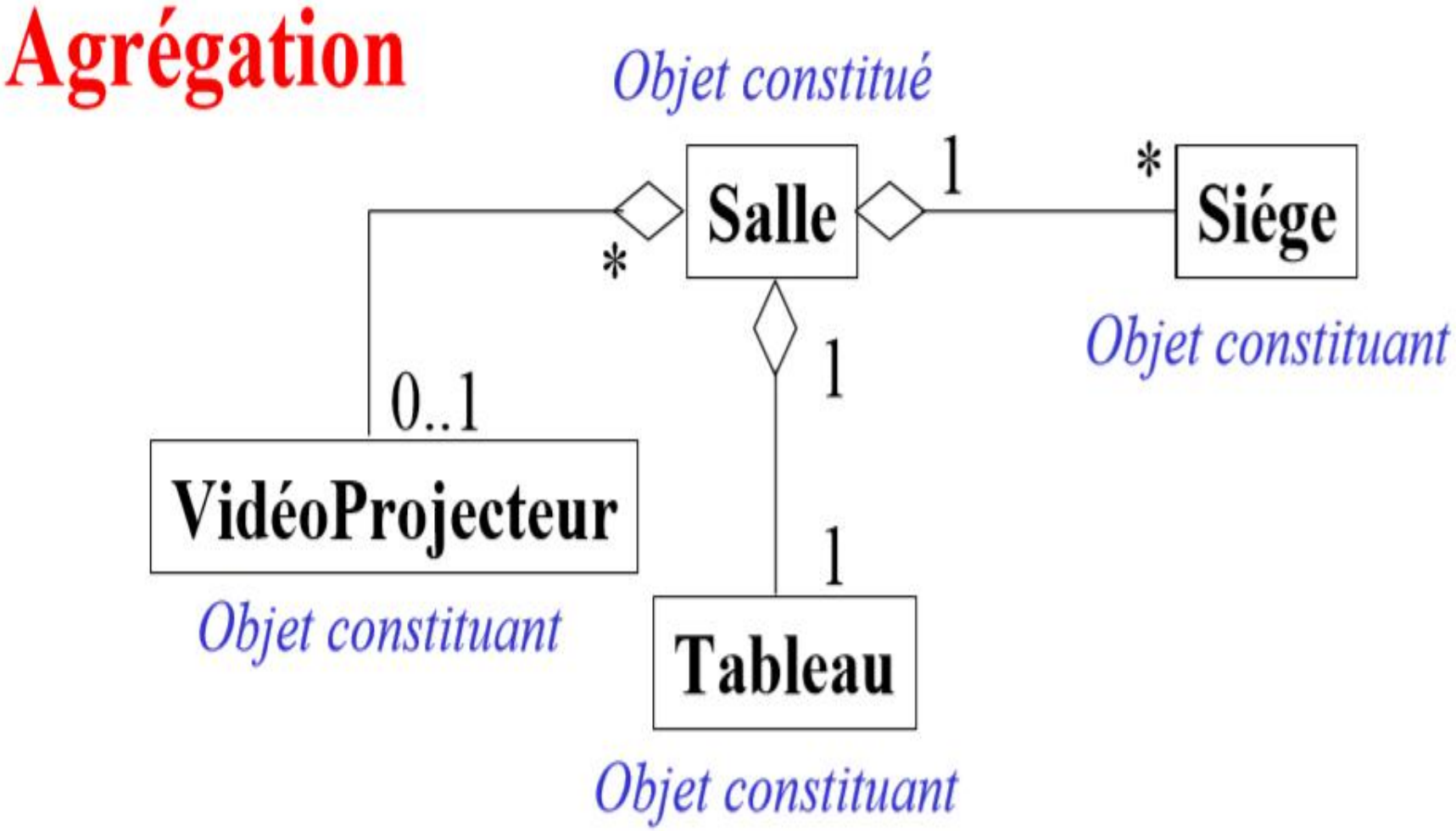
Types d’association

- Il existe plusieurs types d’associations entre classes :
  - Agrégation,
  - Composition,
  - Généralisation/spécialisation,
  - Dépendence, etc.

Agrégation

- Relation entre 2 classes de type "Agrégat/élément", spécifiant que les objets d'une classe sont des composants de l'autre classe.
  - un changement d'état d'une classe, entraîne un changement d'état d'une autre, une action sur une classe, entraîne une action sur une autre
- Une instance de l'élément agrégé peut exister sans Agrégat (et inversement)

Exemple


Agrégation
Objet constitué
Salle
1
*
Siége
*
Objet constituant
1
0..1
VidéoProjecteur
Objet constituant
1
Tableau
Objet constituant

- Exemple 1
  - Une personne est dans une foule
  - Une foule contient plusieurs personnes

- Exemple 2 (Agrégation partagée)
  - Une personne fait partie de plusieurs équipes
  - Une équipe contient plusieurs personnes

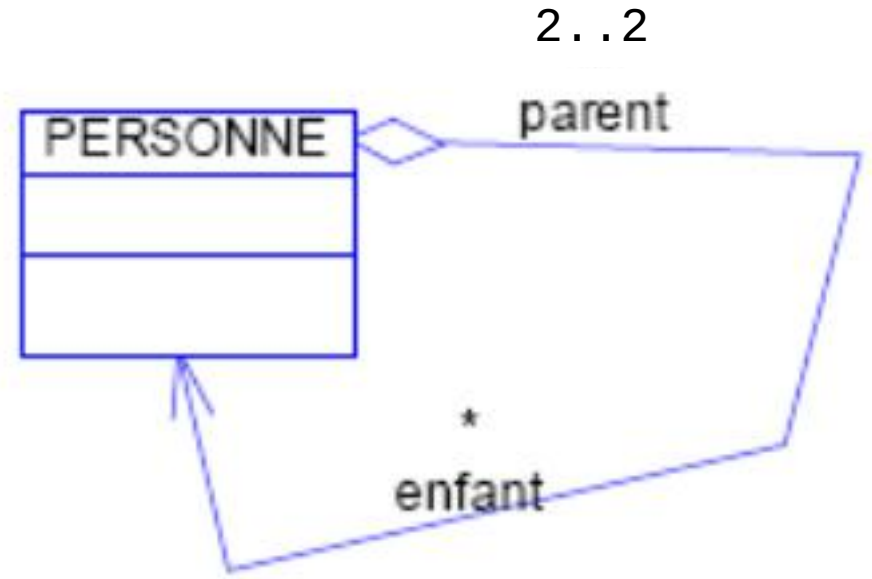
Agrégat multiple

- Classification de véhicules: chaque véhicule est classifié selon son type. Chaque véhicule fait partie d'un et un seul type de véhicule. Un type est associé à

- Agrégat multiple: un véhicule peut appartenir à

Agrégation réflexive

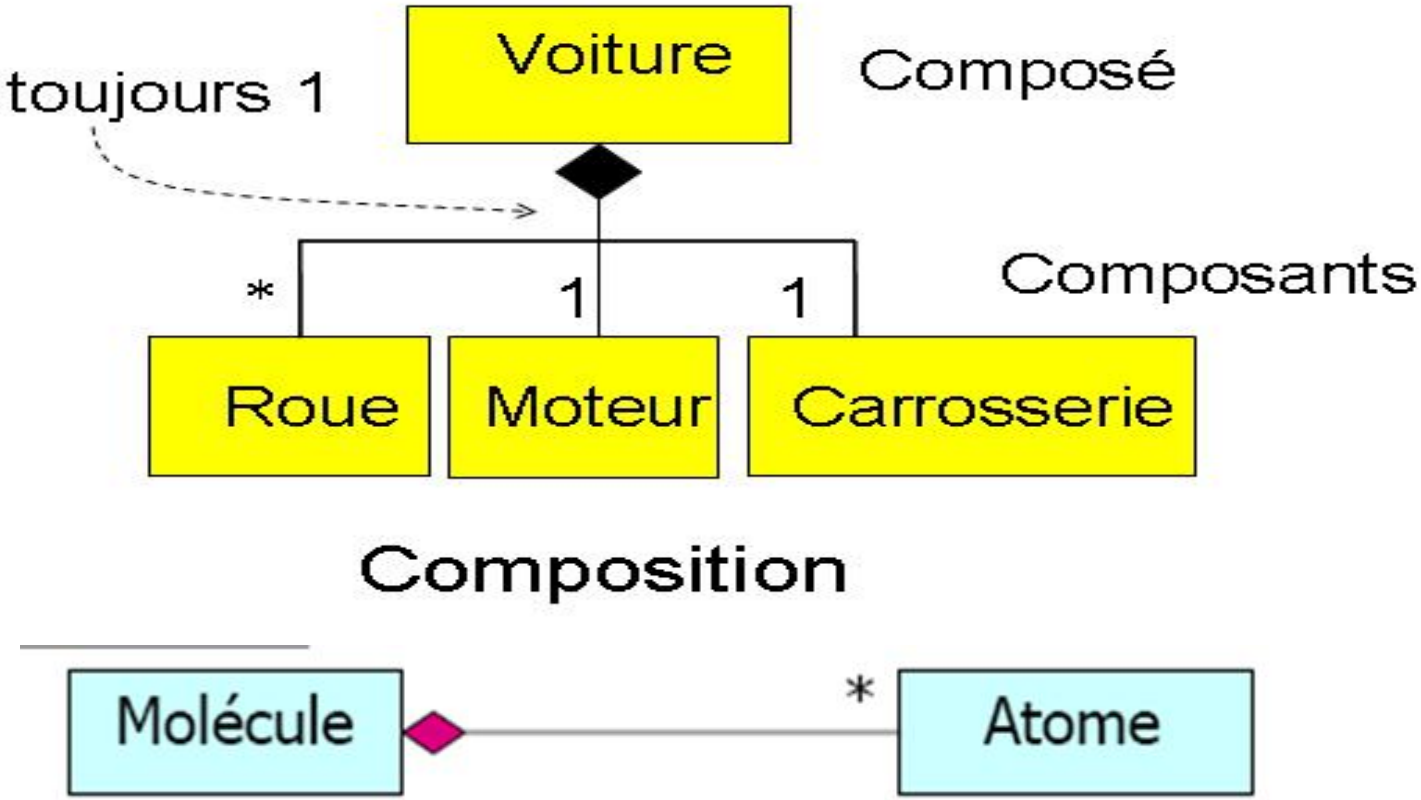
- dès que l'on modélise des relations hiérarchiques ou des liens de parenté par exemple.



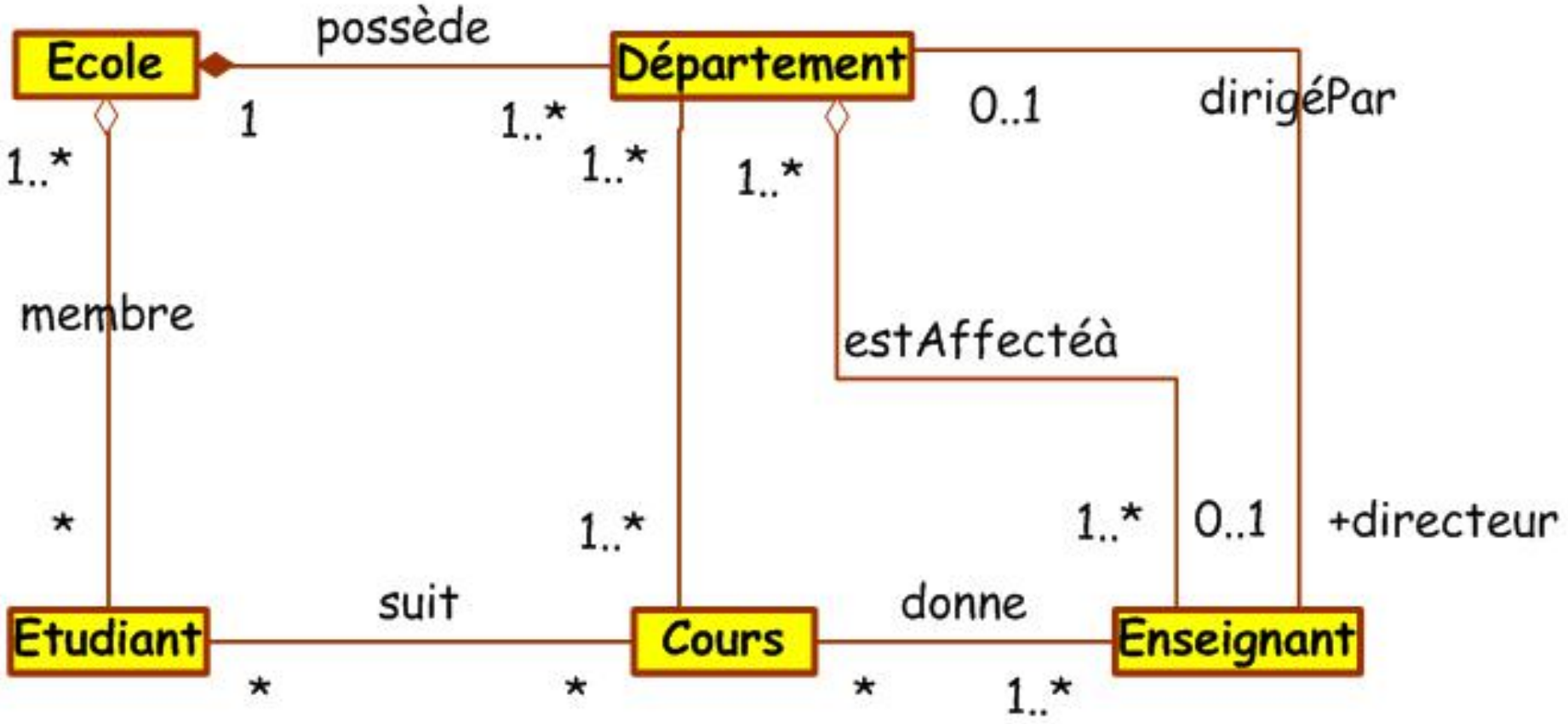
Composition

- Agrégation forte : Les cycles de vies des éléments agrégés et de l'agrégat sont liés : si l'agrégat est détruit (ou copié), ses composants le sont aussi.
  - A un même moment, une instance de composant ne peut être liée qu'à un seul agrégat
  - Le
    instances de classes composées.

Exemple


toujours 1
Voiture
Composé
*
1
1
Composants
Roue
Moteur
Carrosserie
Composition
Molécule
*
Atome

Exemple

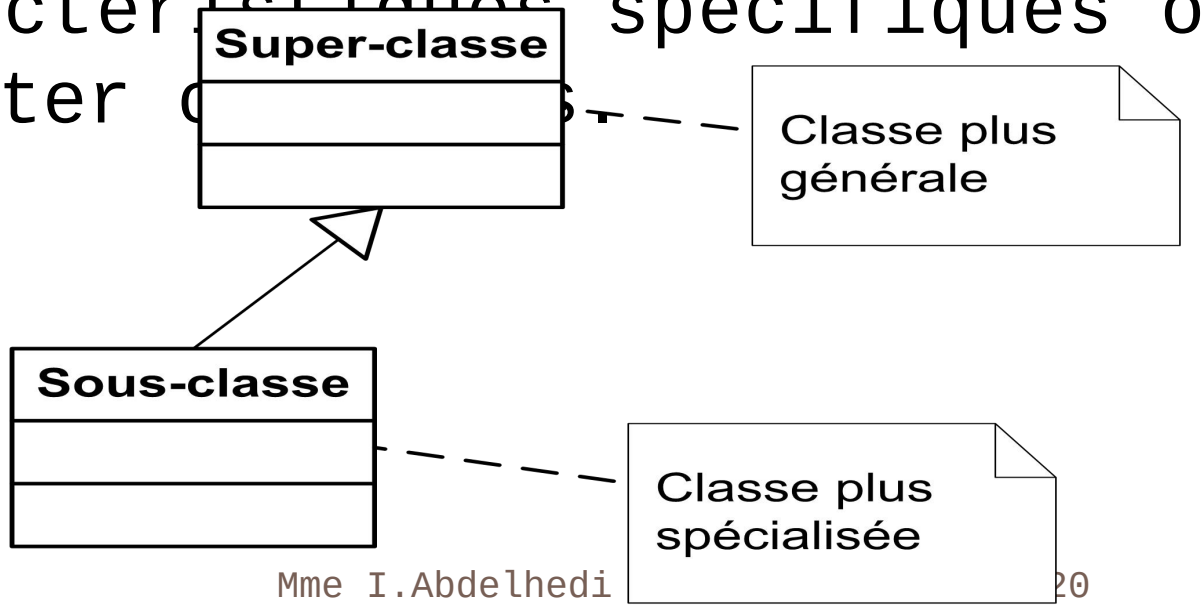

Ecole
possède
Département
1
1..*
1..*
1..*
0..1
dirigéPar
1..*
membre
estAffectéà
*
1..*
1..*
0..1
+directeur
Etudiant
suit
Cours
donne
Enseignant
*
*
*
1..*

Héritage

- Relation entre une super-classe (classe de base) et ses sous-classes (classes dérivées)
  - Permet la transmission des caractéristiques d'une classe (ses attributs et opérations) à une sous classe d'objets.
  - Evite la duplication et encourage la réutilisation du code.
- 2 relations d'héritage : Généralisation et Spécialisation

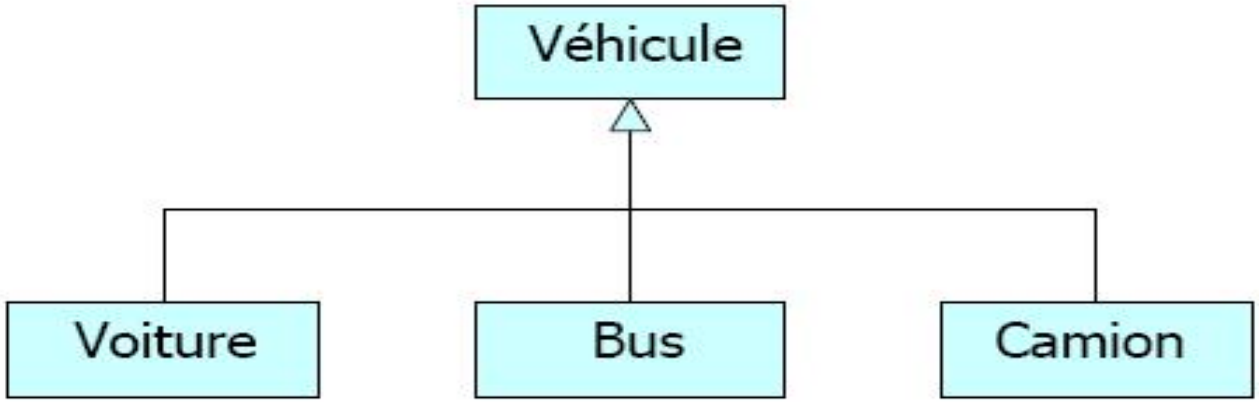
La spécialisation

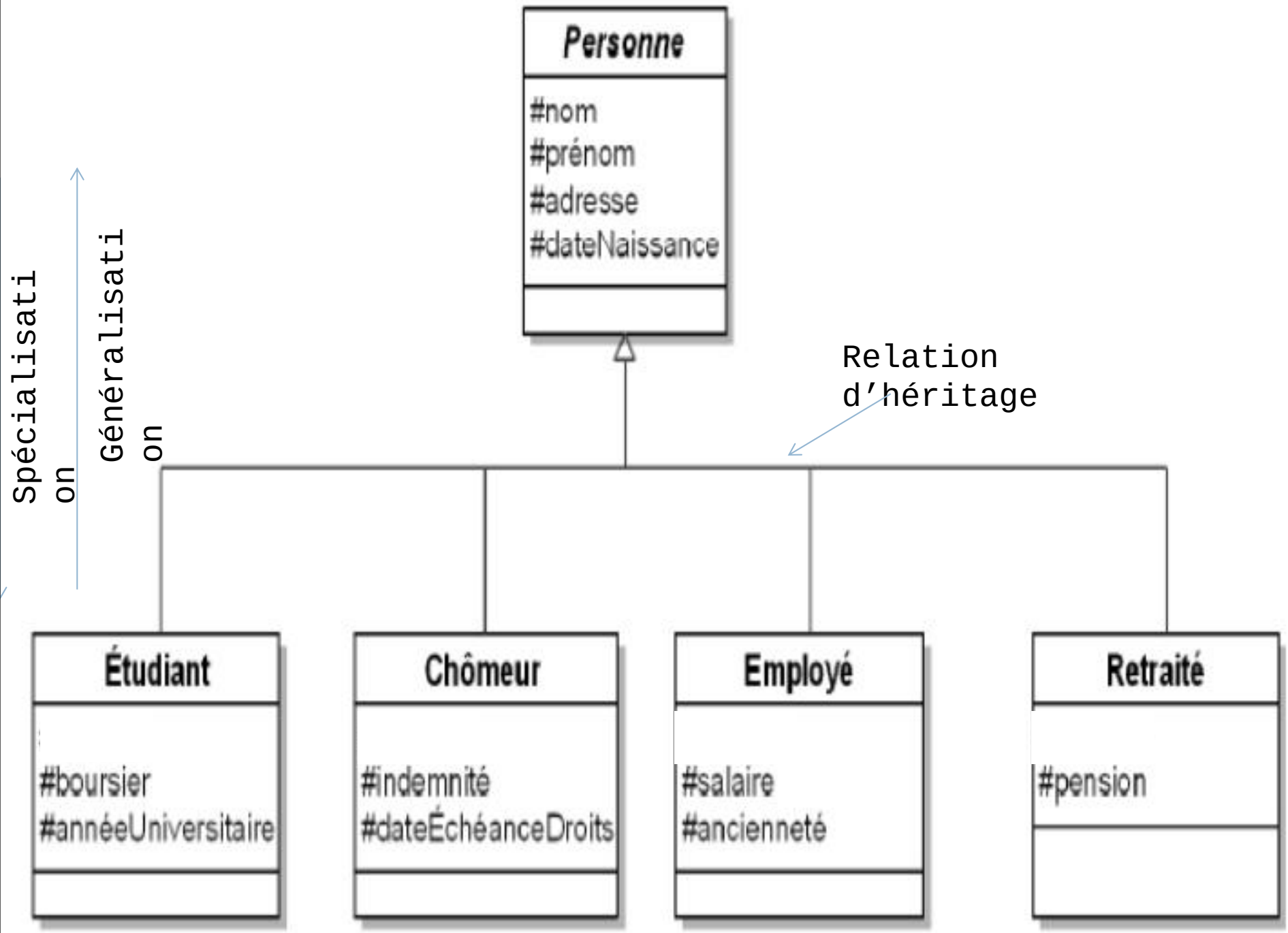
- Démarche descendante
  - Etendre les propriétés d'une classe, sous forme de sous-classes, plus spécifiques afin d'y ajouter des caractéristiques spécifiques ou d'en adapter c



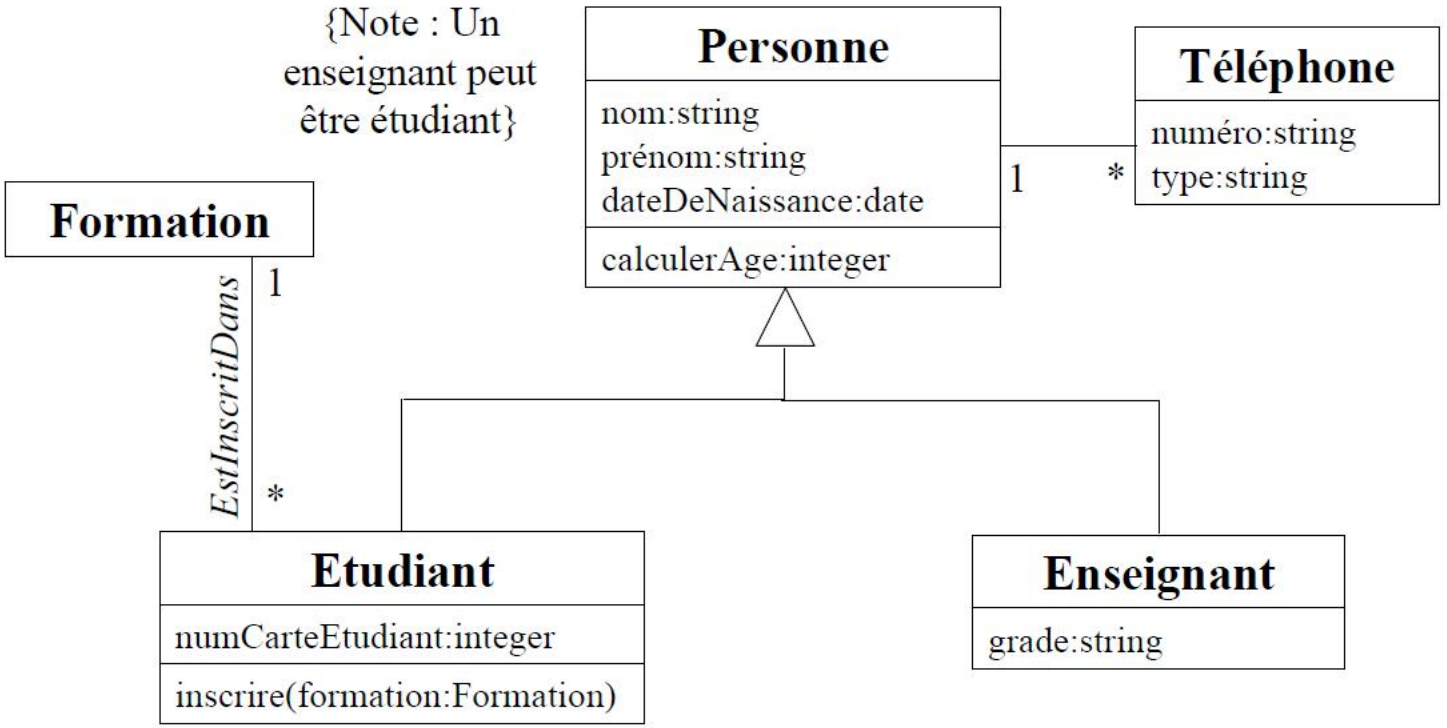
La généralisation

- Démarche ascendante.
  - Factoriser les propriétés d'un ensemble de classes, sous forme d'une super-classe, plus abstraite (permet de gagner en généricité), afin de regrouper les caract un ensemb

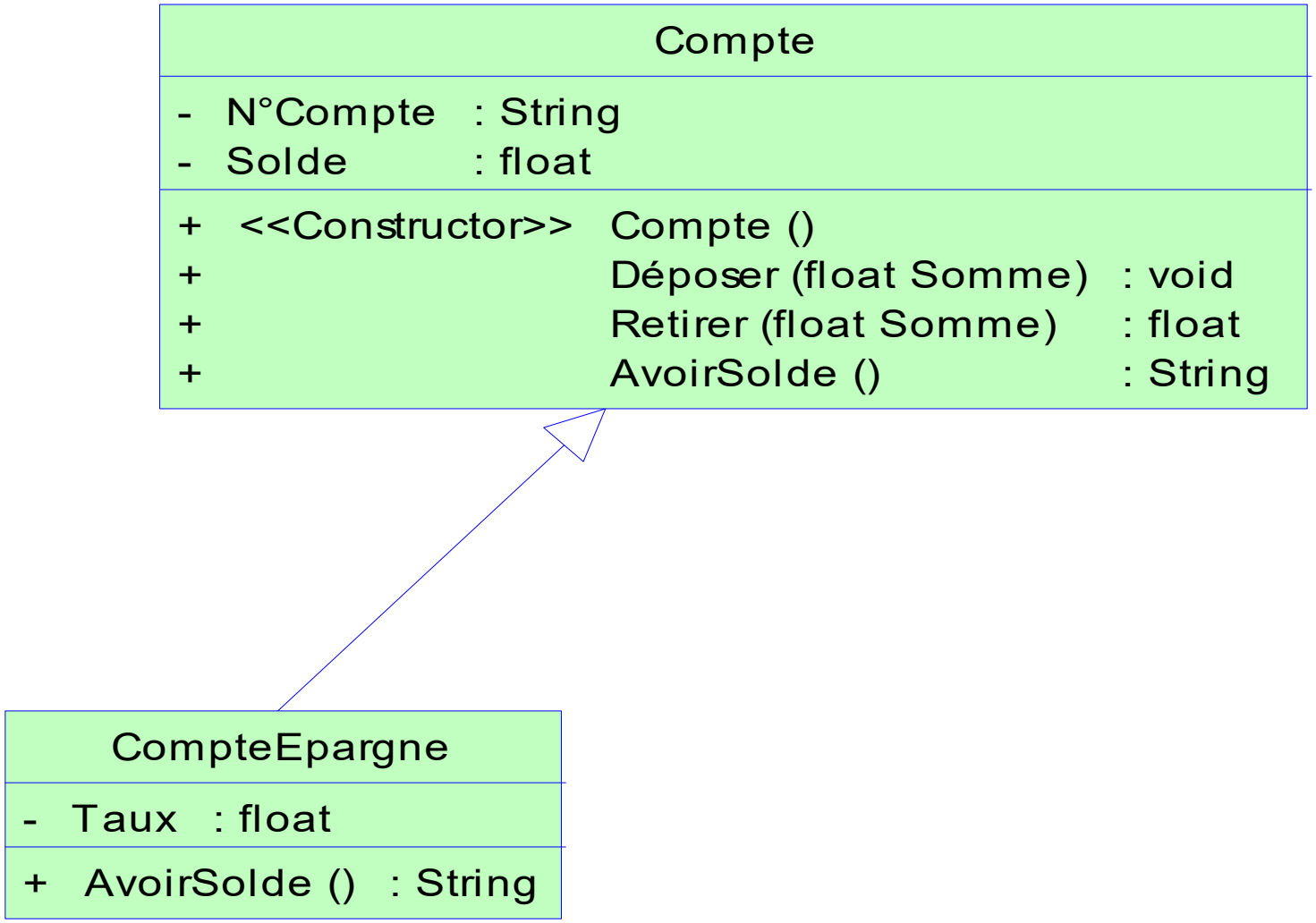



Personne
#nom
#prénom
#adresse
#dateNaissance
Spécialisation
Généralisation
Relation d'héritage
Étudiant
#boursier
#annéeUniversitaire
Chômeur
#indemnité
#dateÉchéanceDroits
Employé
#salaire
#ancienneté
Retraité
#pension

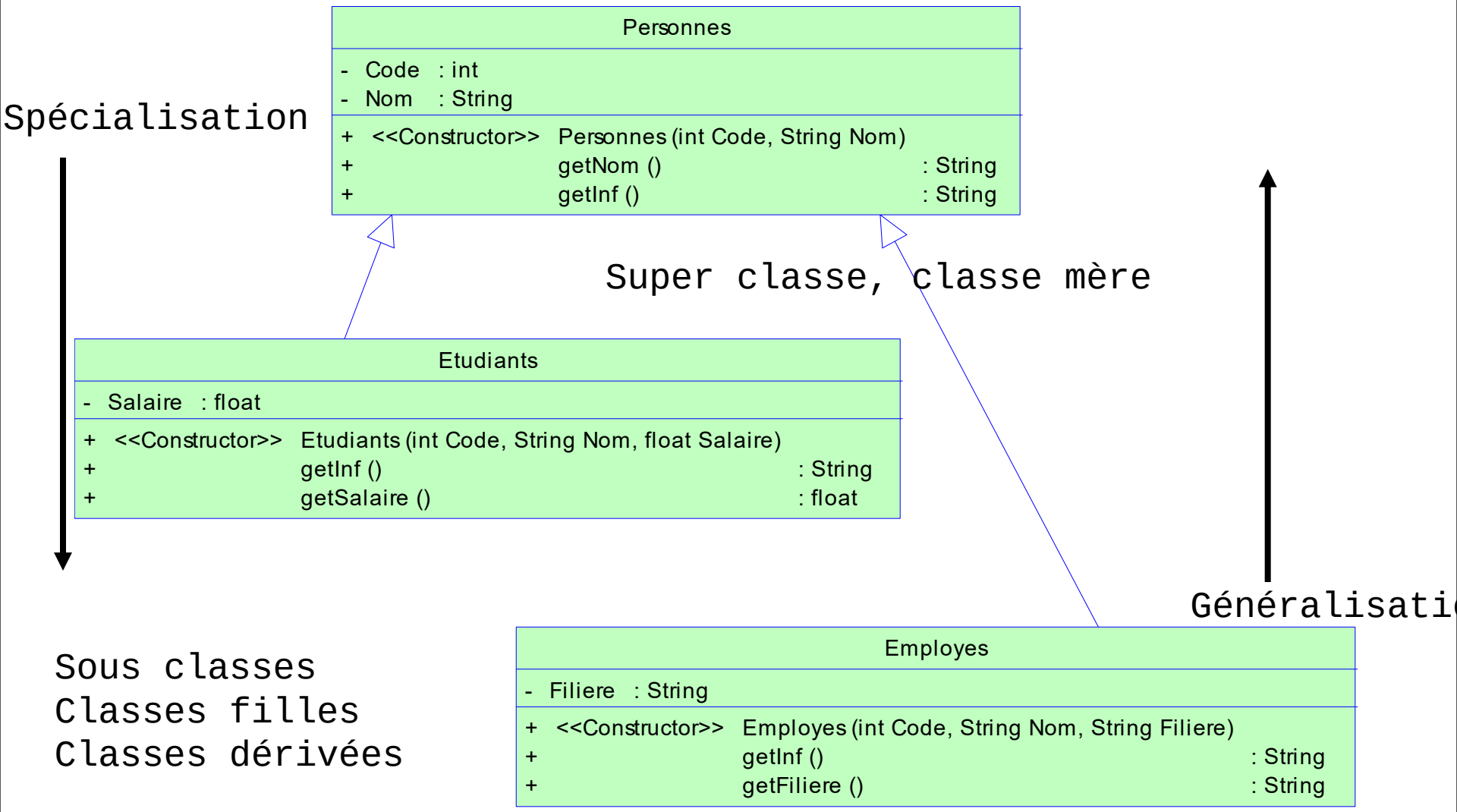
Exemple


{Note : Un enseignant peut être étudiant}
Personne
nom:string
prénom:string
dateDeNaissance:date
calculerAge:integer
Téléphone
numéro:string
type:string
1
*
Formation
1
EstInscritDans
*
Etudiant
numCarteEtudiant:integer
inscrire(formation:Formation)
Enseignant
grade:string

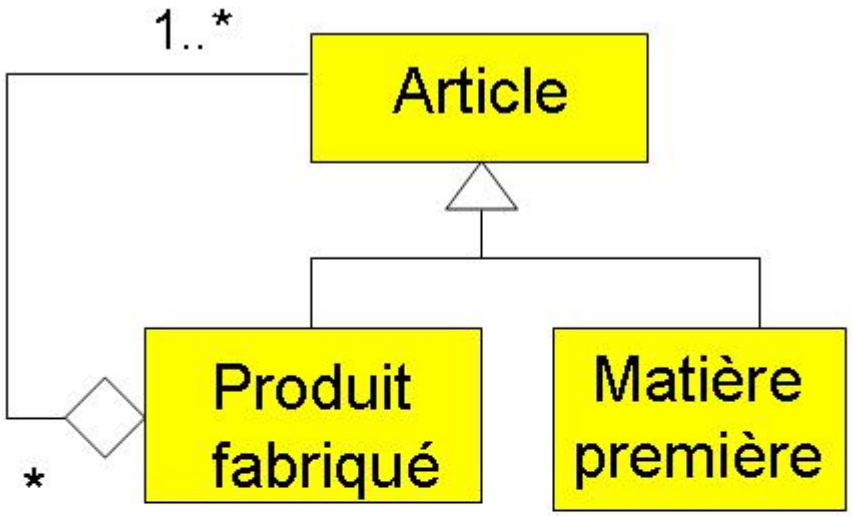
Généralisation / Spécialisation et héritage



Exemple


Personnes
- Code : int
- Nom : String
+ <<Constructor>> Personnes (int Code, String Nom)
+ getNom () : String
+ getInf () : String
Spécialisation
Super classe, classe mère
Etudiants
- Salaire : float
+ <<Constructor>> Etudiants (int Code, String Nom, float Salaire)
+ getInf () : String
+ getSalaire () : float
Généralisati
Employes
- Filiere : String
+ <<Constructor>> Employes (int Code, String Nom, String Filiere)
+ getInf () : String
+ getFiliere () : String
Sous classes
Classes filles
Classes dérivées

Exemple

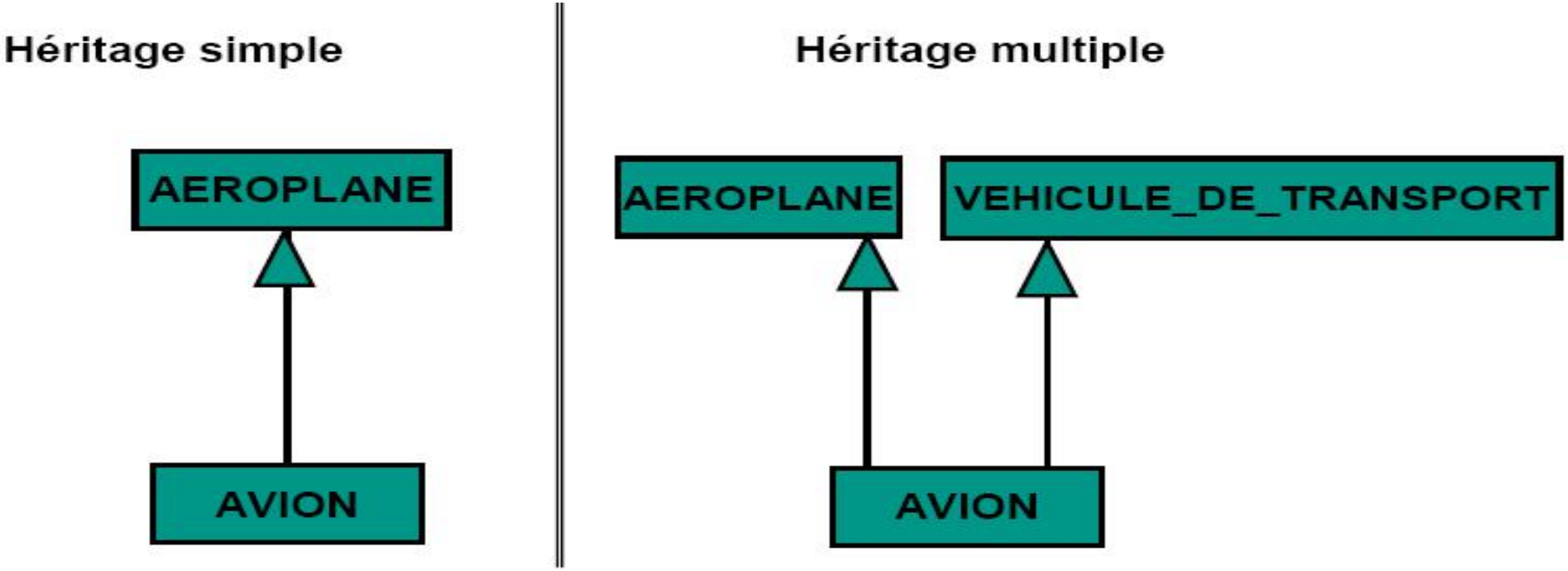


(design pattern « Composite »)

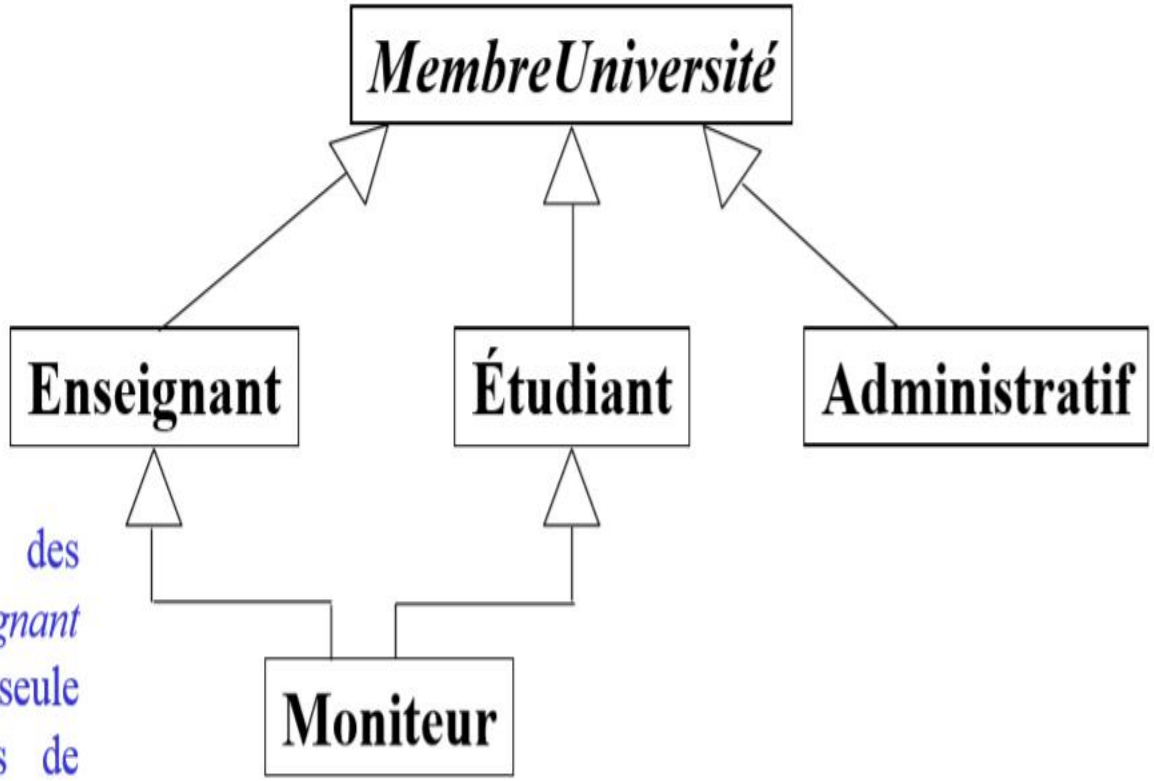
Un article est acheté (matière première) ou fabriqué à partir d'autres articles et/ou matières premières.

Héritage multiple

- Les classes peuvent avoir plusieurs superclasses ;
- Plusieurs flèches partent de la sous-classe vers les différentes superclasses. La généralisation



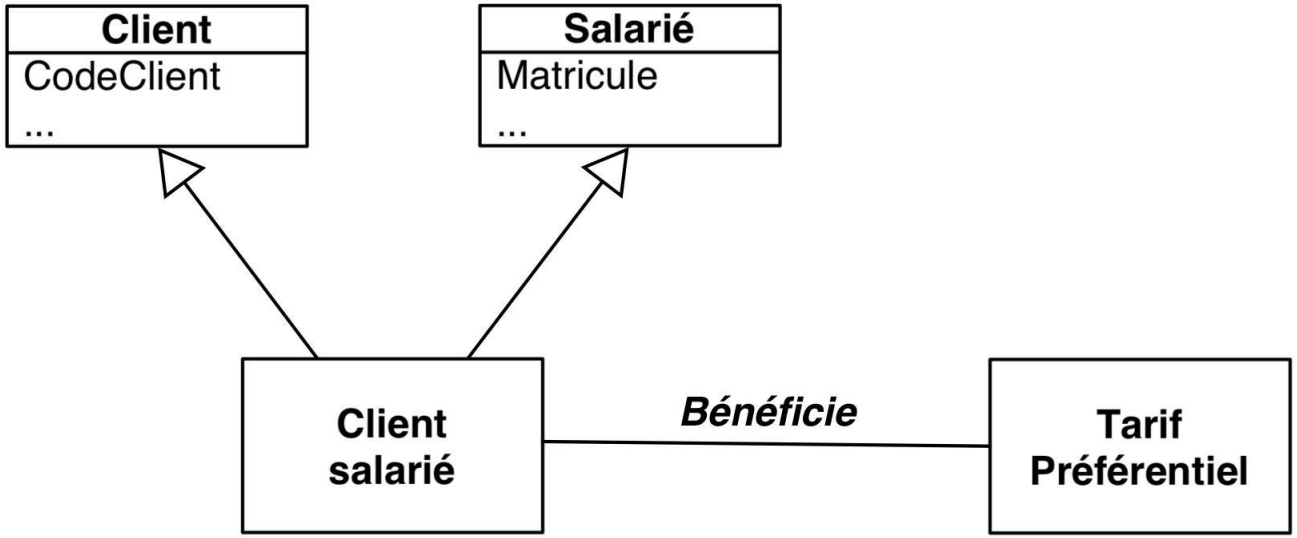
Exemple



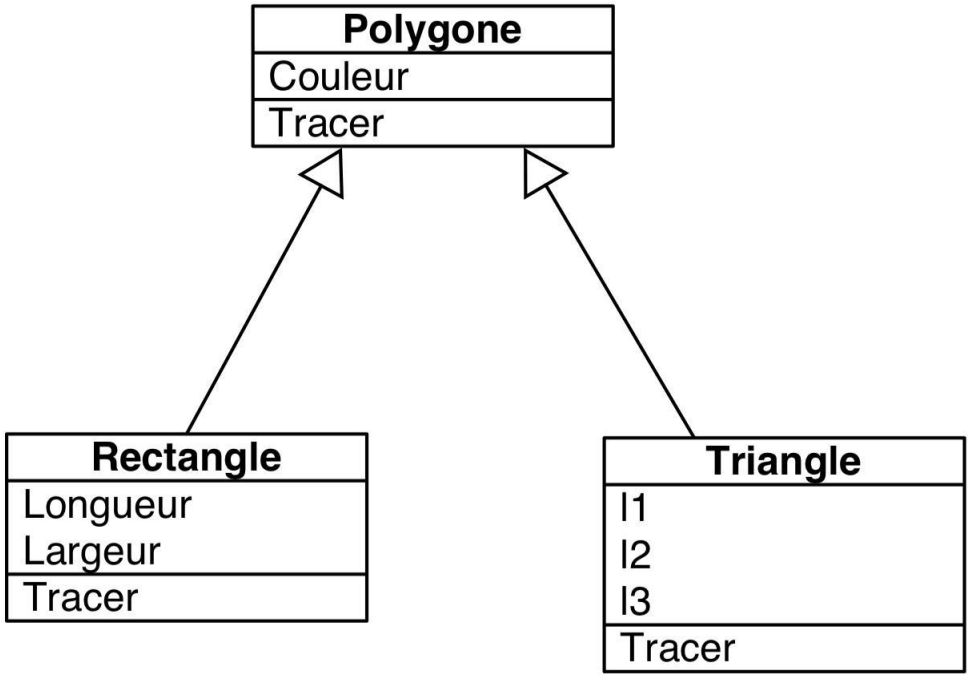
Moniteur hérite des propriétés de *Enseignant* et d'*Etudiant* et une seule fois des propriétés de *MembreUniversité*.

Héritage multiple

La classe « client spécial » est une spécialisation de client et de salarié, qui bénéficie des tarifs préférentiels aux salariés.



Généralisation / Spécialisation

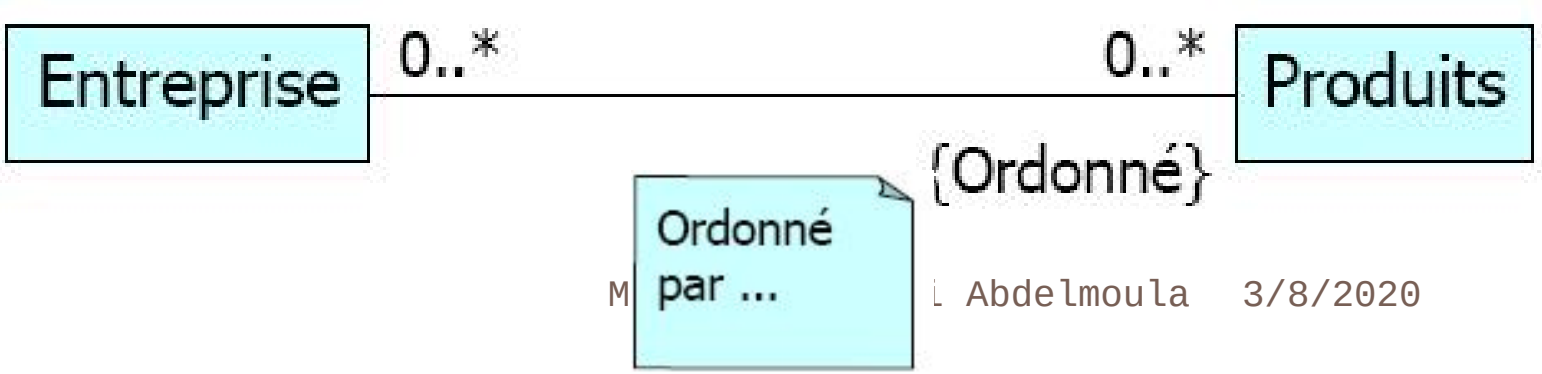


- **polymorphisme** = opérations de même nom, comportement spécifique

Contraintes sur les associations

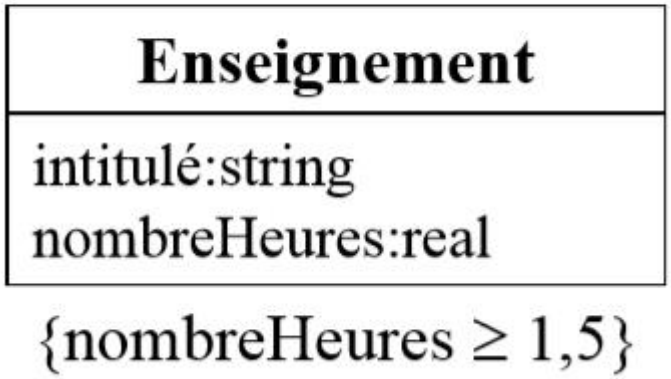
Association ordonnée

- Stéréotype {Ordonné} du côté de la classe dont les instances sont ordonnés : exprimer que les objets sont ordonnés (selon la clé, le nom, la date, etc.)
  - Le modèle ne spécifie pas comment les objets sont ordonnés => On utilise un commentaire



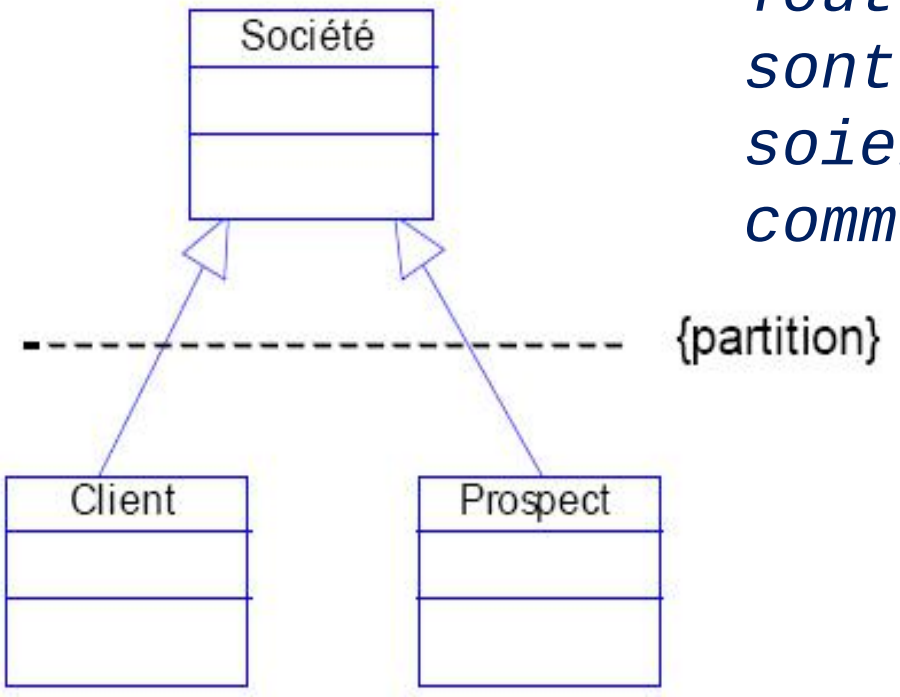
Contrainte s'appliquant sur tous les objets de la classe

Contrainte sur les objets :

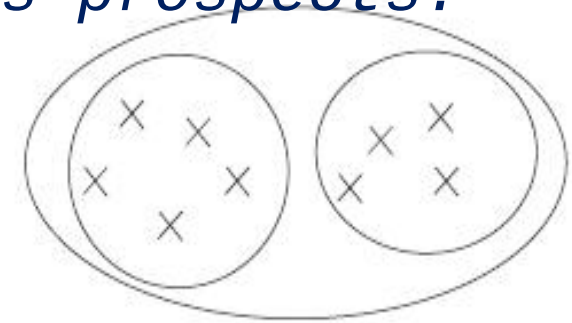


La partition

- Indique que toutes les instances d'une classe correspondent à une et une seule instance des classes liées.

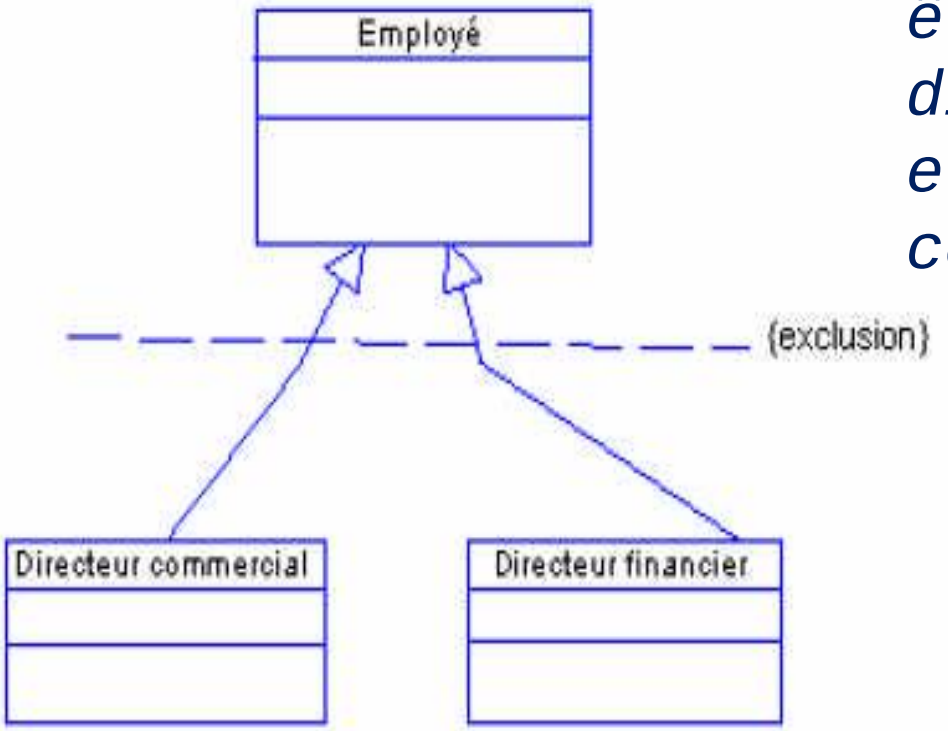


Toutes les sociétés sont soient clientes, soient considérées comme des prospects.

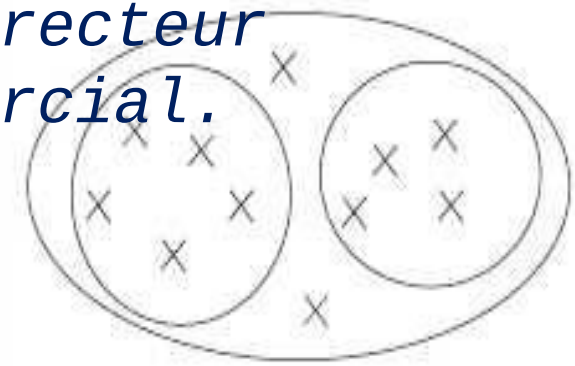


L'exclusion

- Précise qu'une instance d'association exclut une autre



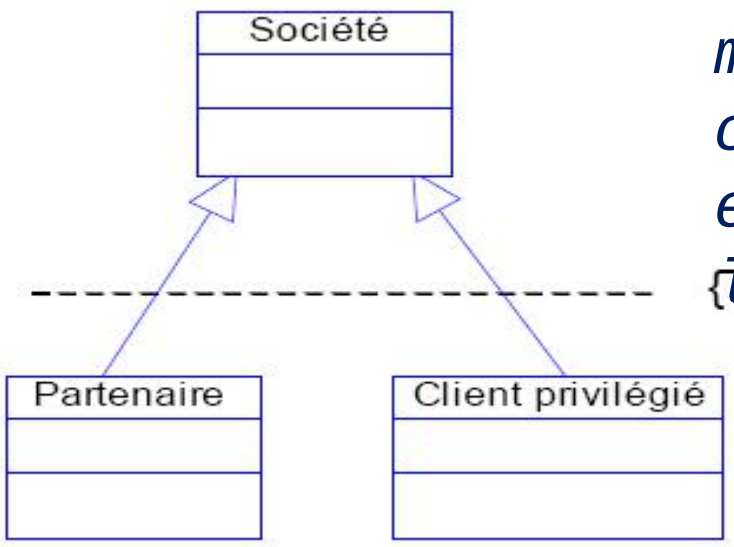
Un employé ne peut être à la fois directeur financier et directeur commercial.



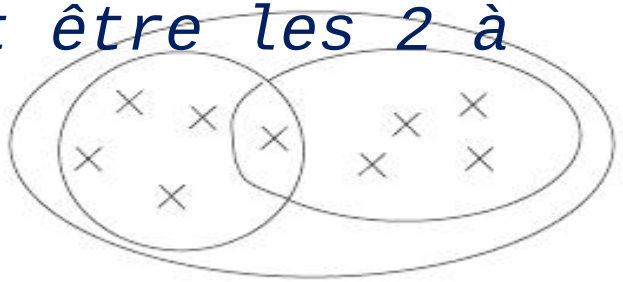
Mais tout employé n'est pas directeur commercial ou

La totalité

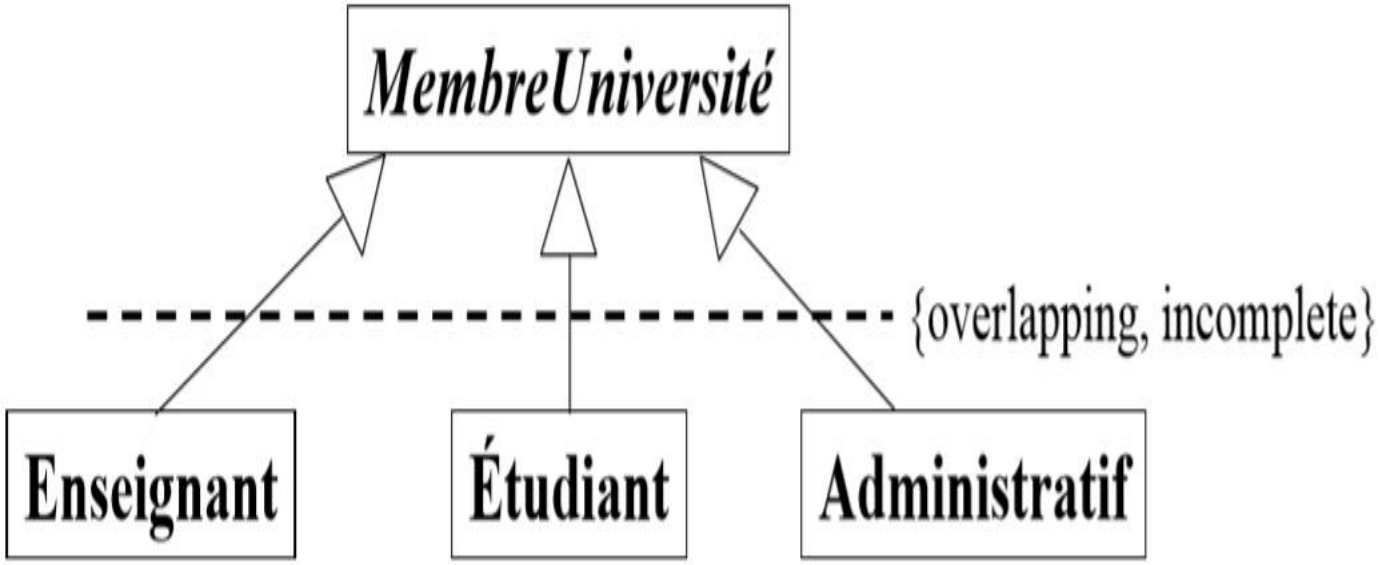
- Toutes les instances d'une classe correspondent au moins à une des instances des classes liées.



Toute société est au moins partenaire ou client privilégiée. Et elle peut être les 2 à la fois.

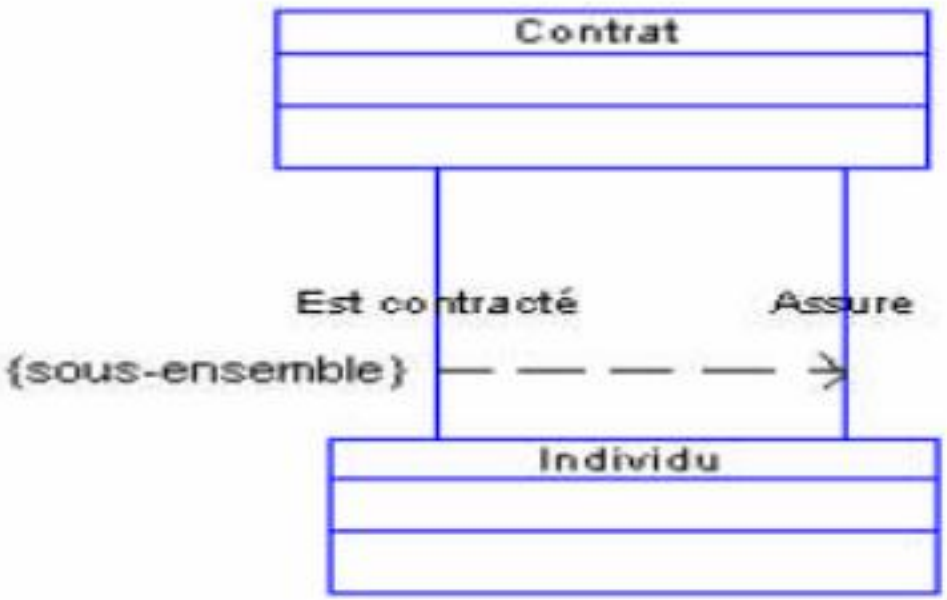


Contrainte incomplète



L'inclusion

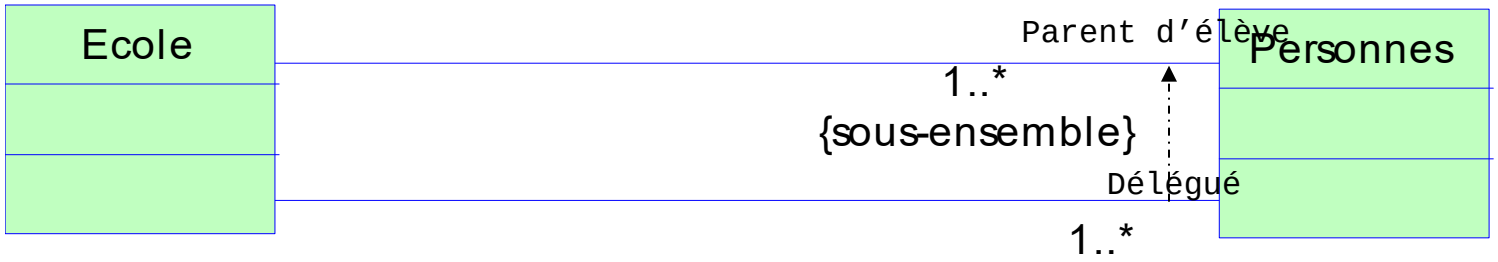
- Permet de préciser qu'une collection est incluse dans une autre collection.



la flèche de la relation de dépendance indique le sens de la contrainte.

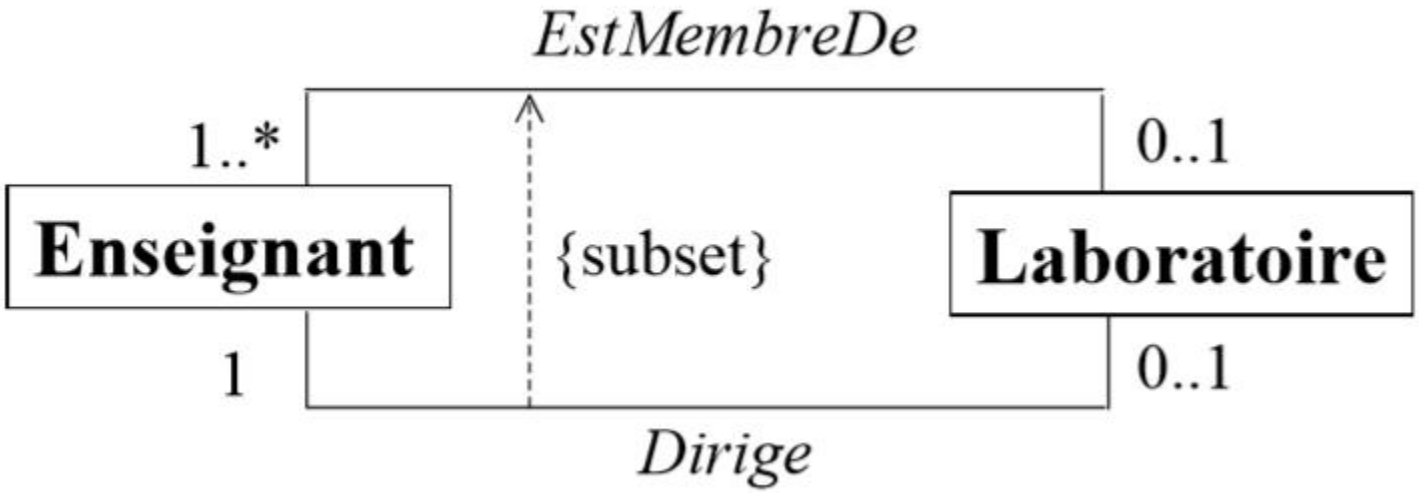
Le contractant d'un contrat fait obligatoirement partie des individus assurés.

Exemples d'inclusion {sous-ensemble}



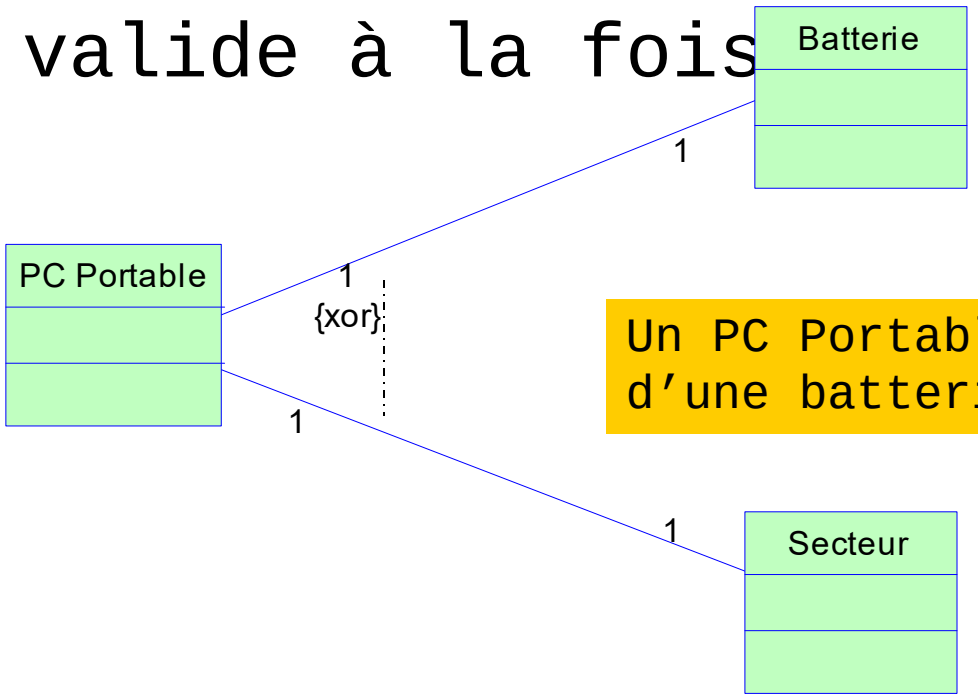
Les personnes qui jouent le rôle de délégué font partie des qui jouent le rôle de parents d'élèves

Exemple d'inclusion



Contrainte {xor}

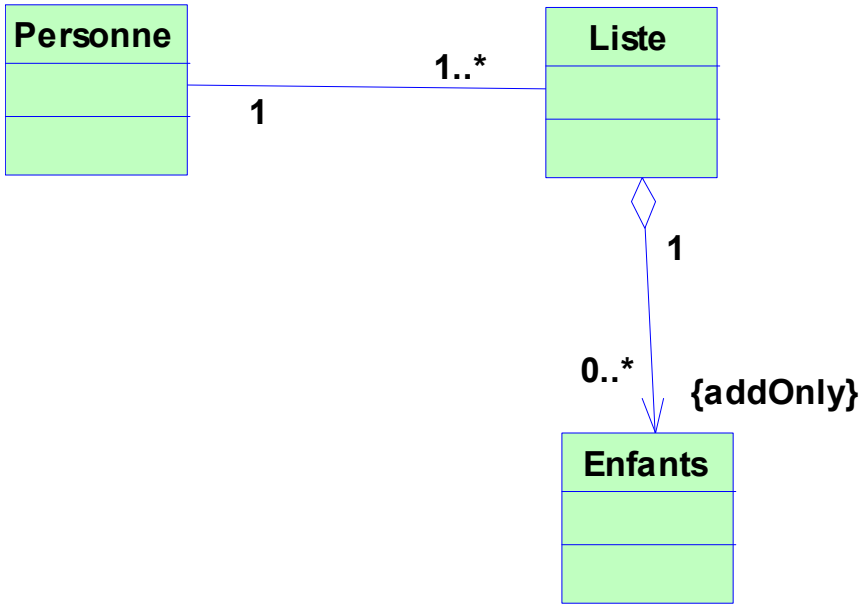
Indique que parmi un groupe d'associations, une seule est valide à la fois



Un PC Portable est alimenté soit à par
d'une batterie, soit à partir d'un sec

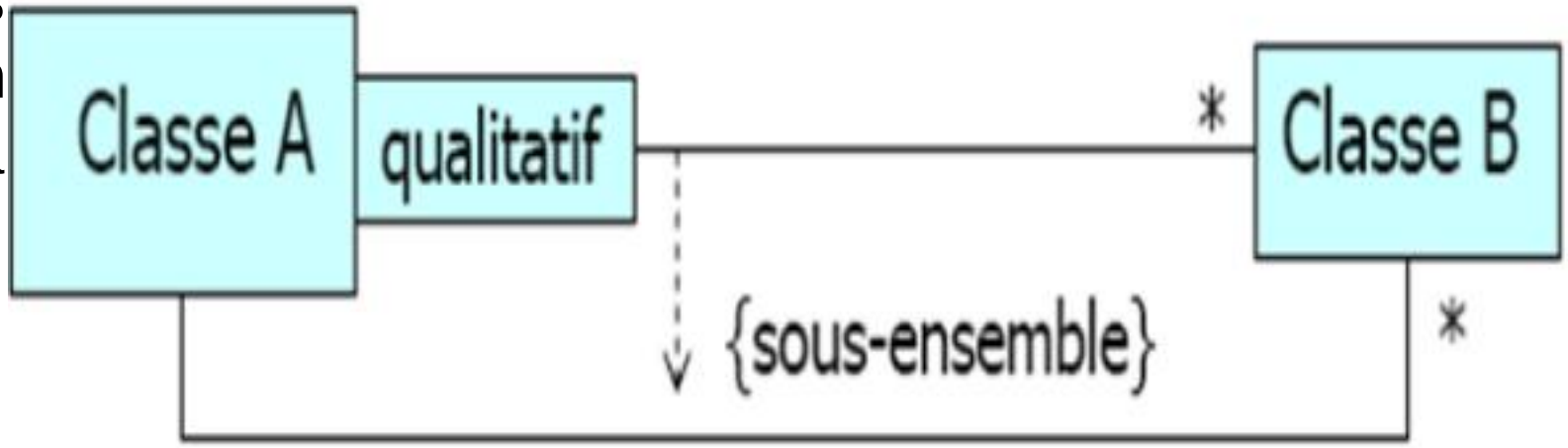
Contrainte {addOnly}

La contrainte prédéfinie {addOnly} autorise l'ajout de nouveaux objets, mais pas leur suppression ni leur mise à jour.

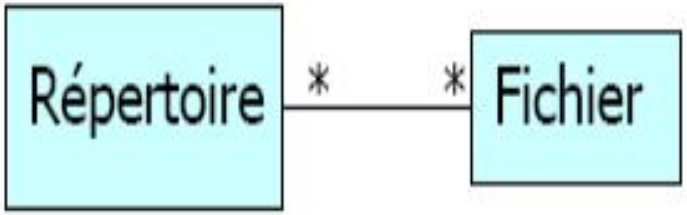


Association qualifiée

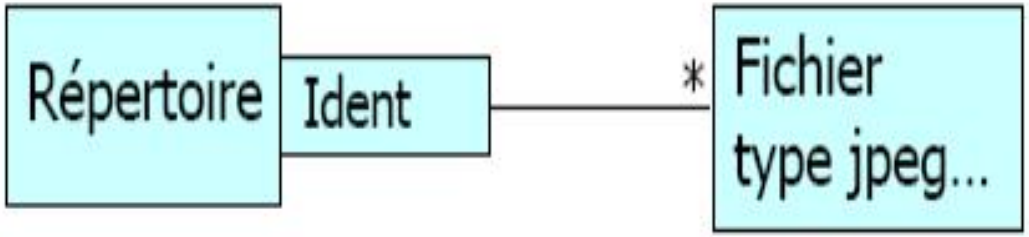
- **Qualificateur :** Attribut permettant de distinguer un sous-ensemble d'objets parmi l'ensemble des objets qui participent à une association en réduisant la multiplicité « plusieurs » à « un »
- **Ass**
  con
  qua



Exemple

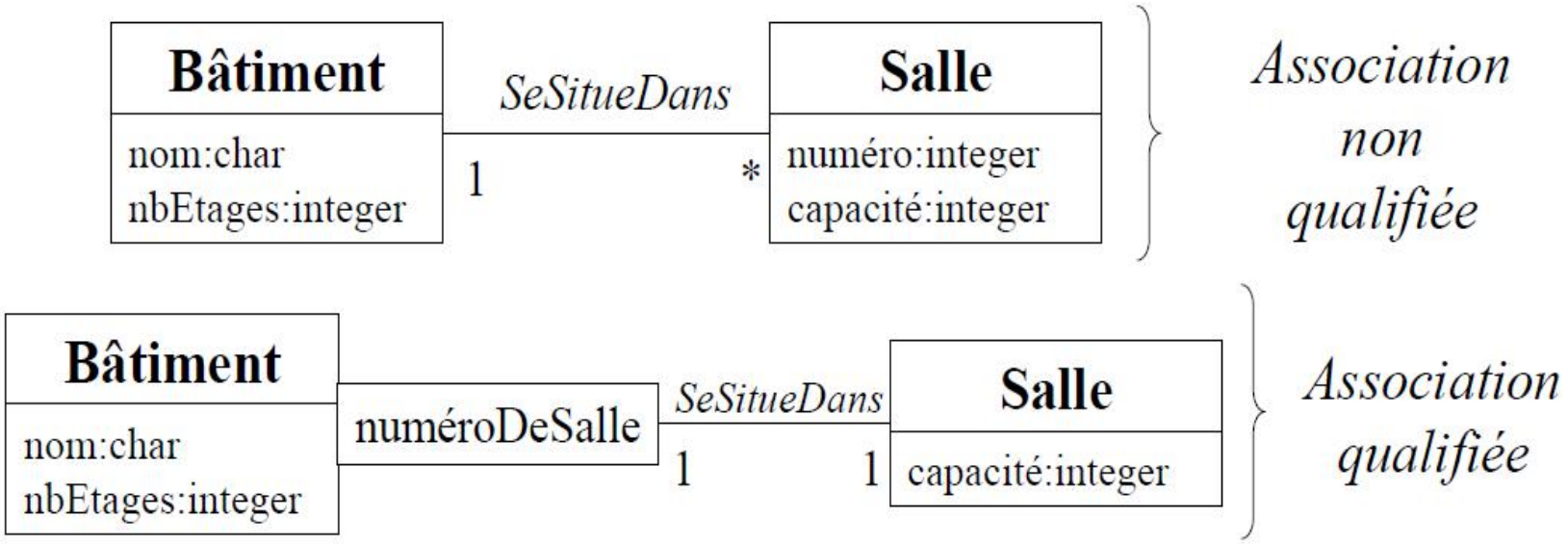


Un répertoire contient 0 ou plusieurs fichiers. Un fichier peut exister dans 0 ou plusieurs répertoires



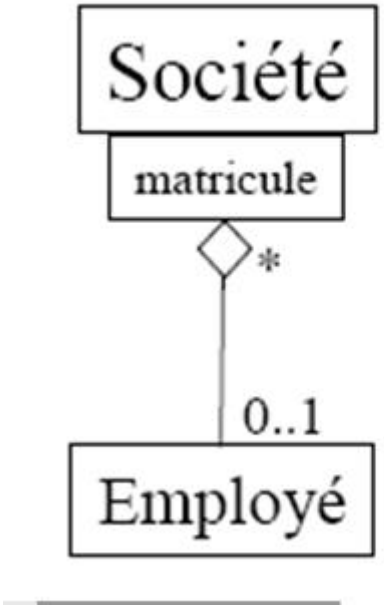
Chaque fichier est identifié
par un identificateur dans
le répertoire

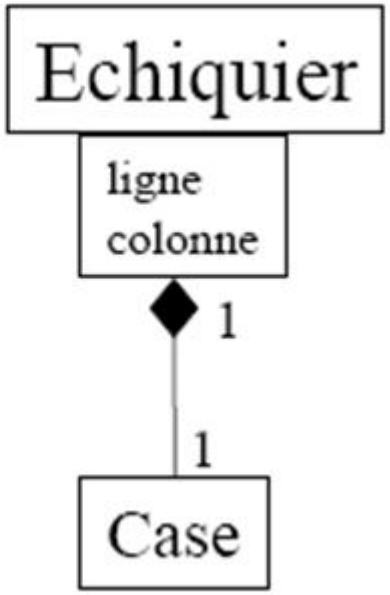
Exemples



Un numéro de Salle permet d'identifier une salle unique dans un Bâtiment donné
Un numéro de Salle est relatif à un Bâtiment

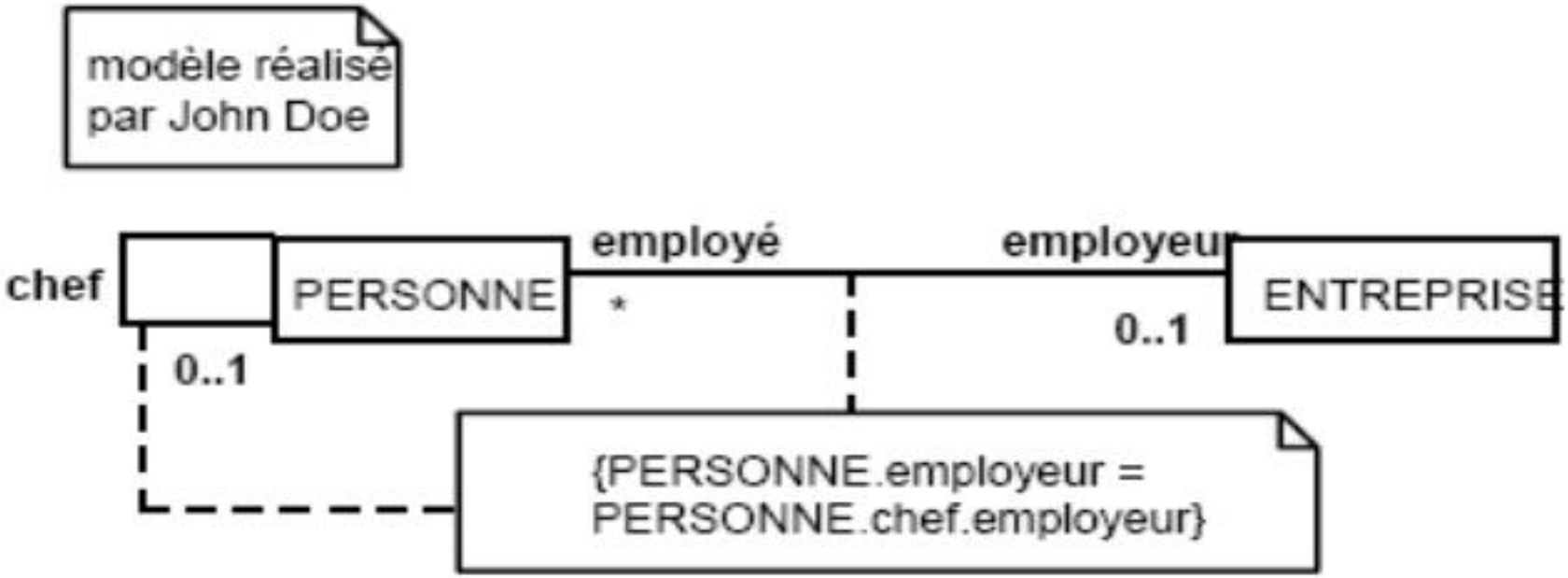
Exemple


Société
matricule
*
0..1
Employé


Echiquier
ligne
colonne
1
1
Case

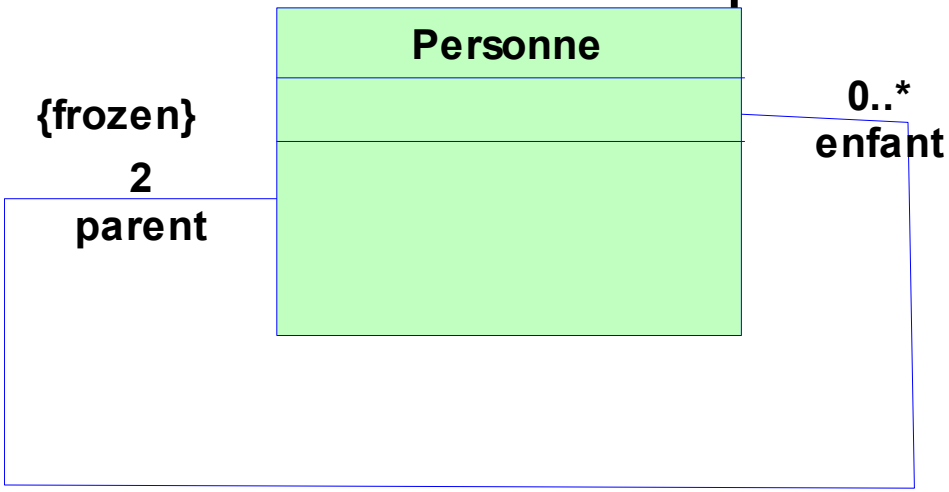
Les notes de commentaire

- Compléments de modélisation, attachés à un élément du modèle ou libre dans un diagramme



contrainte {frozen}

La contrainte prédéfinie {frozen} interdit l'ajout, la suppression ou la mise à jour des liens d'un objet vers les objets de la classe associée, après l'initialisation du premier.



Classe abstraite

Deux cas sont possibles:

1. elle définit au moins une opération abstraite
2. Une classe parent contenant une opération abstraite.
   - *Opération abstraite : lorsqu'on connaît sa déclaration mais pas la manière dont elle peut être réalisée*

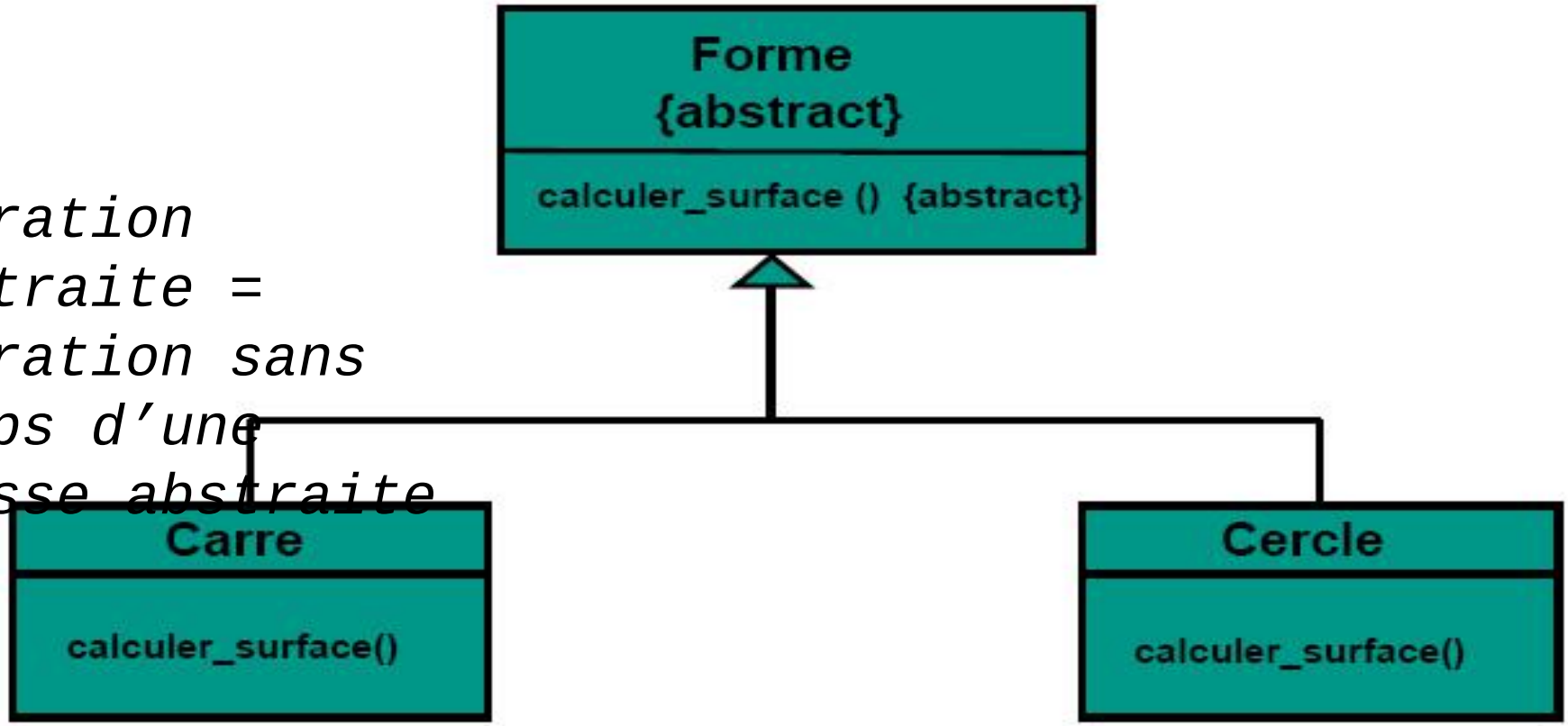
Classe abstraite

- **On ne peut pas instancier une classe abstraite** : elle est vouée à se spécialiser. Une classe abstraite peut contenir des opérations concrètes.
- **Une classe abstraite pure** ne comporte que des opérations abstraites => **Interface**.

Exemple

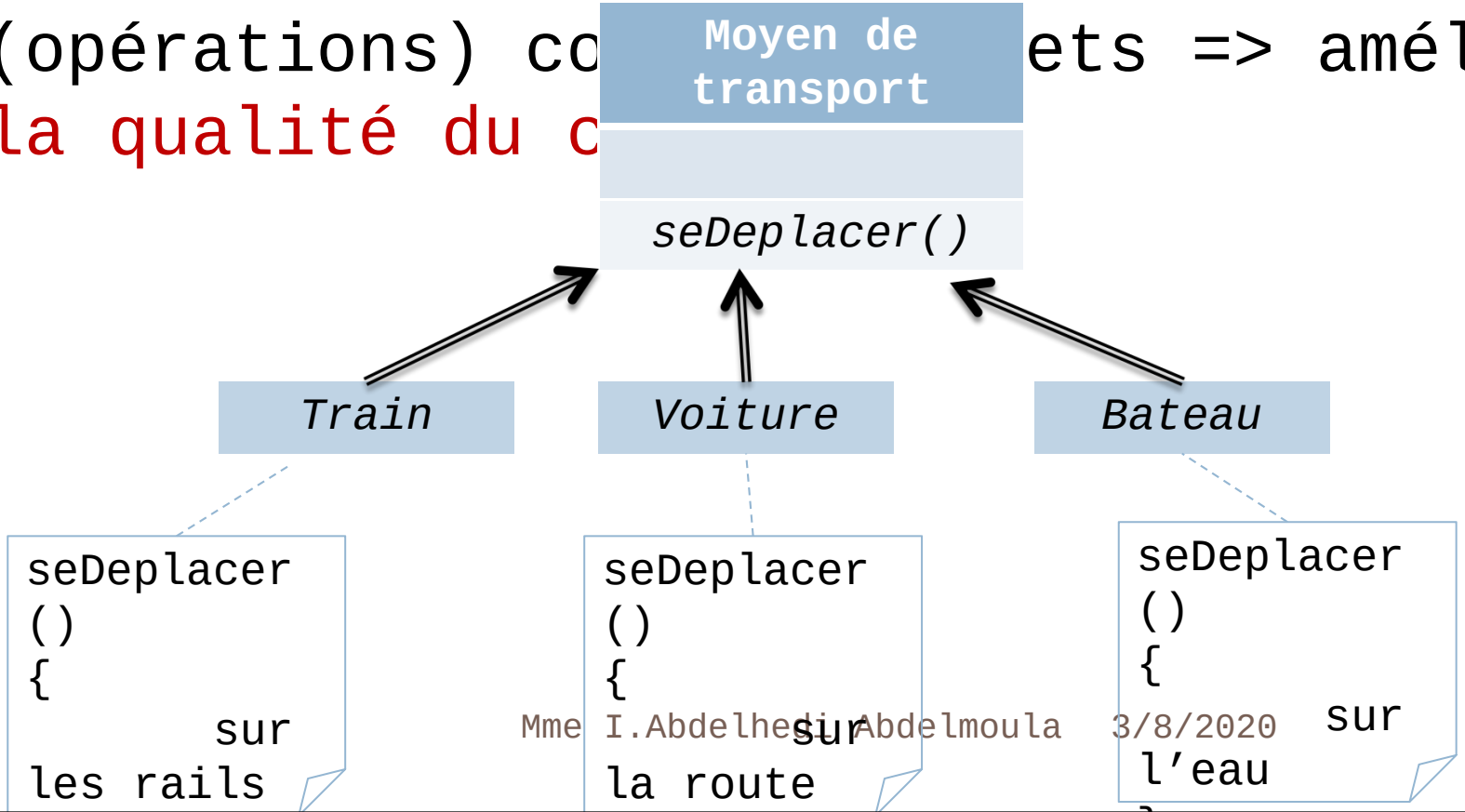
Une instance de «**Forme**» **est** obligatoirement une instance de la (

Opération abstraite = opération sans corps d'une classe abstraite



Polymorphisme

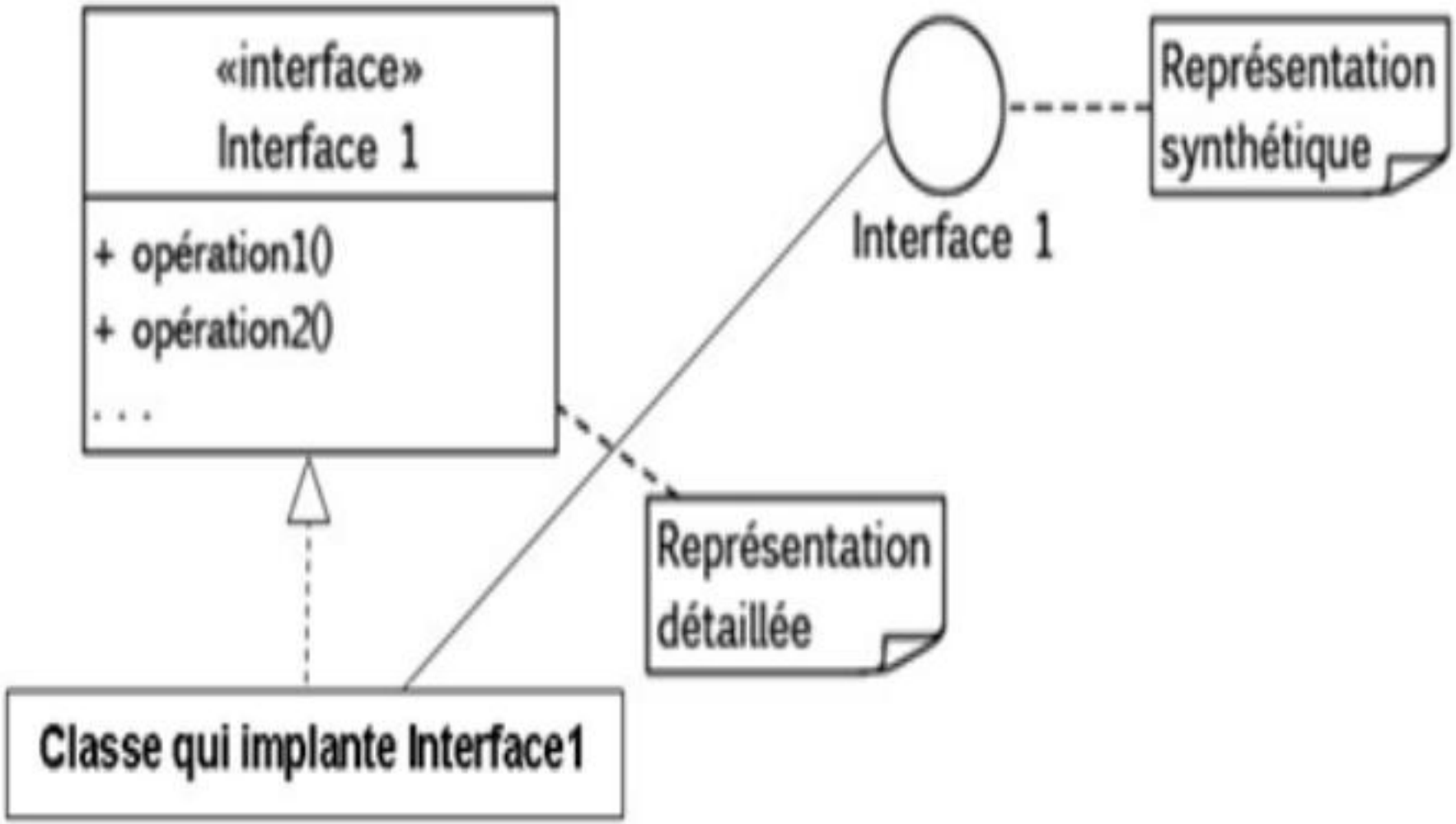
- opération qui s'applique différemment à des objets de classes différentes = Factorisation de comportement (opérations) co ets => améliore la qualité du c



Une interface

- Une classe permet de définir à la fois les objets de la classe et son interface.
- Une interface **fournit une vue totale** (interface complète) **ou partielle d'un ensemble de fonctionnalités** offerts par une classe (une partie d'interface) qui sera commune à plusieurs objets
- Fournit un ensemble d'attributs et d'opérations assurant une fonctionnalité cohérente
- Définit le comportement visible d'une classe (opérations ayant une visibilité « public »)

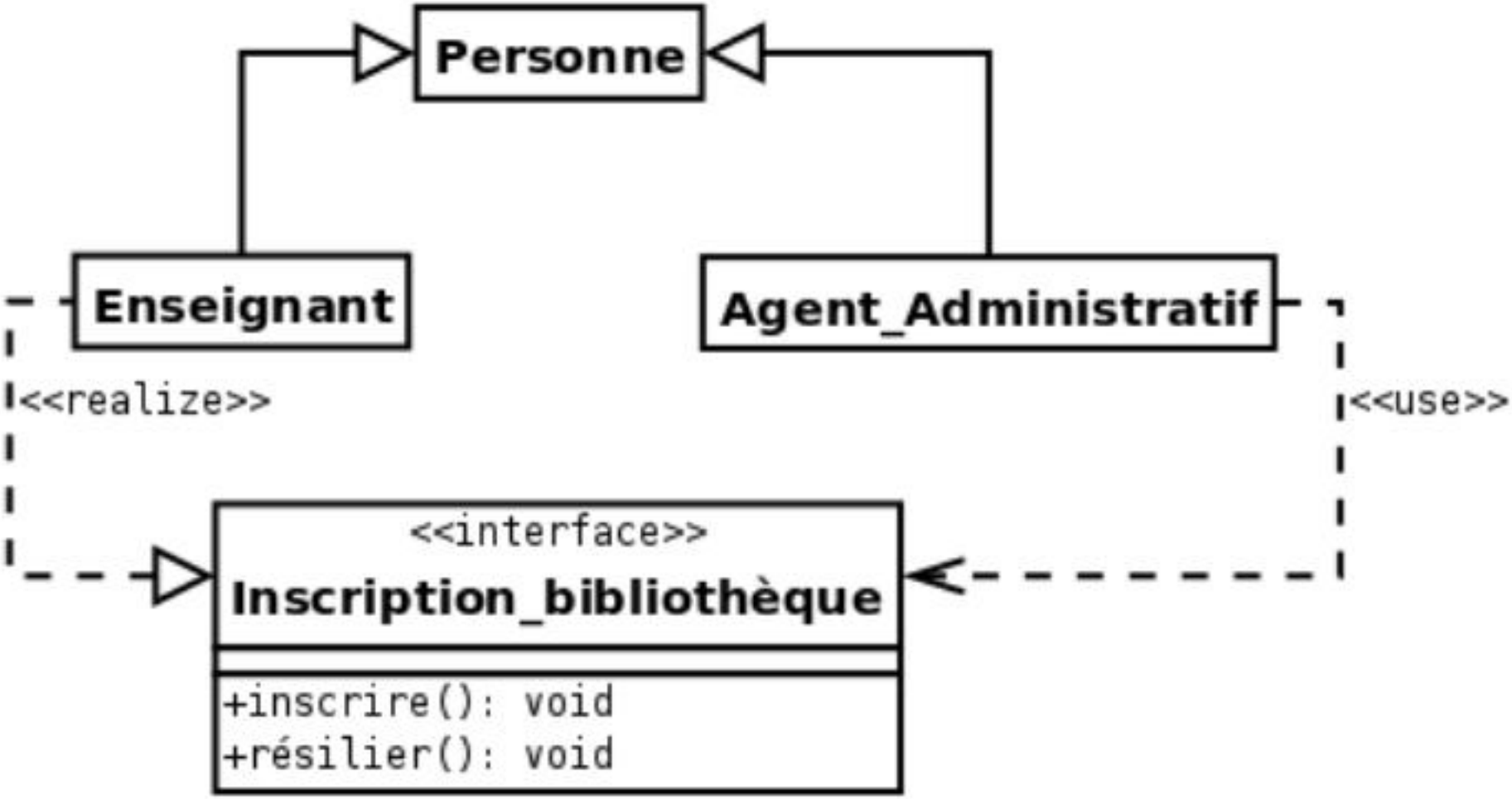
Interface – formalisme



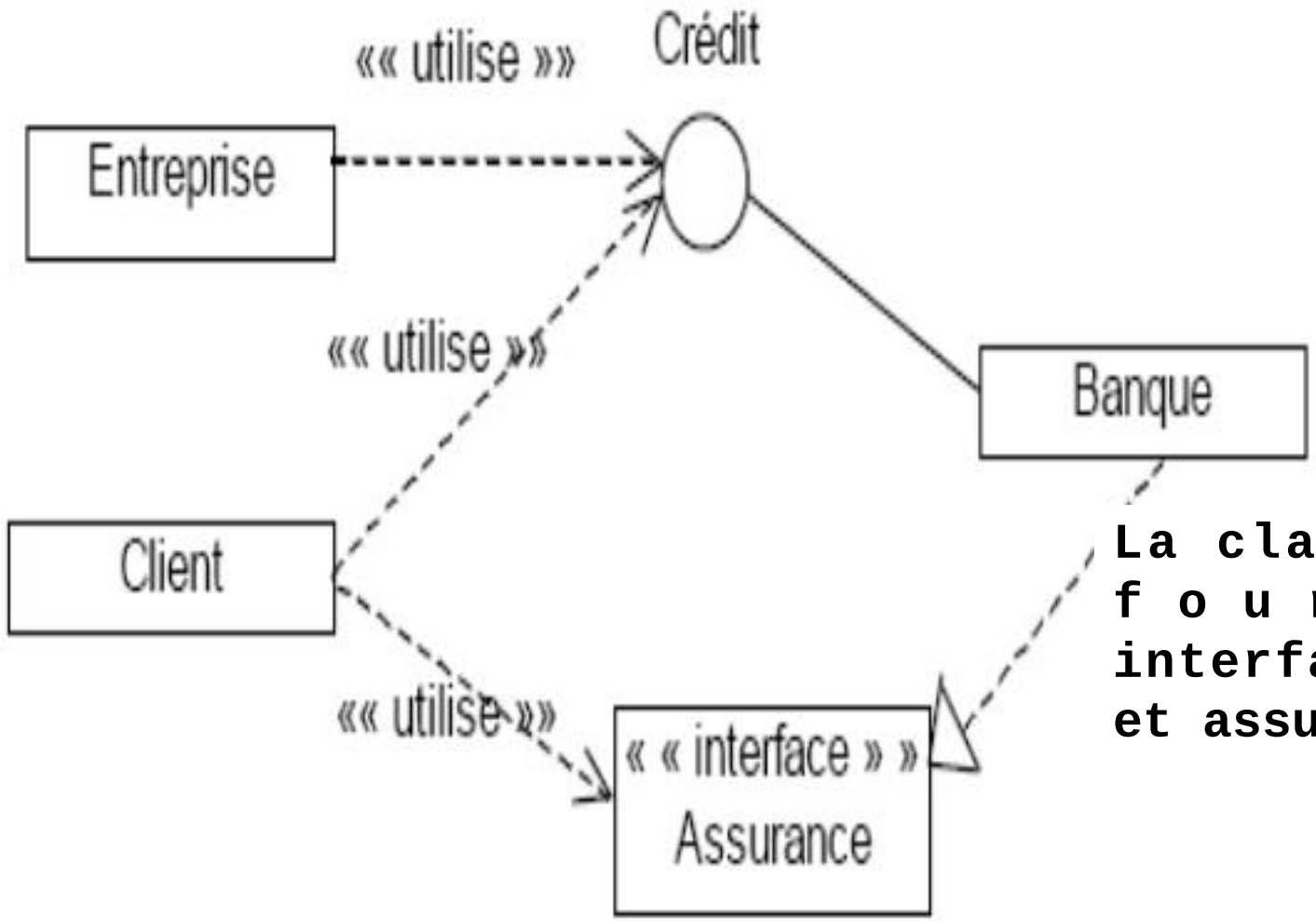
Relations possibles sur une interface

- **<<fournit>>** : Classe fournit une à plusieurs interfaces à ses clients et chaque interface définit un des services de la classe ⇒ réduire la visibilité d'une classe.
- **<<utilise>>** : concerne toute classe cliente souhaitant accéder à la classe interface de manière à accéder à ses opérations.
- **<<réalise>>** : Classe réalise les opérations offertes par l'interface.

Exemple


Personne
Enseignant
Agent_Administratif
<<realize>>
<<use>>
<<interface>>
Inscription_bibliothèque
+inscrire(): void
+résilier(): void

Exemple



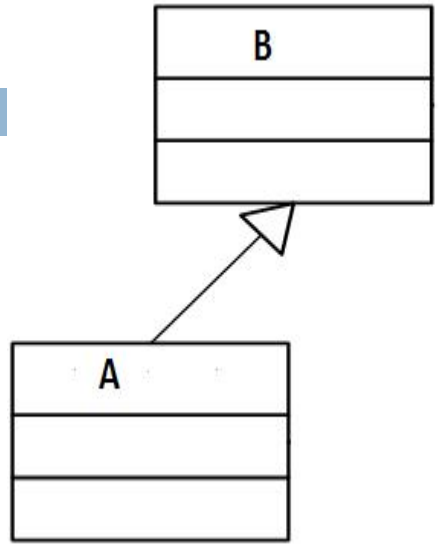
La classe Banque fournit 2 interfaces crédit et assurance

Association de dépendance

- Association unidirectionnelle exprimant une dépendance sémantique entre des éléments du modèle.
- Elle indique que la modification de la cible peut impliquer une modification de la source.
- Utilisée souvent quand une classe en utilise une autre

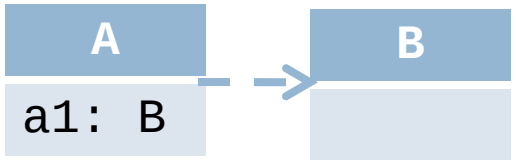
Dépendance entre classes

- Une classe A dépend d'une classe B



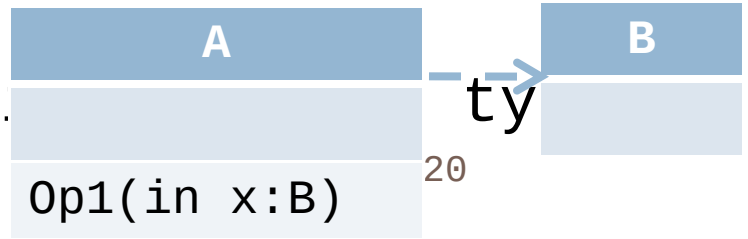
1. A hérite de B

2. A est associée à B (navigabilité restreinte de A vers B)

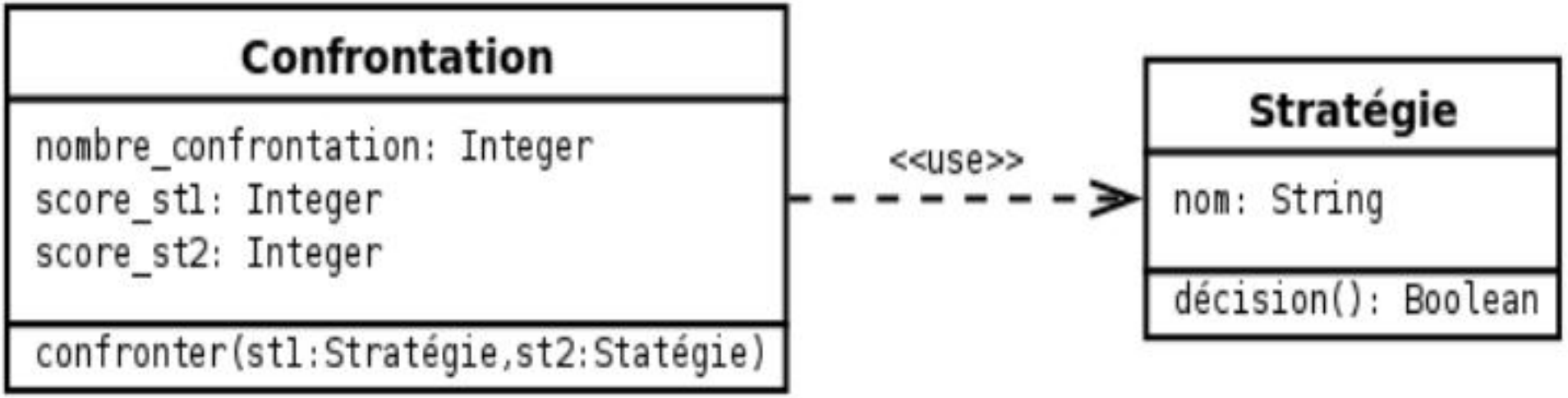


3. A possède un attribut dont le type est B

4. A possède une opérat… ty… l'un de ses paramètres



Exemple

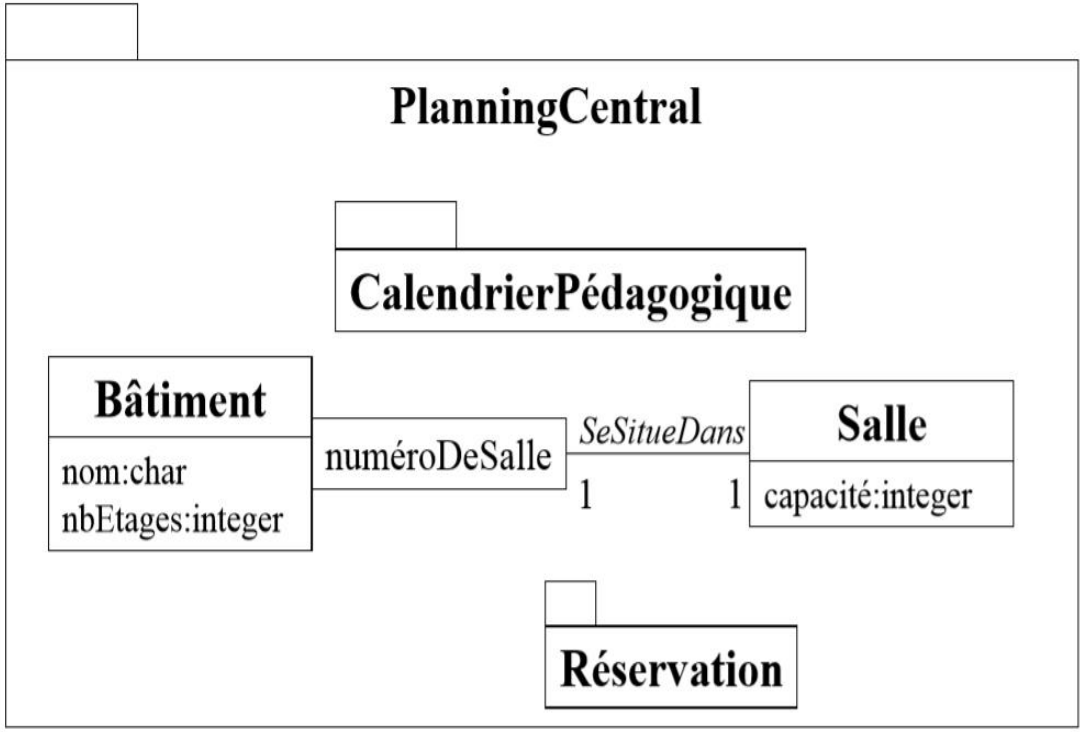


- la classe *Confrontation* utilise la classe *Stratégie* car la classe *Confrontation* possède l'opération *confronter* dont ses deux paramètre sont du type *Stratégie*.
- Si la classe *Stratégie*, notamment son interface, change, alors des modifications devront également être apportées à la classe *Confrontation*.

Définition de package

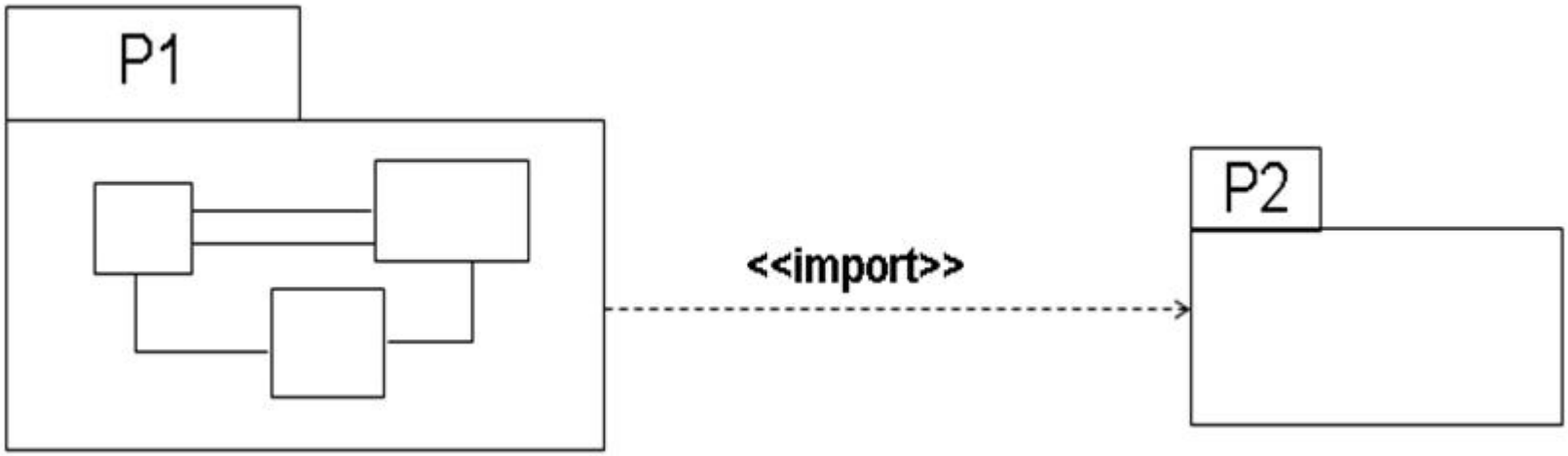
Package : groupe d'éléments partageant un thème commun

- Utile pour organiser les grands modèles
- Ne définir une classe (i.e. représenter ses propriétés) que dans un seul package
- Référencer une classe d'un autre package en n'utilisant que le nom de la classe



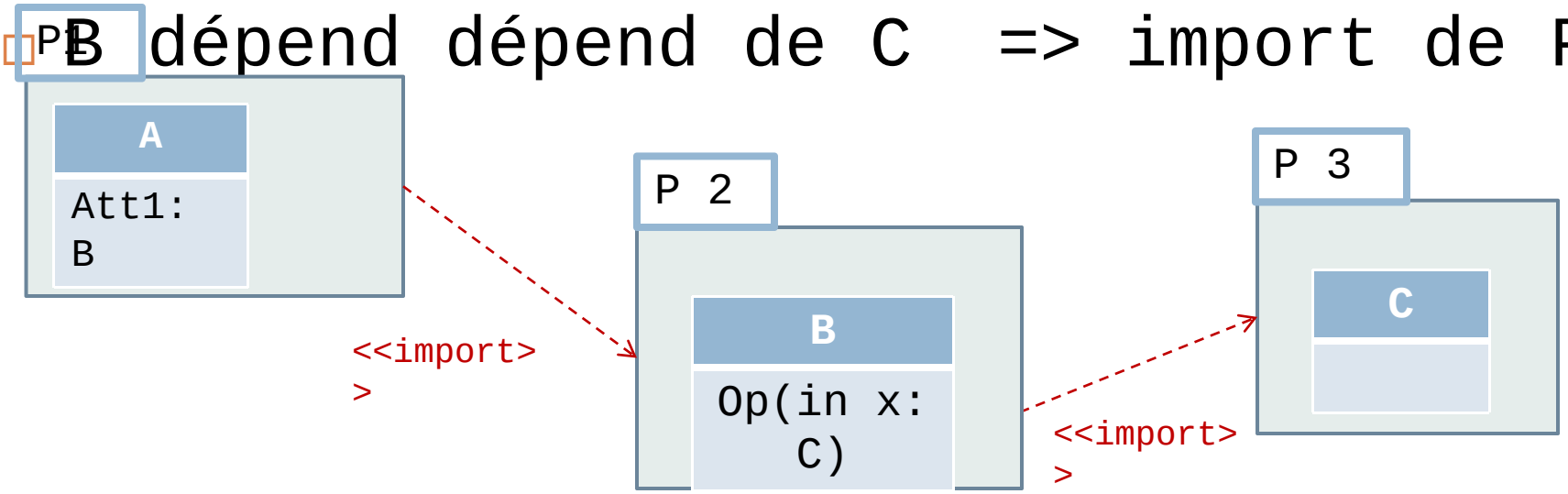
Dépendance entre paquetages

- Le paquetage P2 utilise le paquetage P1

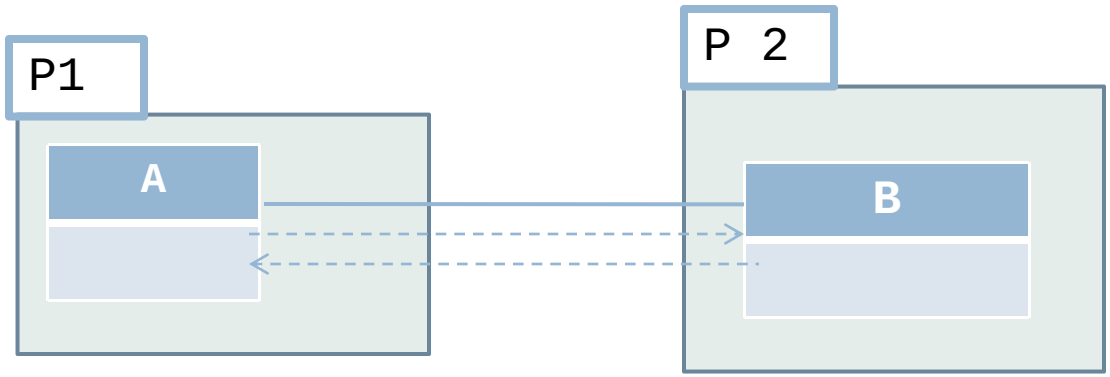


Dépendance entre packages

- A dépend de B => relation d'import de P1 vers P2
- B dépend dépend de C => import de P2

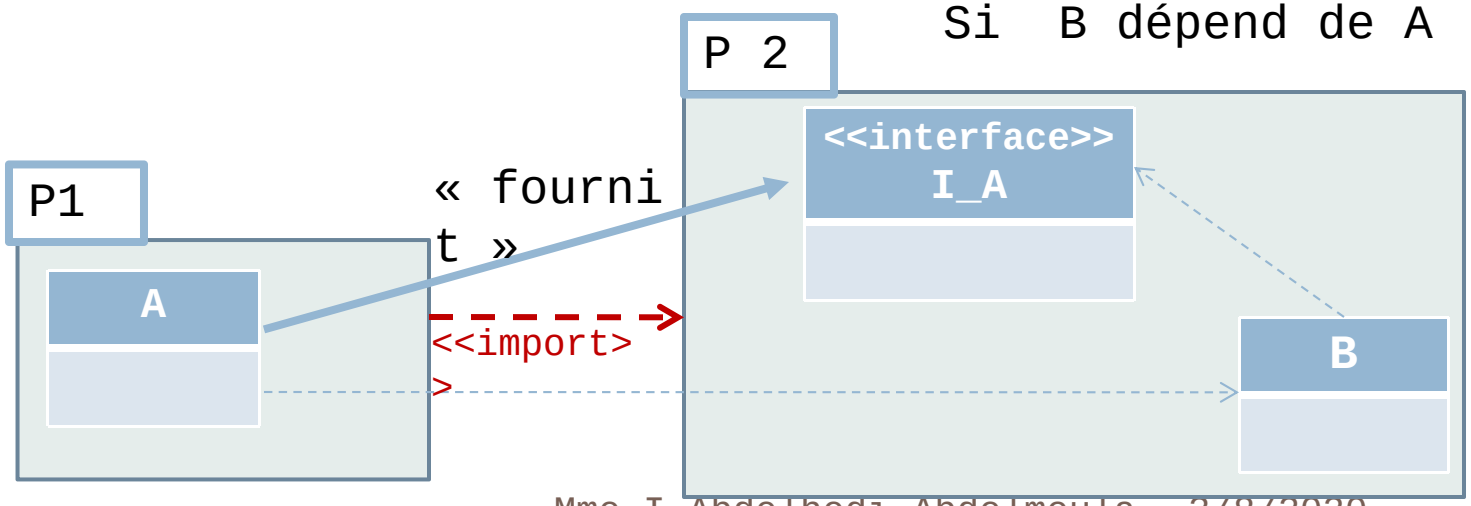


Casser les cycles de dépendance



Association bidirectionnelle entre A et B : dépendance mutuelle

Si B dépend de A



Exercices d'application

1. Un pays possède une capitale.
2. Une personne dîne avec une fourchette.
3. Un chemin peut représenter un fichier ou un répertoire.
4. Un chemin est un répertoire avec éventuellement un nom de fichier.
5. Un fichier contient des enregistrements ordonnés.
6. Un fichier est accessible par un utilisateur selon des droits d'accès.
7. Un dessin est soit du texte, soit une forme géométrique, soit un groupe de dessins.
8. Des personnes utilisent un langage pour un projet.

Exercice 1

- Un ordinateur est composé d'un ou plusieurs moniteurs, d'un boîtier, d'une souris optionnelle et d'un clavier.
- Un boîtier a un châssis métallique, une carte mère, plusieurs barrettes de mémoire (RAM, ROM et cache), un ventilateur optionnel, des supports de stockage (disque-dur, CD-ROM, DVD-ROM...), et des cartes périphériques (son, réseau, graphique...).

Exercice 2

- L'université comporte des personnels administratifs et techniques, des enseignants, des étudiants et des chercheurs (qui sont tous des personnes).
- Certains étudiants peuvent être des chercheurs (les doctorants) ou des enseignants (les assistants enseignants).
- Certaines personnes (étudiants

Exercice 3

- Un éditeur de documents graphiques supporte le groupement d'objets graphiques.
- Un document se compose de plusieurs feuilles, chacune contenant des objets graphiques (texte, forme géométrique et groupe d'objets).
- Un groupe est un ensemble d'objets pouvant contenir d'autres groupes. Un groupe doit contenir au moins deux éléments. Les formes

Exercice 4

- Une personne physique peut avoir jusqu'à trois sociétés (personnes morales) qui l'emploient.
- Chaque personne physique possède un numéro de sécurité sociale qui l'identifie.
- Une voiture a un numéro d'immatriculation. Une voiture est la propriété d'une personne (physique ou morale).
- Un emprunt dans une banque peut

PARTIE II - Diagrammes d’objets DOB

Types de diagrammes d’objets

- Diagrammes d’objets (point de vue statique)
- Diagrammes d’interaction (point de vue dynamique)
  - Diagramme de séquence
  - Diagramme de collaboration
- Deux niveaux de représentation des collaborations
  - Niveau Spécification (des rôles et des messages)

Définition d'un DOBJ

- DOBJ = Représentation d'un ensemble d'objets et de leurs liens, **exprimant des vues statiques des instances des éléments** qui apparaissent dans les diagrammes de classes.
- Une instance du DCL illustrant l'état statique du système à un moment donné.

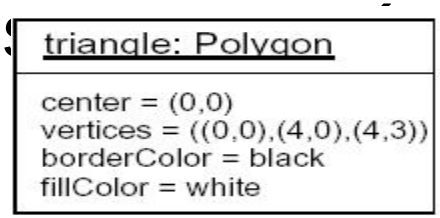
Il est composé de:

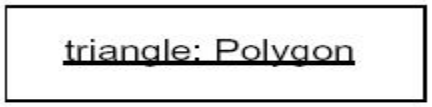
- **objets (instances de classes),**
- **liens (instances d'associations)**

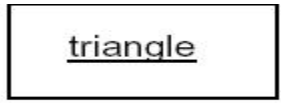
- Montre un contexte e.g., avant ou après une interaction,
- Facilite la compréhension des

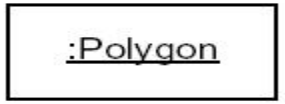
Objet

- Un objet est une instance d'une classe : il représente l'état d'une classe à un ins
- *Représentation*
  - Nom : Classe
  - Nom
  - :Classe



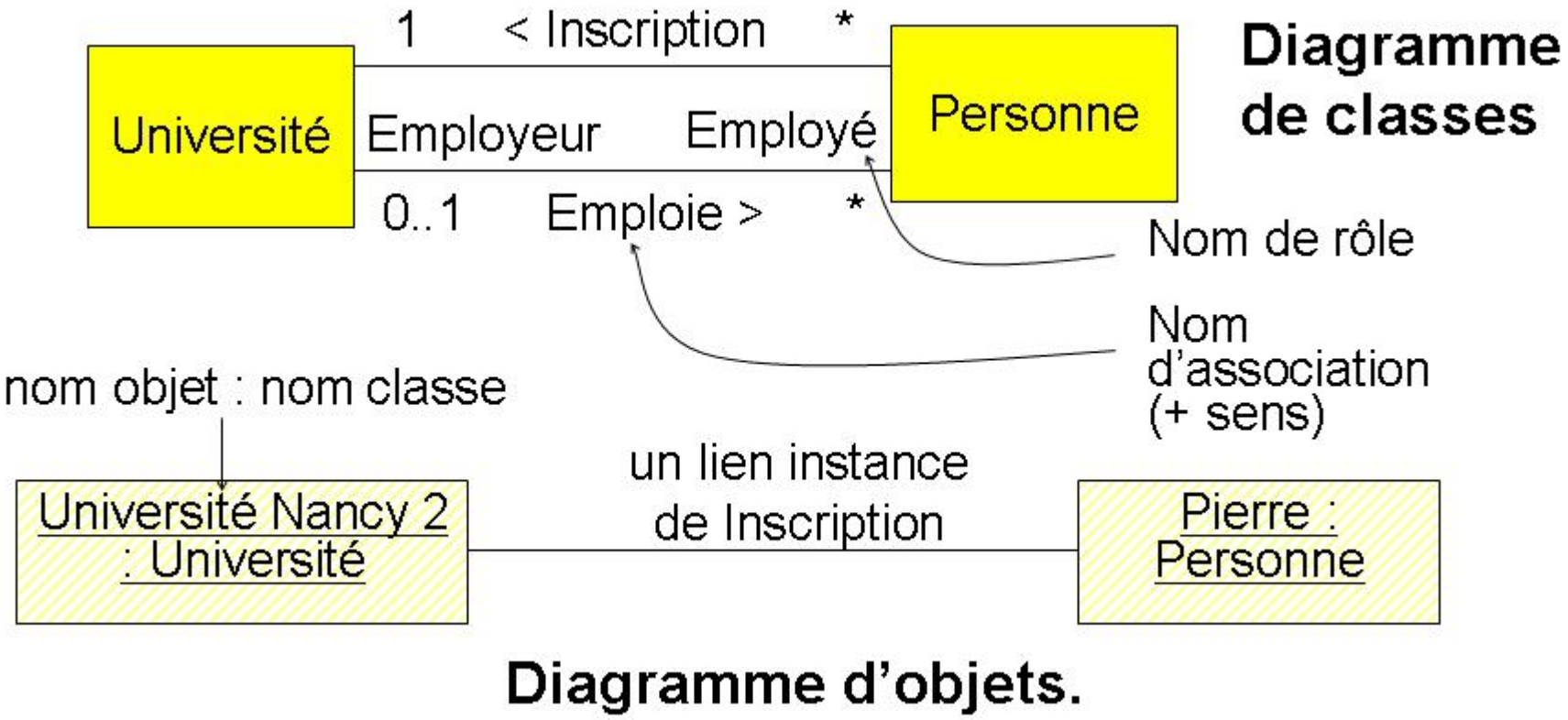






- Des groupes d'objets instances d'une même classe peuvent se représenter.
- *Un message envoyé vers un groupe est reçu par tous les* objets du

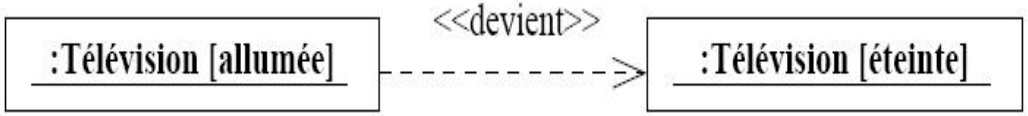
Exemple


1
< Inscription
*
Université
Employeur
Employé
Personne
0..1
Emploie >
*
Diagramme de classes
Nom de rôle
Nom d'association (+ sens)
nom objet : nom classe
un lien instance de Inscription
Université Nancy 2 : Université
Pierre : Personne
Diagramme d'objets.

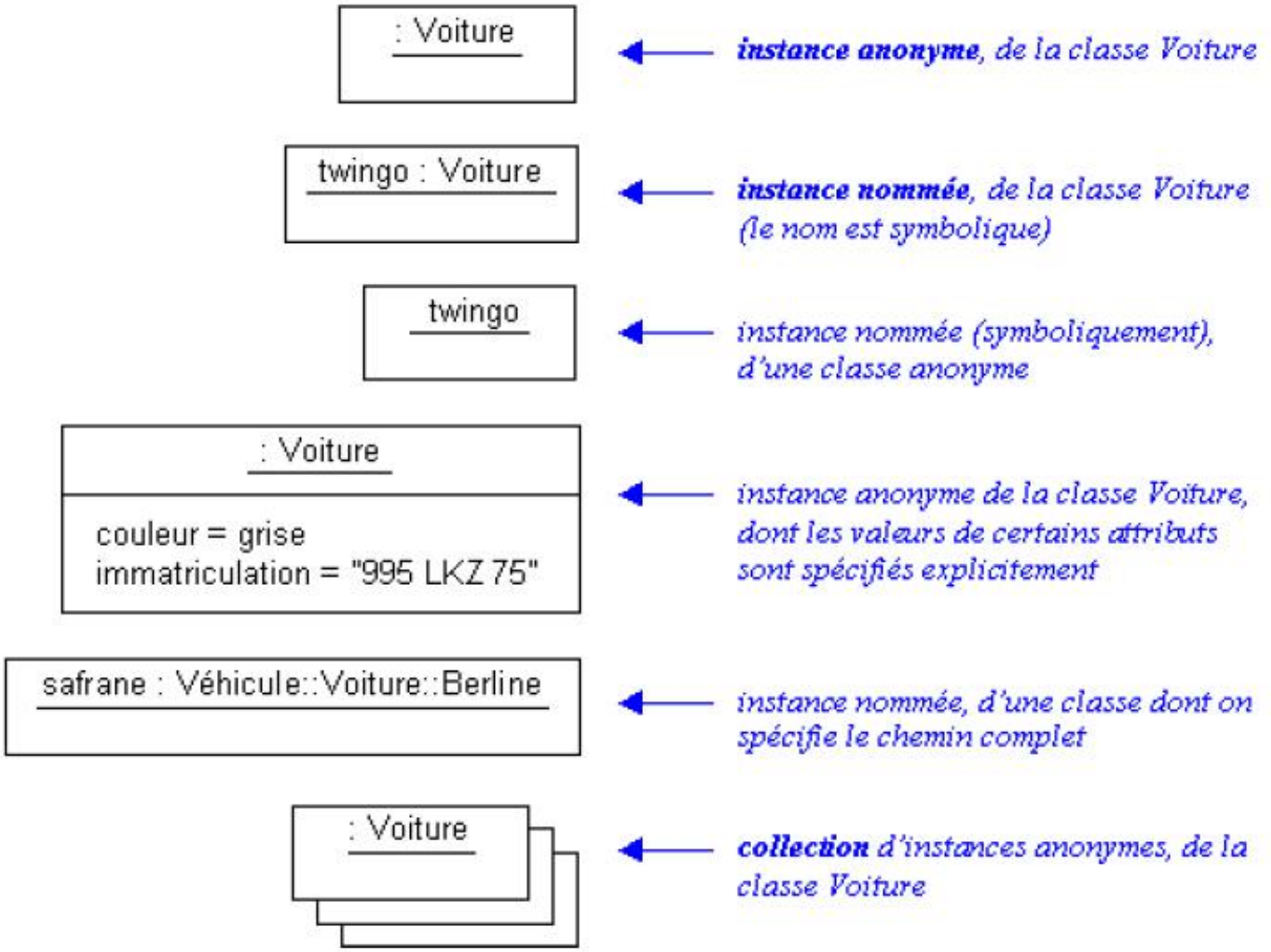
Objet (suite)

- L'état d'un objet est déterminé par **les valeurs de ses attributs** ⇒ nommer un état afin d'indiquer clairement dans qu :Ordinateur [calcule] trouve un objet.

- Les représentations des :Voiture (vitesse = 100, couleur = rouge) s peuvent contenir des att s significatifs.



Exemples de notations d'objets



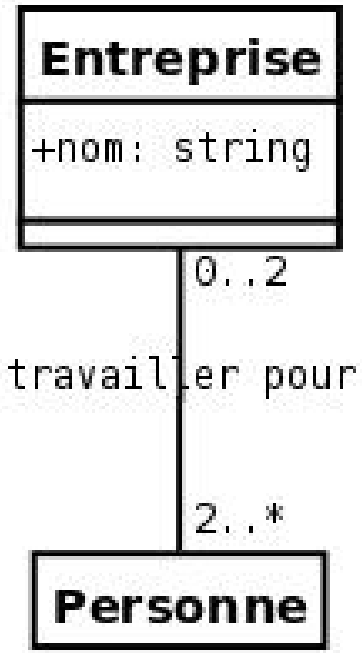
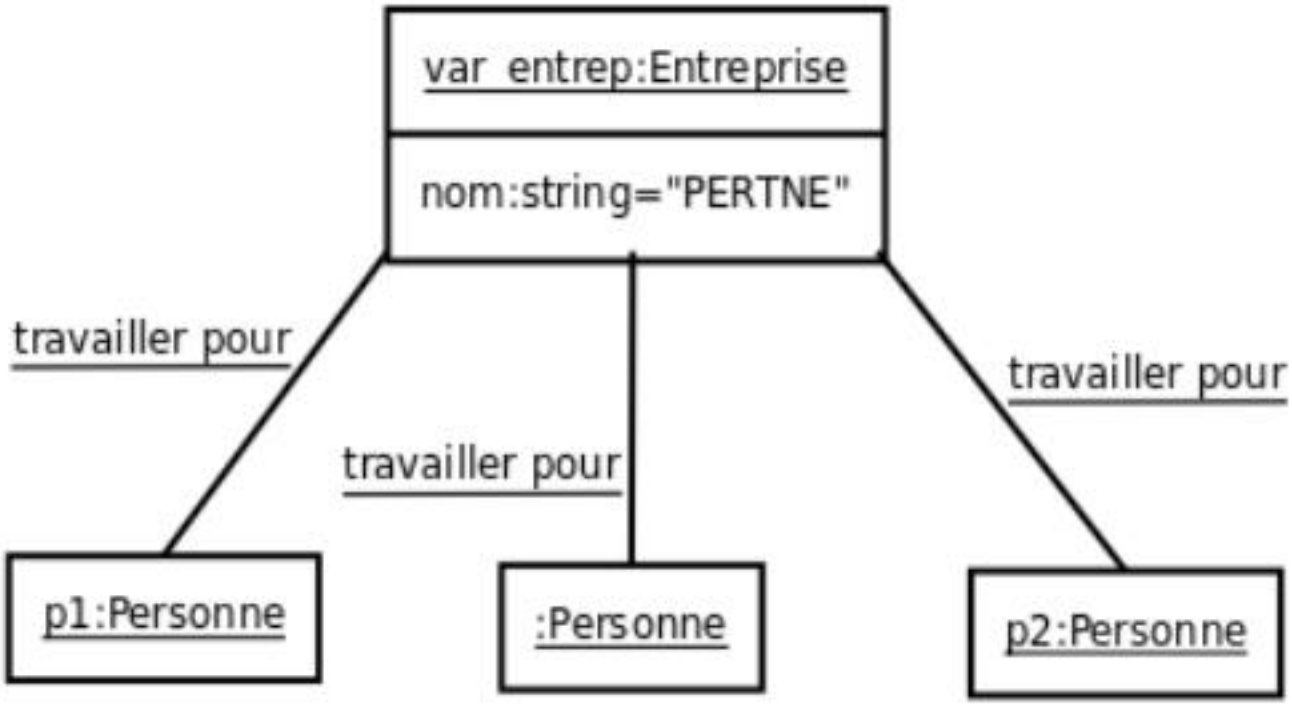
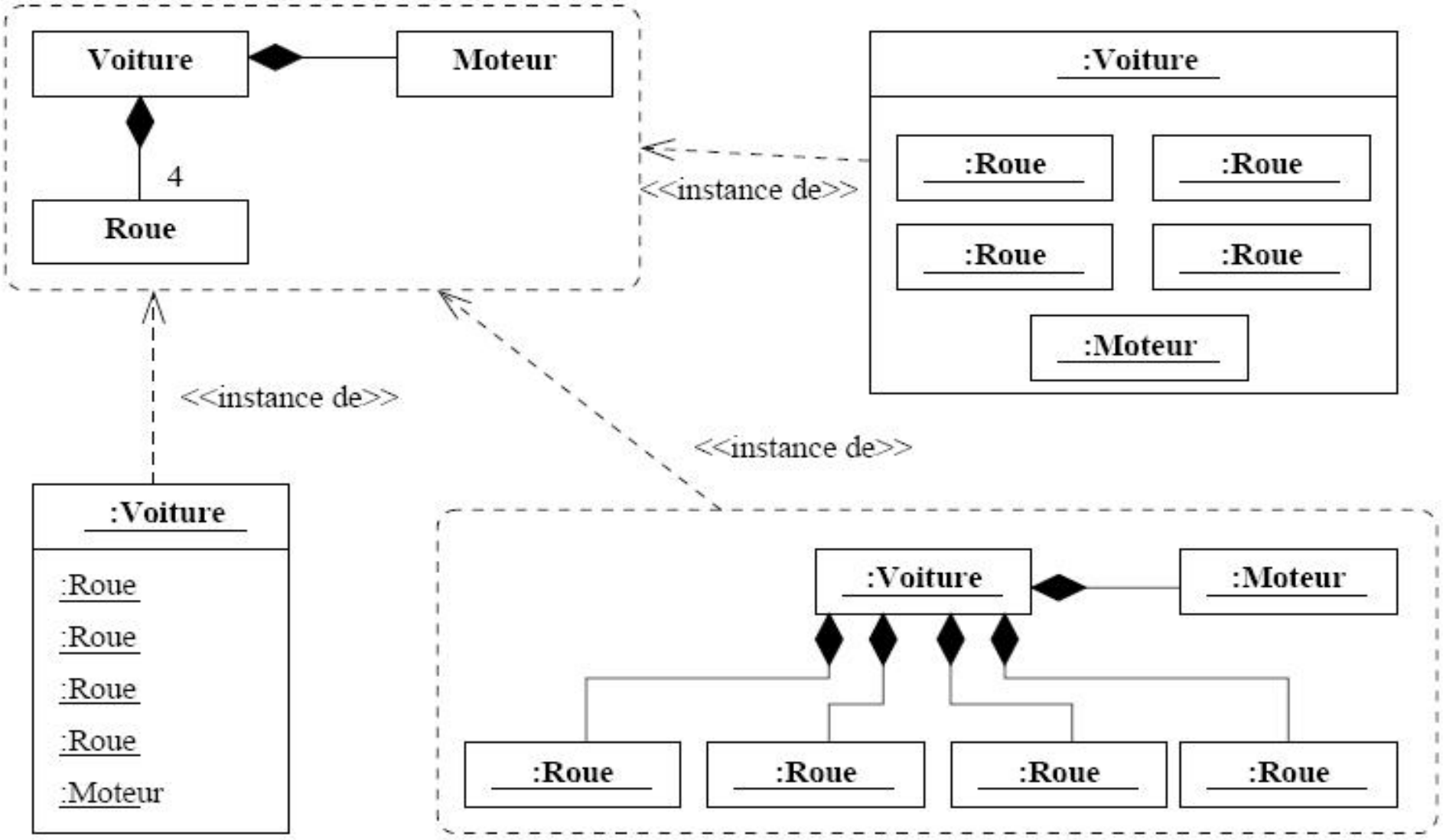


Diagramme de classes

Diagramme d'objets correspondant



Objet composite



Liens entre objets

- Les objets sont reliés par des instances d'associations : les liens.
- *Il représente une relation entre objets à* ***un instant donné.***
- *la multiplicité des extrémités des liens est toujours de* ***1***

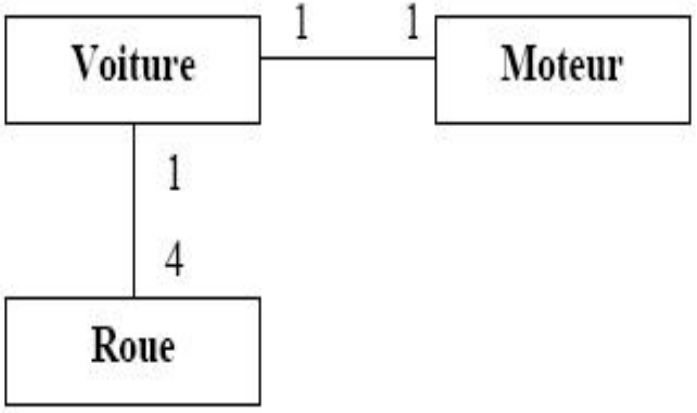


Diagramme de classes

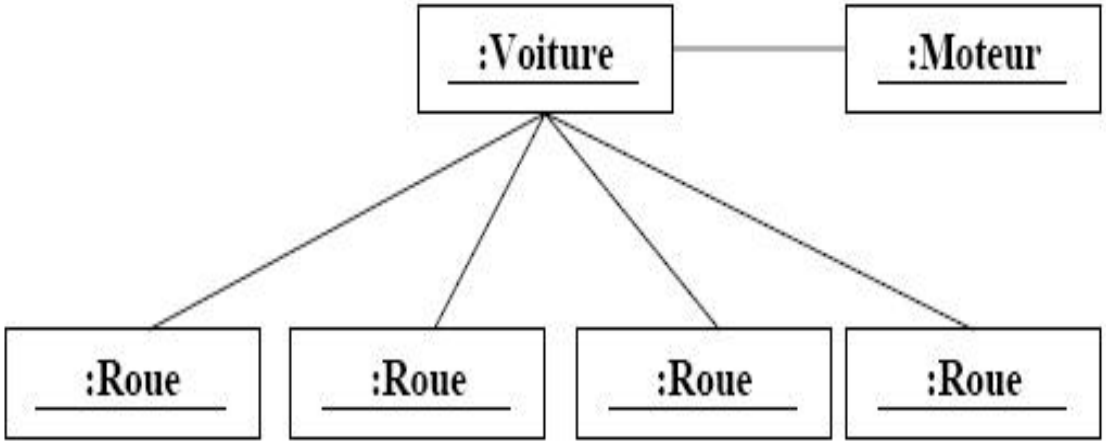
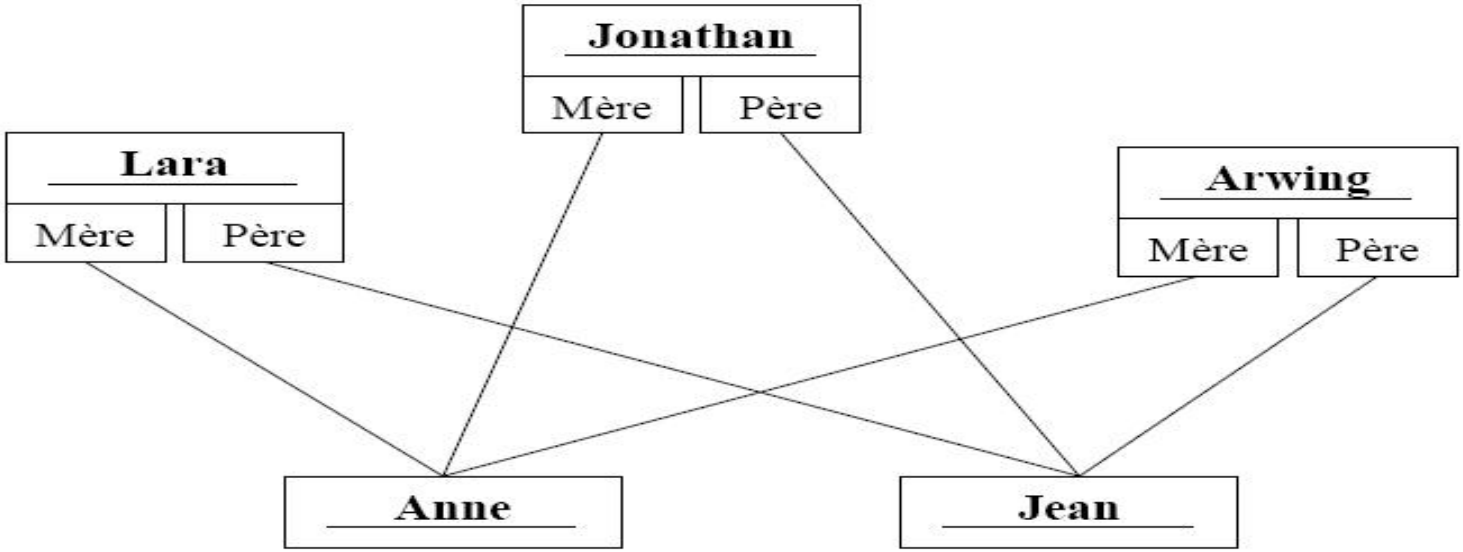


Diagramme d'objets

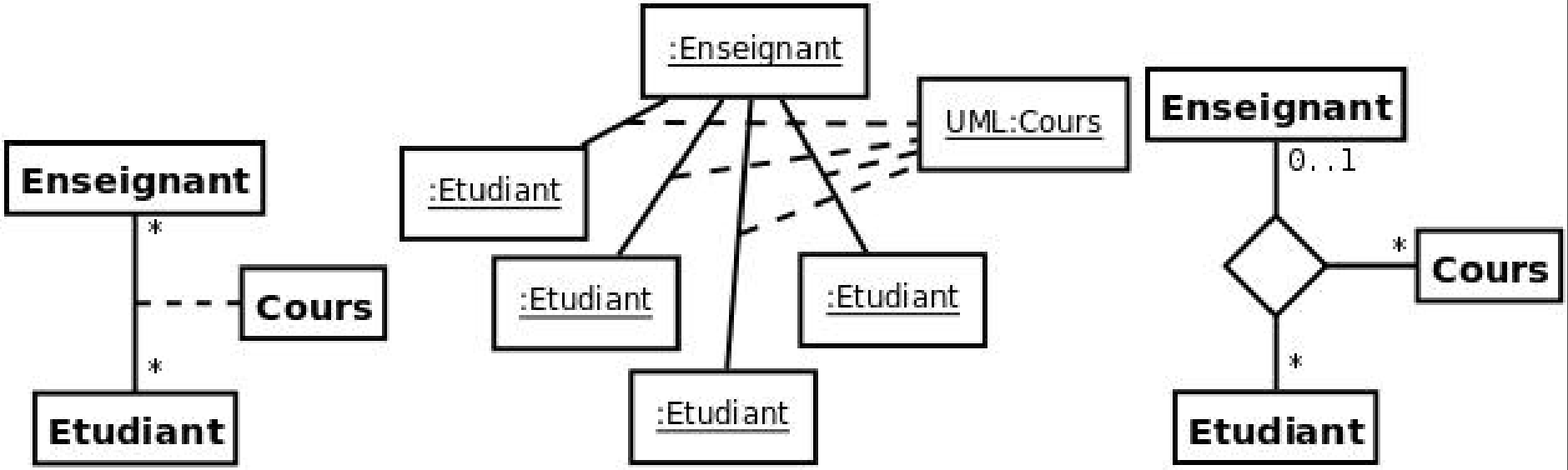
Le rôle des objets

- Les rôles des associations peuvent être représentés explicitement



Classe association

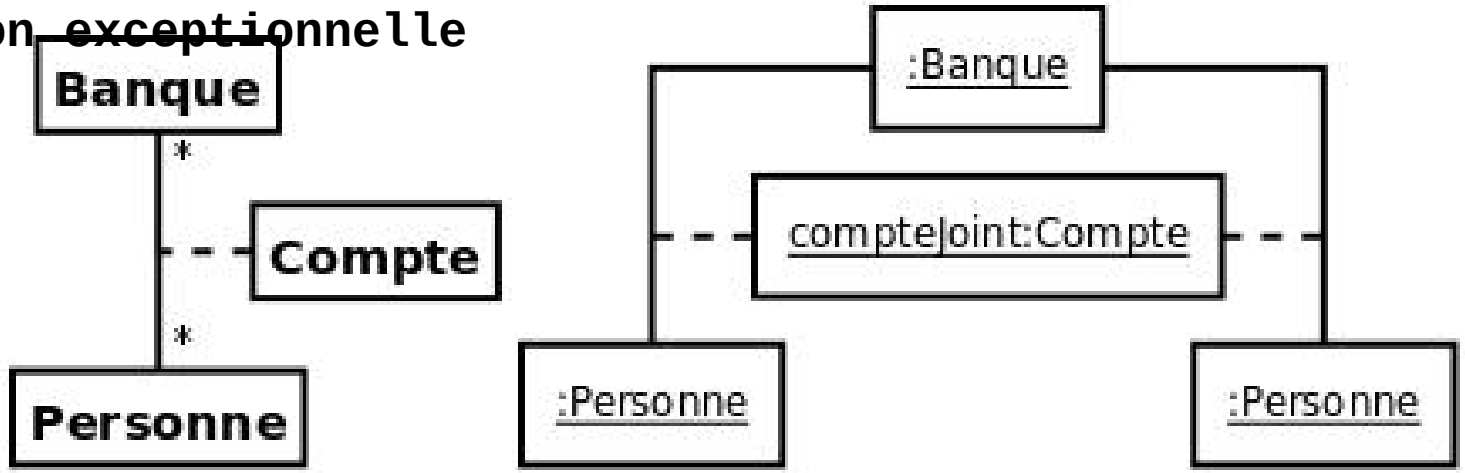
- Si un même cours doit concerner plusieurs couples *Enseignant/Etudiant*, il ne faut pas utiliser une classe-association, mais une association ternaire comme sur le modèle de droite.



Instance de classe association

- Une instance de classe-association ne peut être associée qu'à une instance de chacune des classes associées

Le diagramme d'objets de droite, illustrant le cas de figure d'un compte joint, n'est pas une instance *normale* du diagramme de classe de gauche mais peut préciser **une situation exceptionnelle**



Exercices d’application

1. Un pays possède une capitale et plusieurs frontières avec d’autres pays. Cas de la Tunisie.
2. Une route connecte deux villes.
3. Un chemin peut représenter un fichier ou un répertoire.
4. Un chemin est un répertoire avec éventuellement un nom de fichier.
5. Un fichier contient des enregistrements ordonnés.
6. Un fichier est accessible par un utilisateur selon des droits d'accès.

Exercices d'application…

7. Un polygone est constitué de plusieurs points. Un point a une abscisse et une ordonnée.
8. Un dessin est soit du texte, soit une forme géométrique, soit un groupe de dessins.
9. Des personnes utilisent un langage pour un projet.
10. Une personne joue dans une équipe pour une certaine durée.
11. Une équipe est composée de plusieurs personnes.

Exercices d'application…

12. Un client demande une réparation qui sera effectuée par un ou plusieurs mécaniciens car elle nécessite diverses compétences.
13. Une galerie expose des œuvres, réalisées par des artistes et représentant divers thèmes. Des clients accueillis par la galerie achètent les œuvres exposées.

- Un ordinateur est composé d'un ou plusieurs moniteurs, d'un boîtier, d'une souris optionnelle et d'un clavier. Un boîtier a un châssis métallique, une carte mère, plusieurs barrettes de mémoire (RAM, ROM et cache), un ventilateur optionnel, des supports de stockage (disque-dur, CD-ROM, DVD-ROM...), et des cartes périphériques (son, réseau, graphique...).

- L'université comporte des personnels administratifs et techniques, des enseignants, des étudiants et des chercheurs (qui sont toutes des personnes). Certains étudiants peuvent être des chercheurs (les doctorants) ou des enseignants (les assistants enseignants). Certaines personnes (étudiants ou non) peuvent être à la fois chercheurs et enseignants.