



Polytech Paris-Sud
Formation initiale 3^e année
Spécialité Informatique
Année 2016-2017

UML

Cours 3

Diagrammes de classes

Delphine Longuet
delphine.longuet@lri.fr

<http://www.lri.fr/~longuet/Enseignements/16-17/Et3-UML>

Objets et classes

Conception orientée objet : Représentation du système comme un ensemble d'objets interagissant

Diagramme de classes

- Représentation de la **structure interne** du logiciel
- Utilisé surtout en conception mais peut être utilisé en analyse

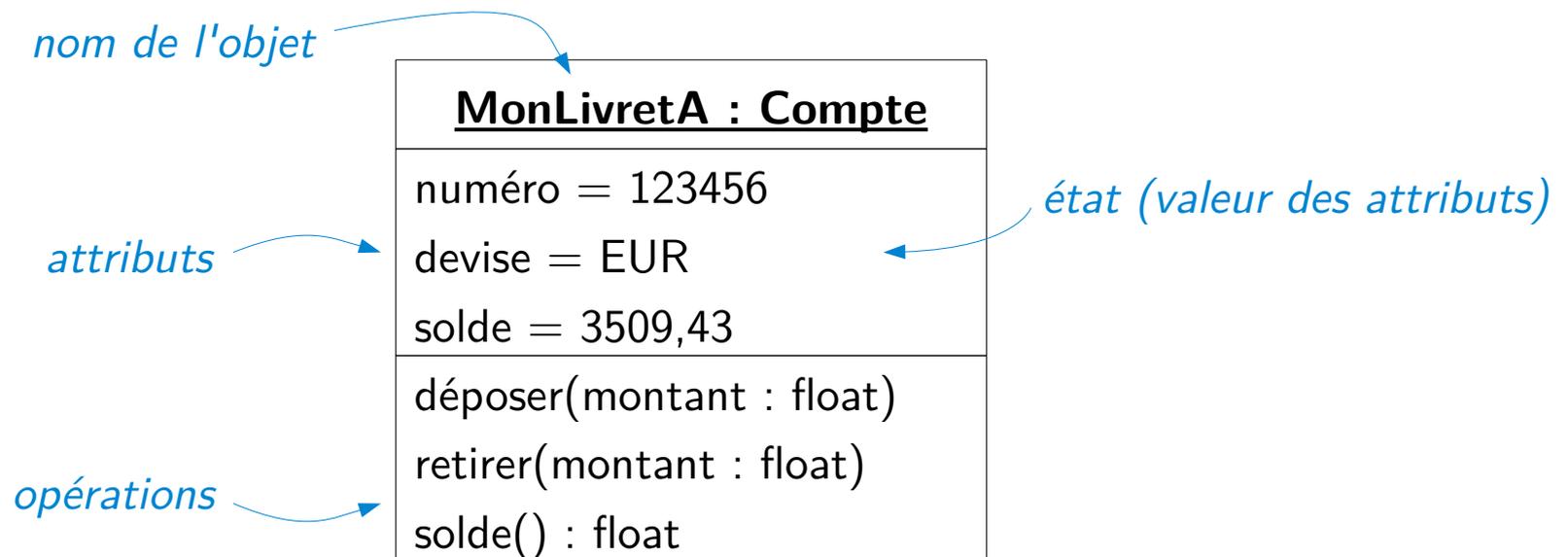
Diagramme d'objets

- Représentation de l'**état** du logiciel (objets + relations)
- Diagramme **évoluant avec l'exécution** du logiciel
 - création et suppression d'objets
 - modification de l'état des objets (valeurs des attributs)
 - modification des relations entre objets

Objets et classes

Objet

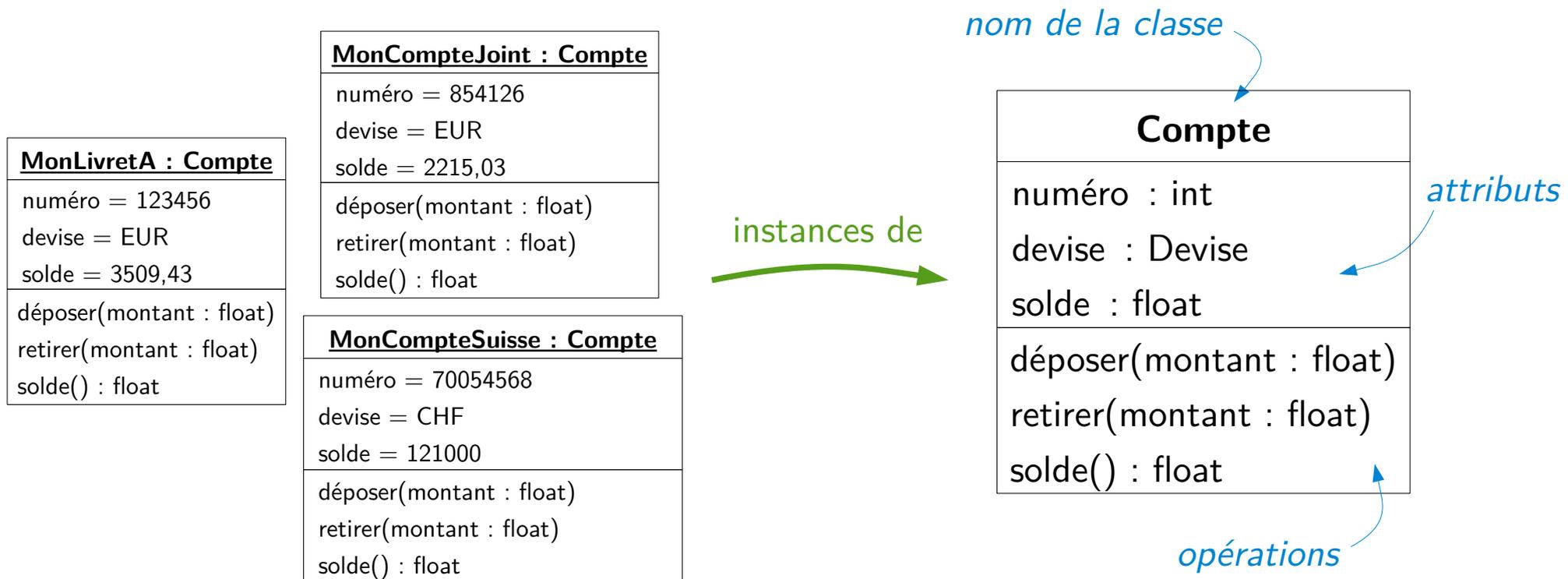
- Entité concrète ou abstraite du domaine d'application
- Décrit par : identité (adresse mémoire)
 - + état (attributs)
 - + comportement (opérations)



Objets et classes

Classe : Regroupement d'objets de même nature (mêmes attributs + mêmes opérations)

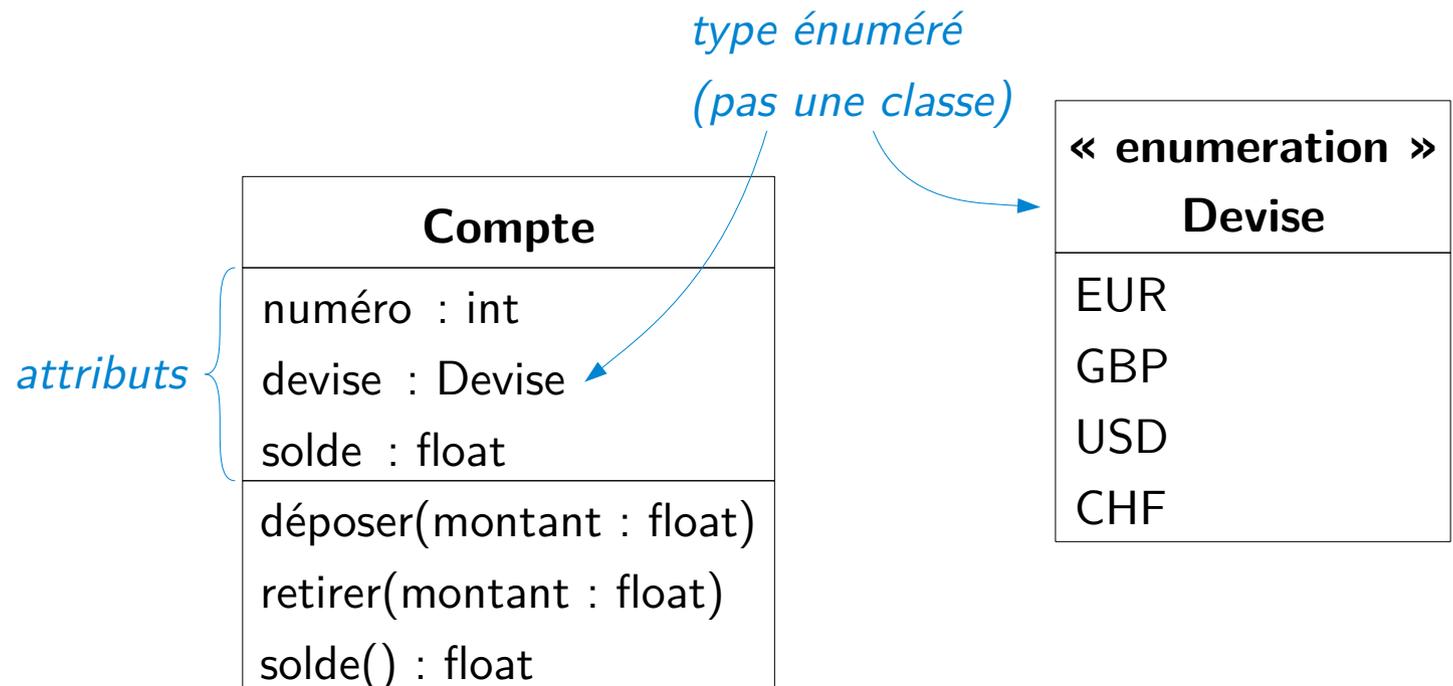
Objet = instance d'une classe



Classes

Attributs

- **Caractéristique partagée** par tous les objets de la classe
- Associe à chaque objet une **valeur**
- **Type associé simple** (int, bool...), primitif (Date) ou énuméré



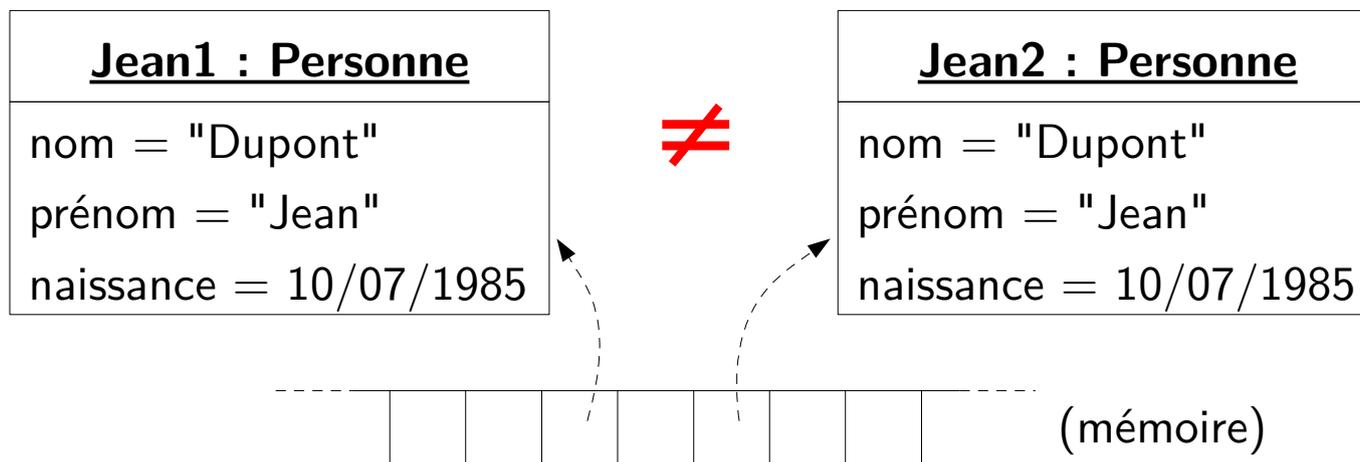
Classes

Attributs

- **Caractéristique partagée** par tous les objets de la classe
- Associe à chaque objet une **valeur**
- **Type associé simple** (int, bool...), primitif (Date) ou énuméré

Valeur des attributs : État de l'objet

- Objets différents (identités différentes) peuvent avoir mêmes attributs

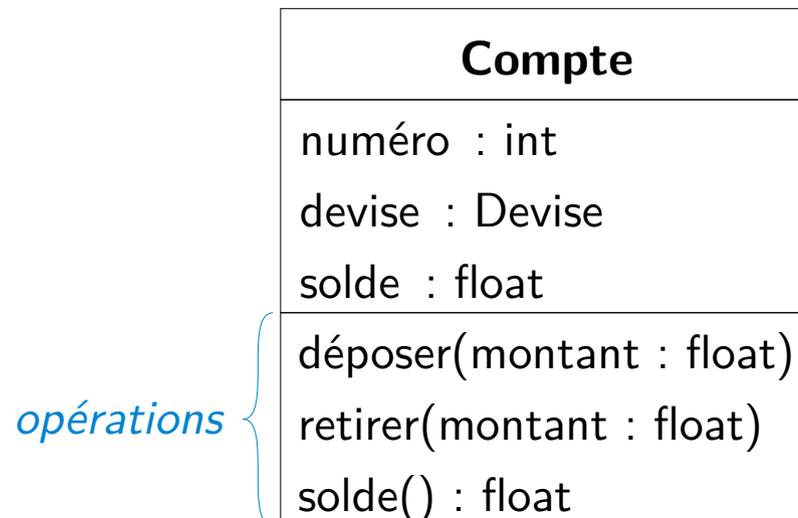


Classes

Opérations

- **Service** qui peut être demandé à tout objet de la classe
- **Comportement commun** à tous les objets de la classe

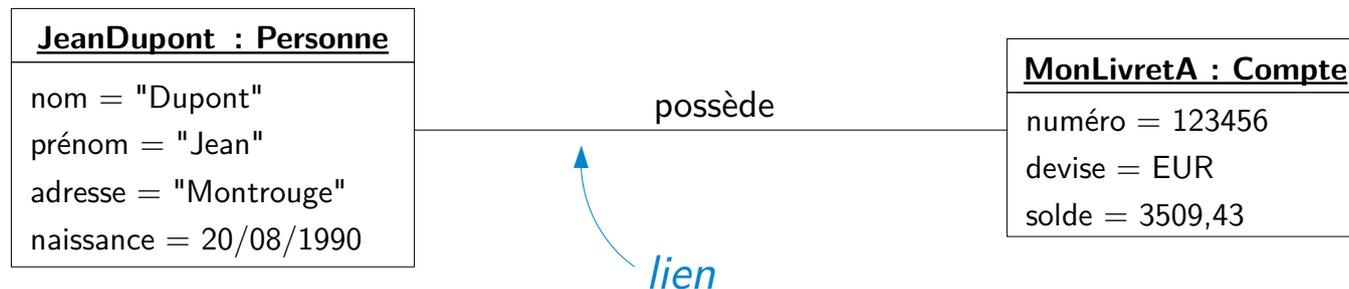
 Ne pas confondre avec une méthode = implantation de l'opération



Relations entre objets

Lien entre objets

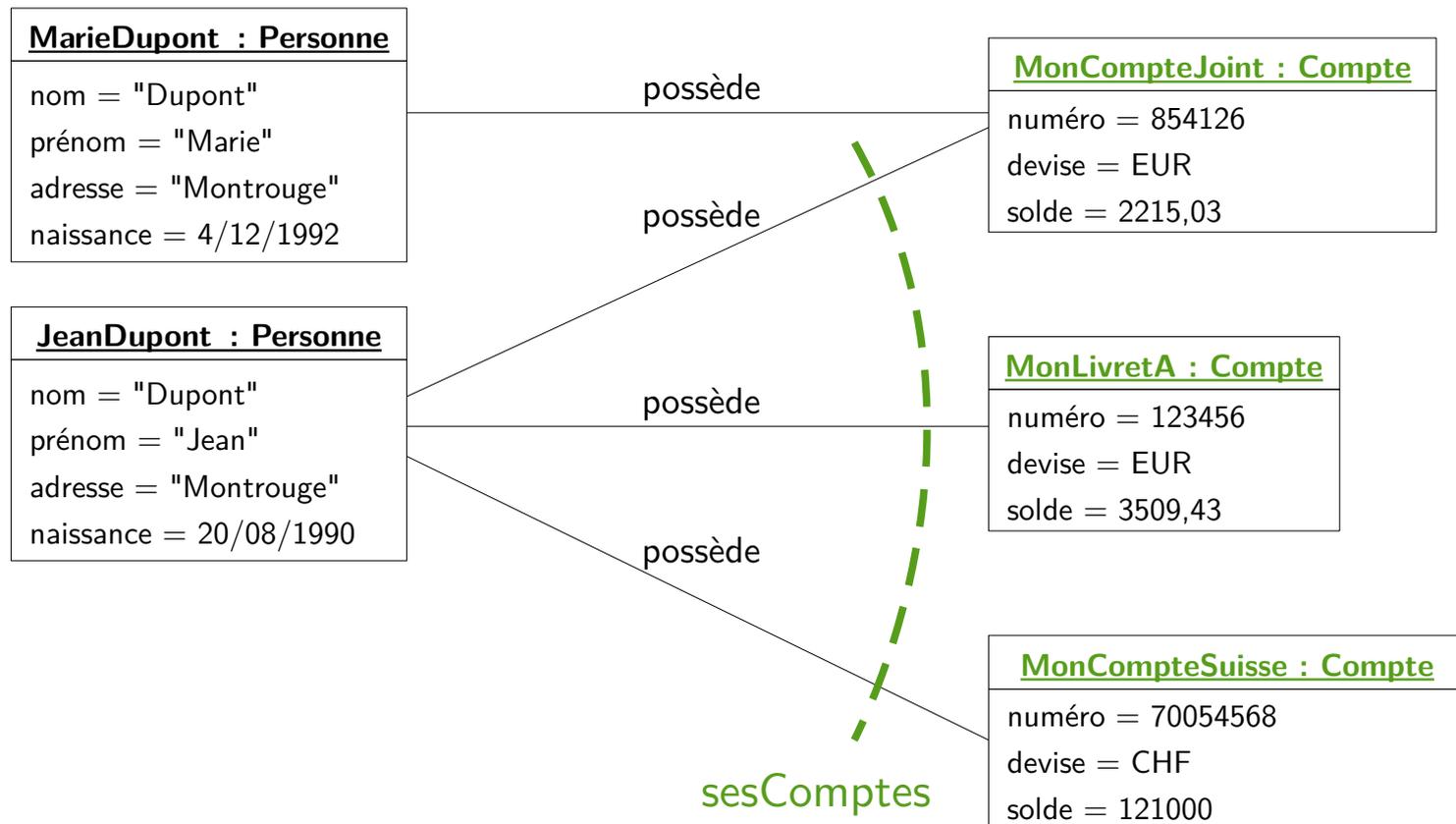
- relation binaire (en général)
- au plus un lien entre deux objets (pour une association)



Relations entre objets

Lien entre objets

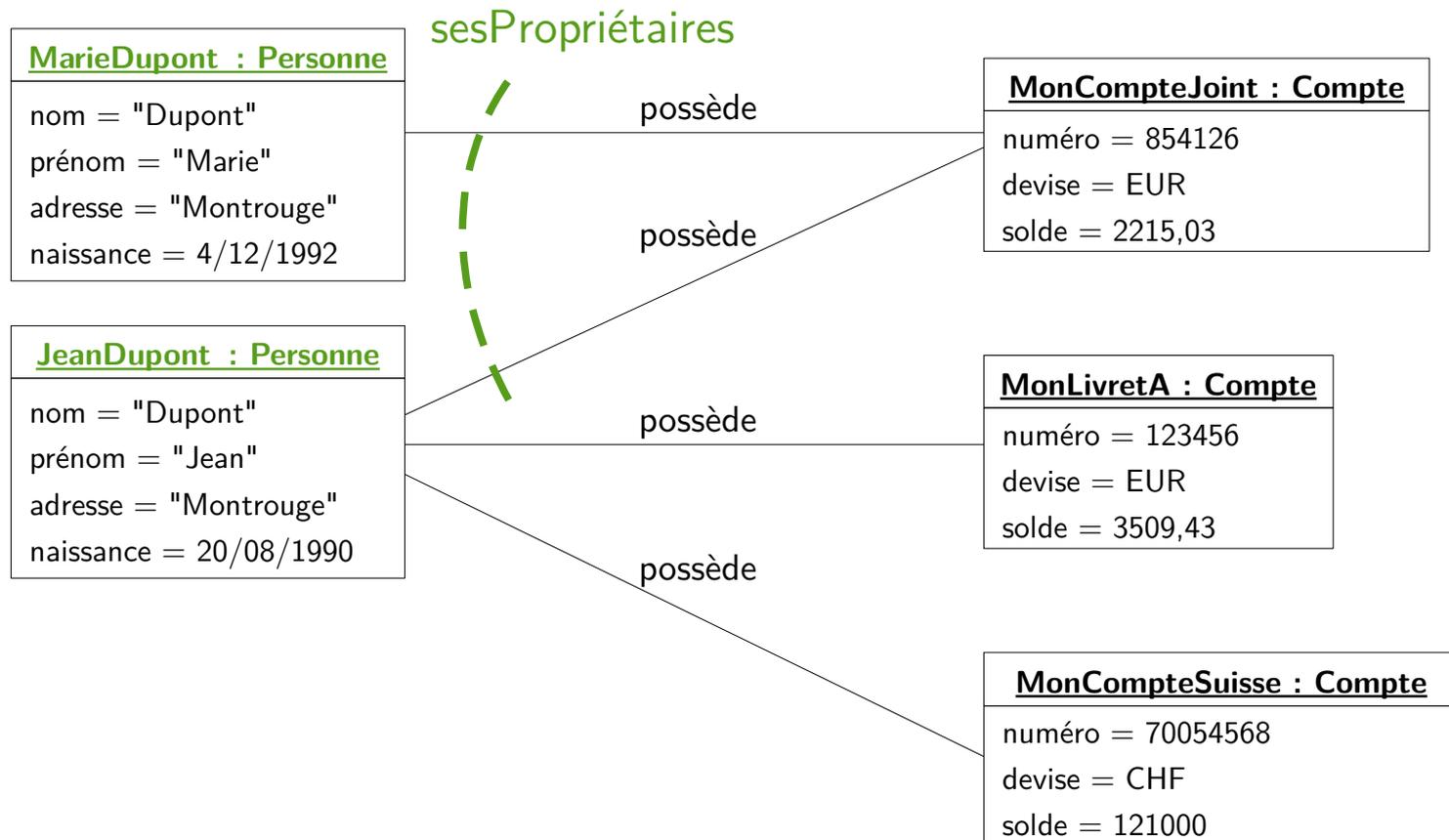
- relation binaire (en général)
- au plus un lien entre deux objets (pour une association)



Relations entre objets

Lien entre objets

- relation binaire (en général)
- au plus un lien entre deux objets (pour une association)

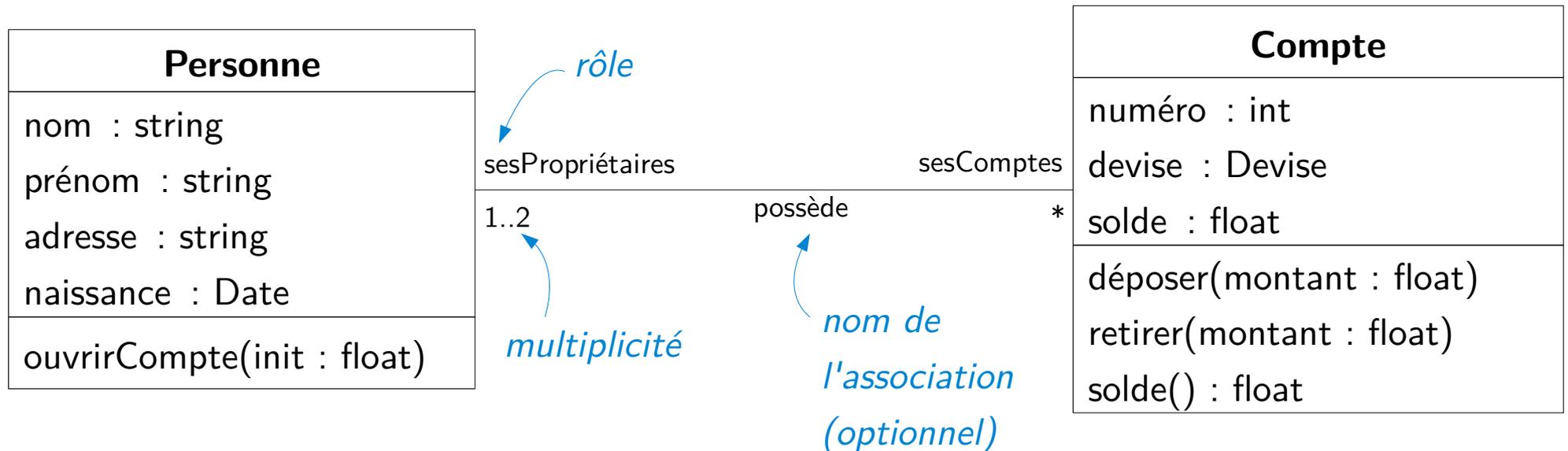


Relations entre classes

Association entre classes : Relation binaire (en général)

Rôle : Nomme l'extrémité d'une association, permet d'**accéder aux objets liés** par l'association à un objet donné

Multiplicité : Contraint le **nombre d'objets liés** par l'association

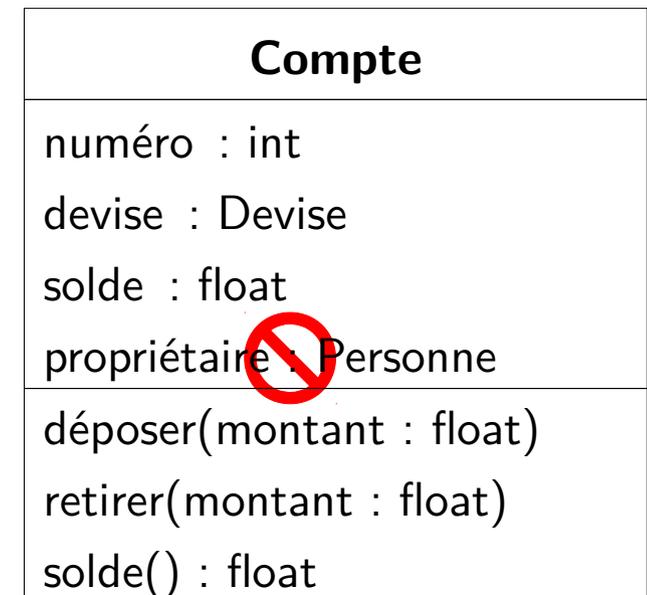


Lien = instance d'association

Attribut et association

Rappel : Types des attributs simple, primitif ou énuméré

En particulier, **pas d'attribut dont le type est une classe** du diagramme

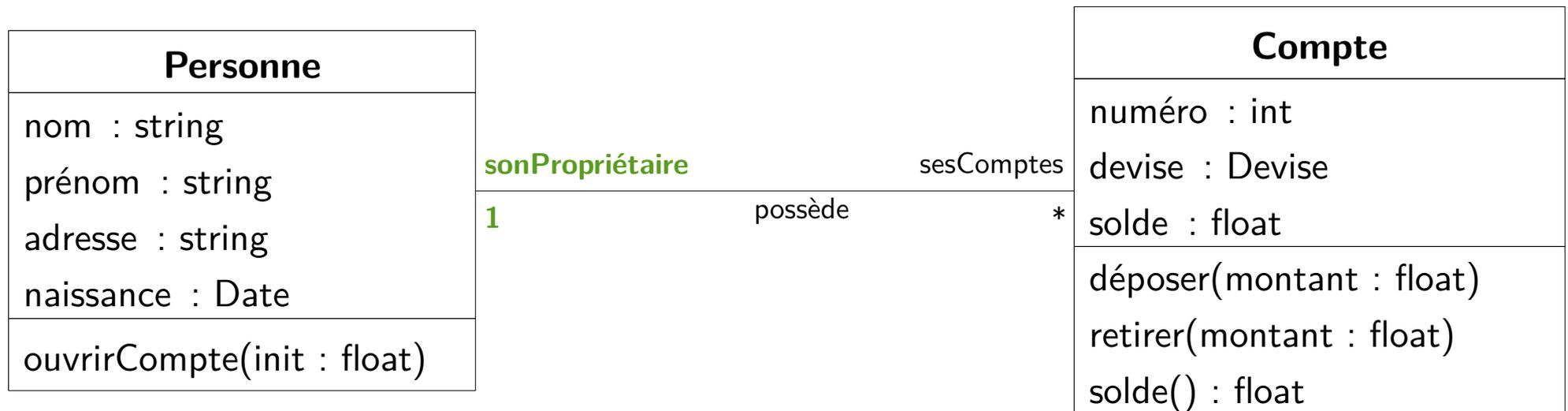


Attribut et association

Rappel : Types des attributs simple, primitif ou énuméré

En particulier, **pas d'attribut dont le type est une classe** du diagramme

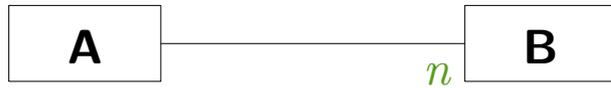
Mais **association vers cette classe**



Multiplicités

Nombre d'objets de la classe B associés à un objet de la classe A

Exactement n



Exactement n ou m ou p



Entre n et m



Au moins n



Plusieurs (0 ou plus)

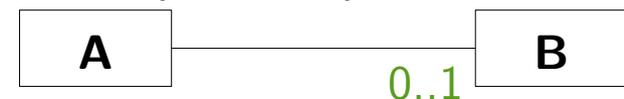


En pratique

Exactement 1



Au plus 1 (0 ou 1)



Au moins 1 (jamais 0)



0 ou plus



Hiérarchie de classes

Principe : Regrouper les classes partageant des attributs et des opérations et les organiser en arborescence

Spécialisation : raffinement d'une classe en une sous-classe

Généralisation : abstraction d'un ensemble de classes en super-classe

CompteCourant
numéro : int
devise : Devise
solde : float
découvertAutorisé : float
fraisDécouvert : float
déposer(montant : float)
retirer(montant : float)
solde() : float

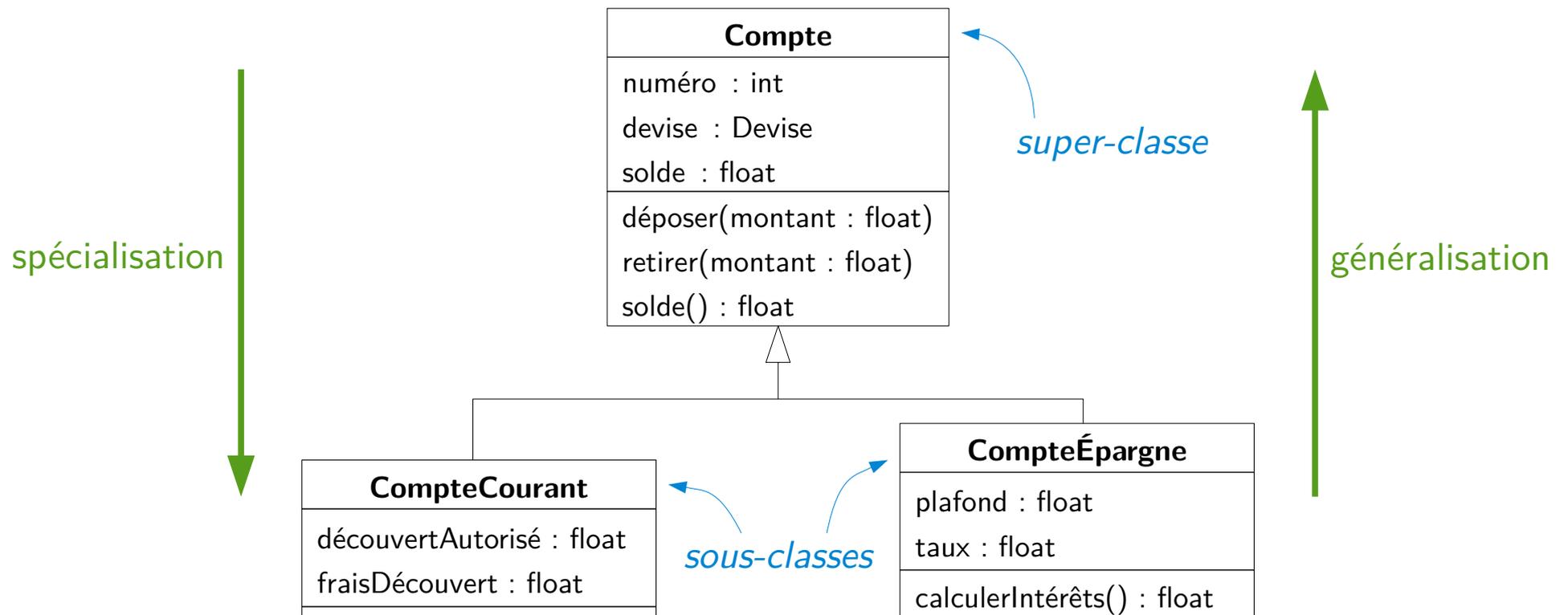
CompteÉpargne
numéro : int
devise : Devise
solde : float
plafond : float
taux : float
déposer (montant : float)
retirer(montant : float)
solde() : float
calculerIntérêts() : float

Hiérarchie de classes

Principe : Regrouper les classes partageant des attributs et des opérations et les organiser en arborescence

Spécialisation : raffinement d'une classe en une sous-classe

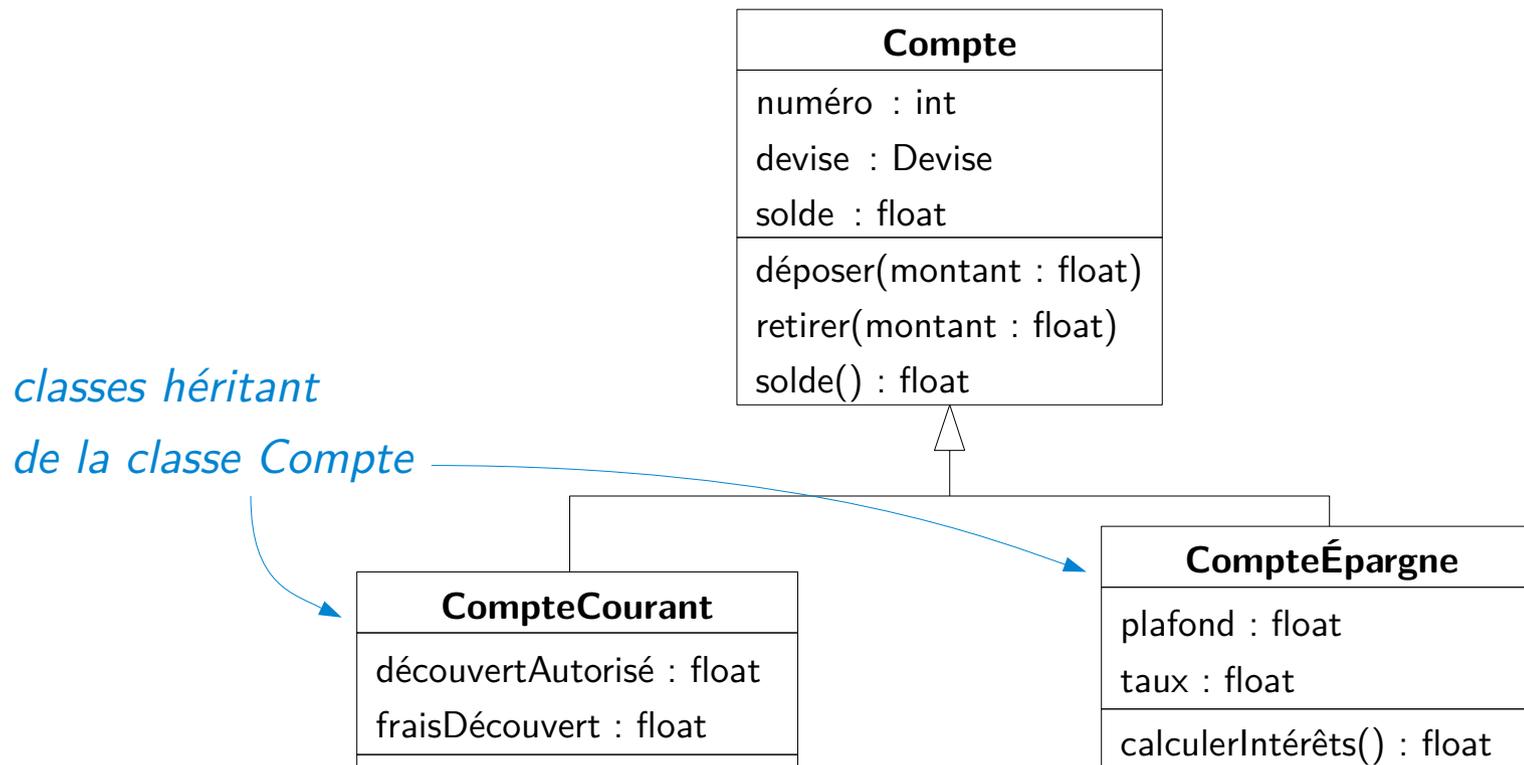
Généralisation : abstraction d'un ensemble de classes en super-classe



Hiérarchie de classes

Principe : Regrouper les classes partageant des attributs et des opérations et les organiser en arborescence

Héritage : Construction d'une classe à partir d'une classe plus haute dans la hiérarchie (partage des attributs, opérations, contraintes...)



Hiérarchie de classes

Diagramme de classes

CompteCourant
numéro : int
devise : Devise
solde : float
découvertAutorisé : float
fraisDécouvert : float
déposer(montant : float)
retirer(montant : float)
solde() : float

CompteÉpargne
numéro : int
devise : Devise
solde : float
plafond : float
taux : float
déposer (montant : float)
retirer(montant : float)
solde() : float
calculerIntérêts() : float

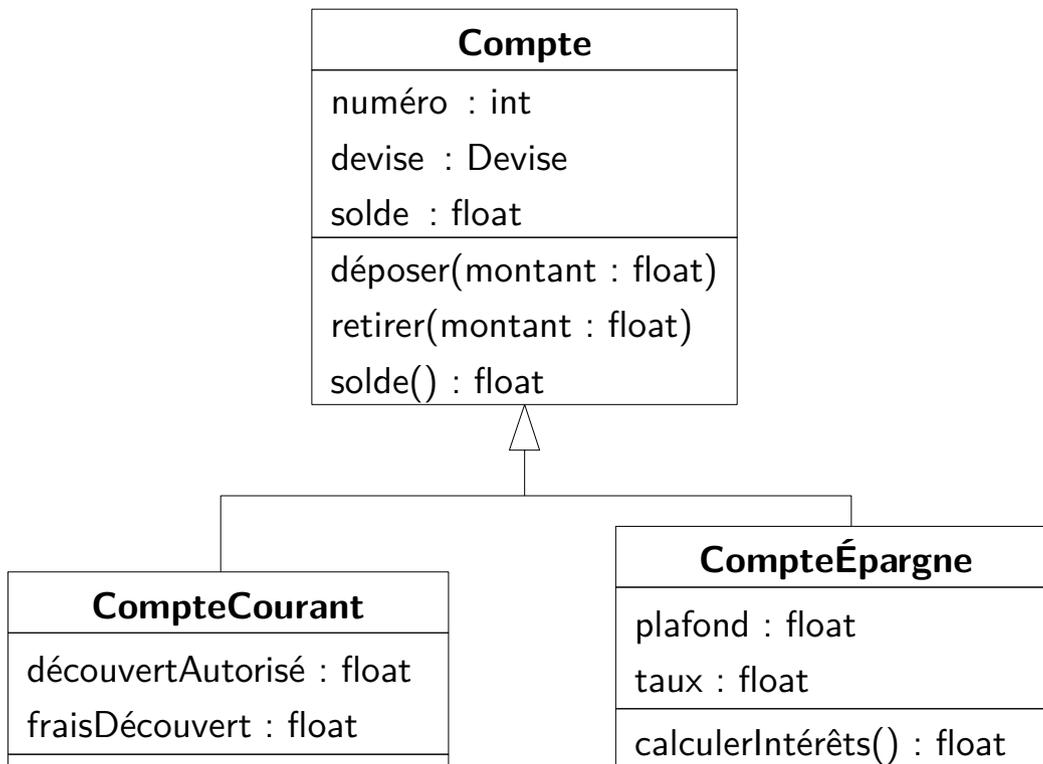
Exemples d'objets

<u>CC:CompteCourant</u>
numéro = 875421
devise = EUR
solde = 1290,30
découvertAutorisé = -200,00
fraisDécouvert = 2,30

<u>LivA:CompteÉpargne</u>
numéro = 094435
devise = EUR
solde = 10542,00
plafond = 22950,00
taux = 0,75

Hiérarchie de classes

Diagramme de classes



Exemples d'objets

C1:Compte

numéro = 463527
devise = EUR
solde = 213,50

CC:CompteCourant

numéro = 875421
devise = EUR
solde = 1290,30
découvertAutorisé = -200,00
fraisDécouvert = 2,30

LivA:CompteÉpargne

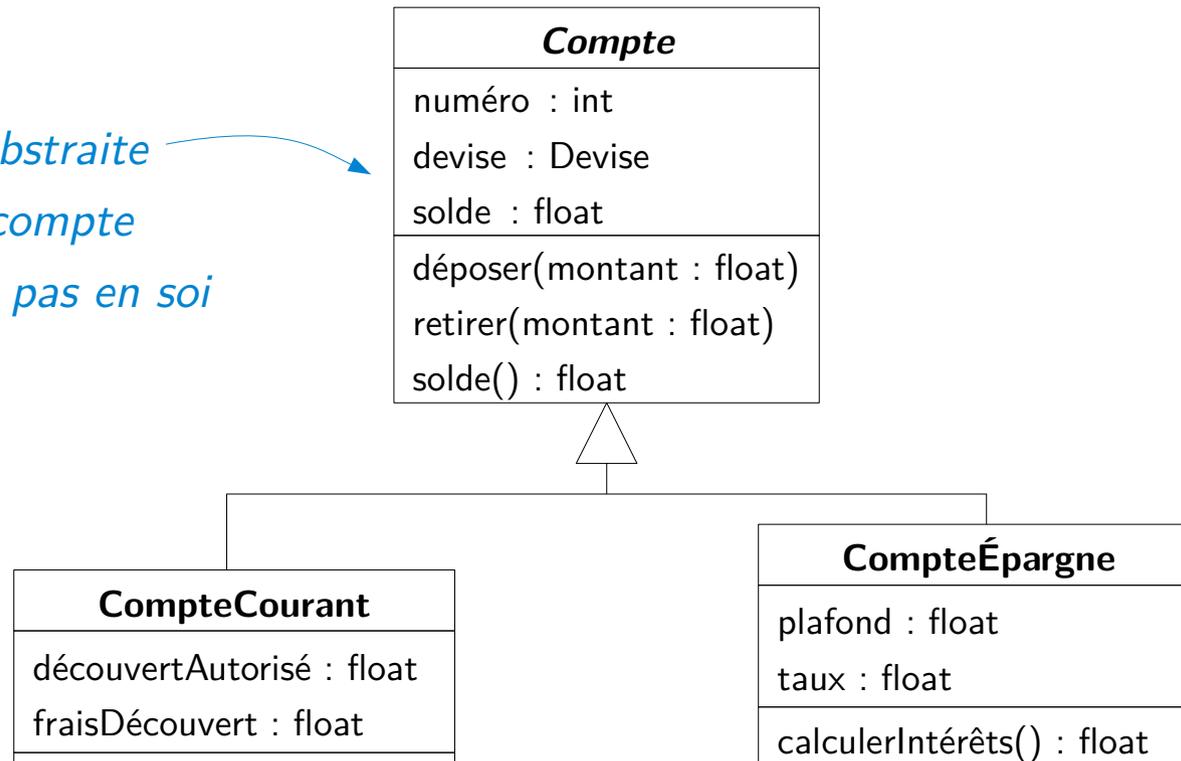
numéro = 094435
devise = EUR
solde = 10542,00
plafond = 22950,00
taux = 0,75

Classe abstraite

Classe sans instance, seulement une base pour classes héritées

Notation : nom de la classe en italique (ou stéréotype « abstract »)

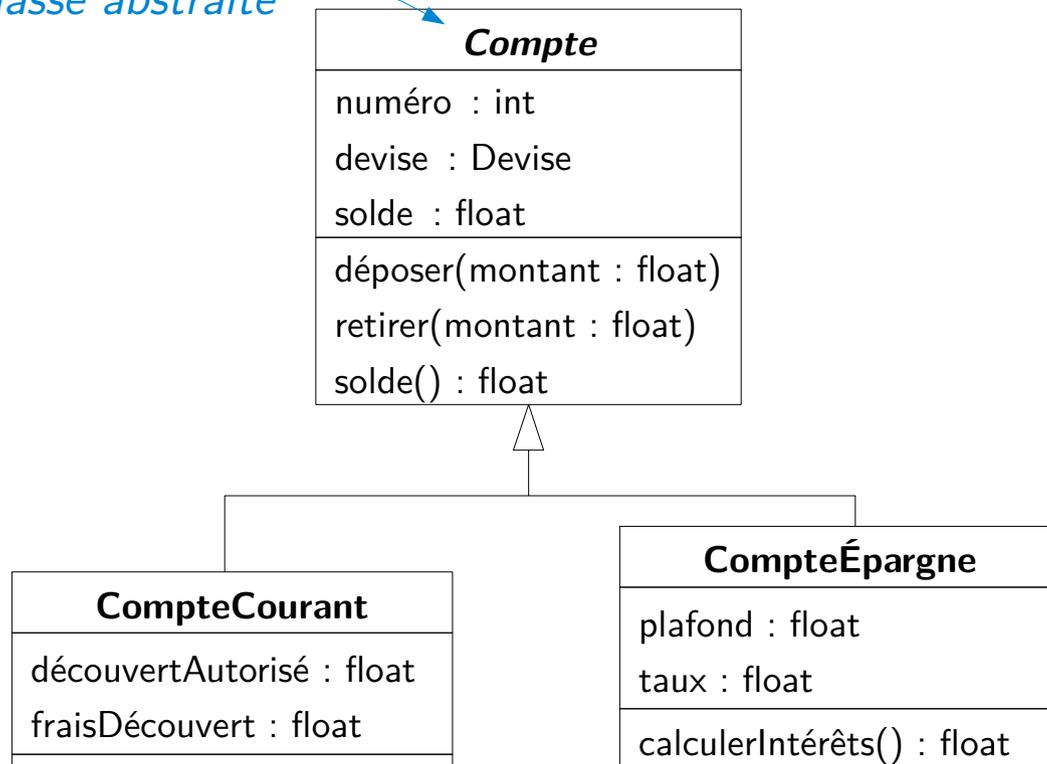
*classe abstraite
car un compte
n'existe pas en soi*



Hiérarchie de classes

Diagramme de classes

classe abstraite



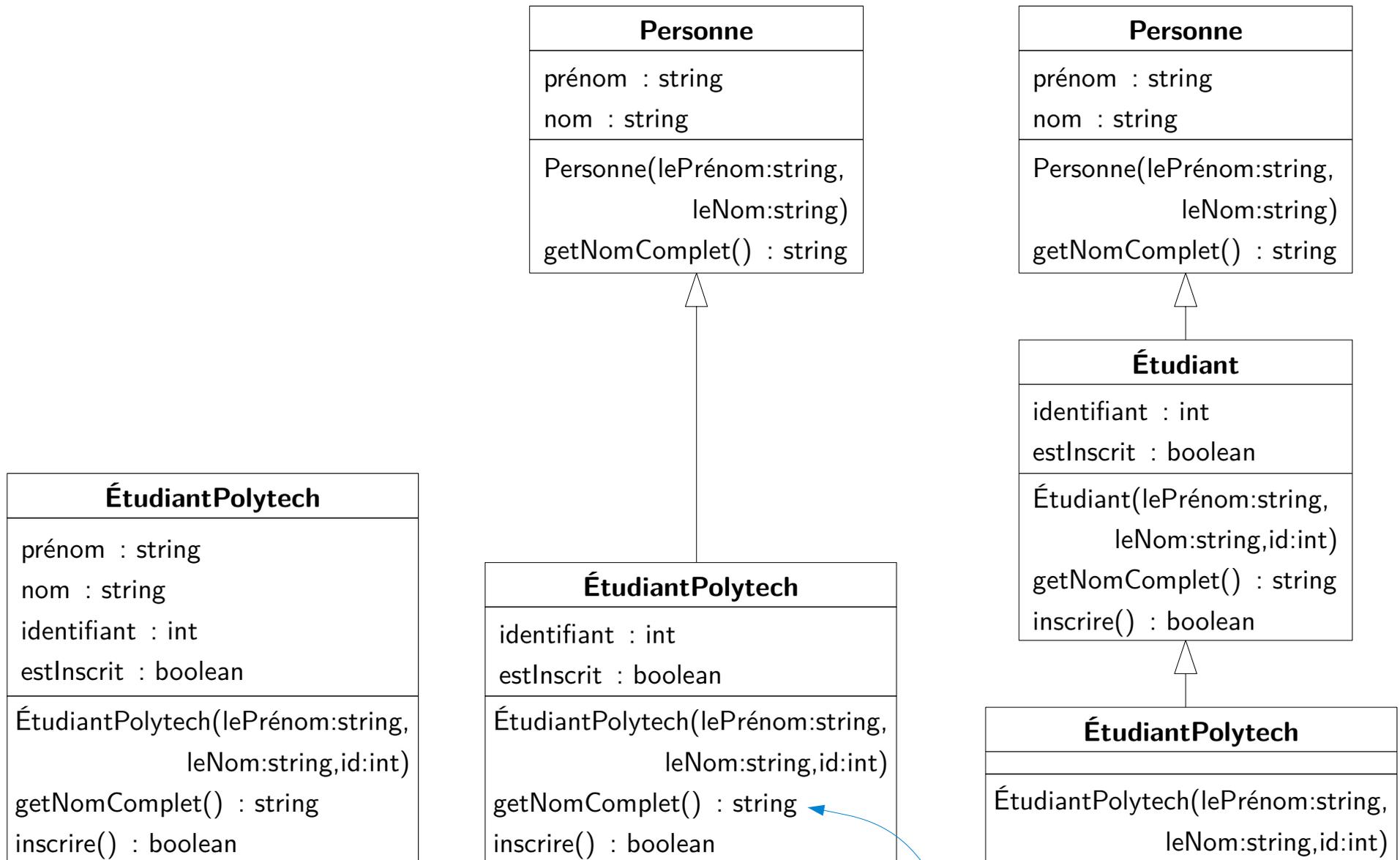
Exemples d'objets

<u>C1:Compte</u>
numéro = 463527
devise = EUR
solde = 213,50

<u>CC:CompteCourant</u>
numéro = 875421
devise = EUR
solde = 1290,30
découvertAutorisé = -200,00
fraisDécouvert = 2,30

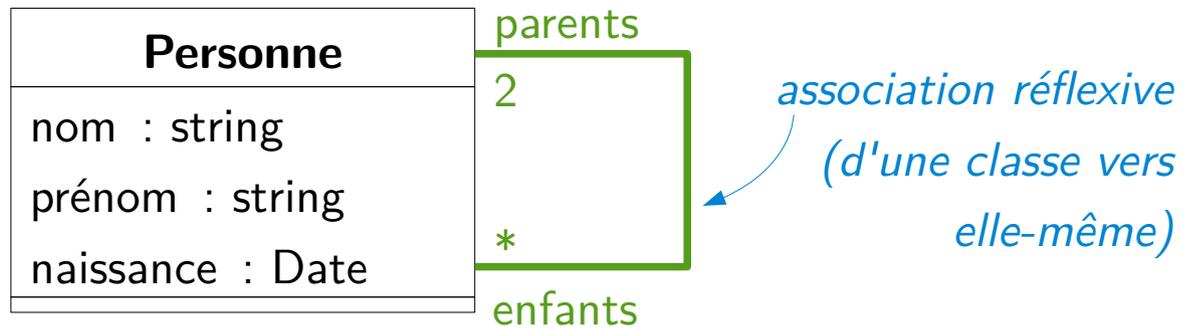
<u>LivA:CompteÉpargne</u>
numéro = 094435
devise = EUR
solde = 10542,00
plafond = 22950,00
taux = 0,75

Exemple tiré du cours de Java

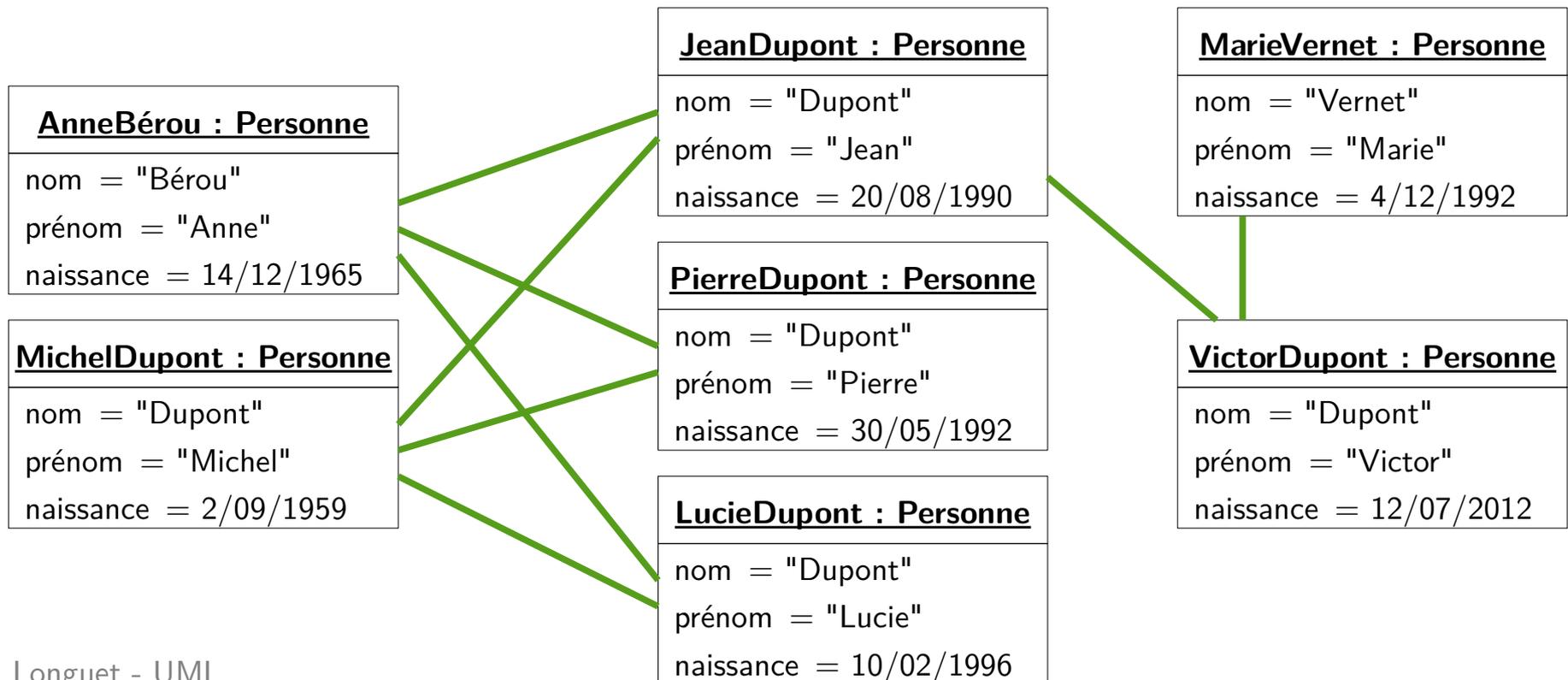


Association réflexive

Diagramme de classes



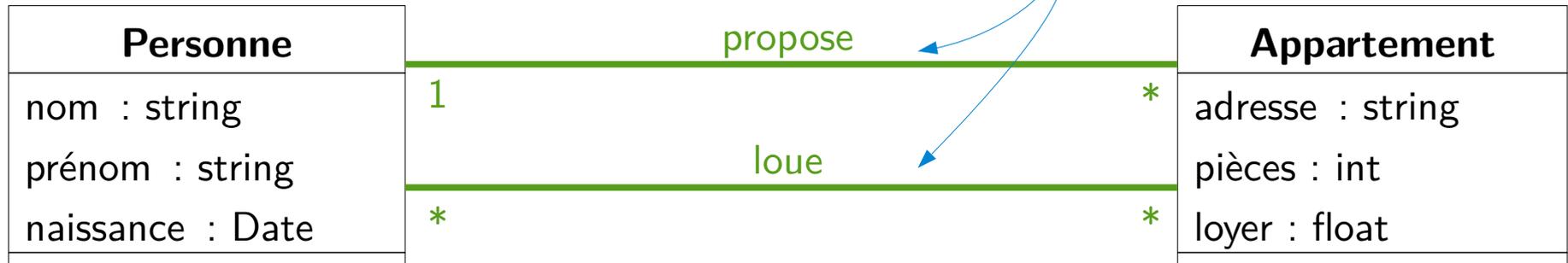
Exemple de diagramme d'objets



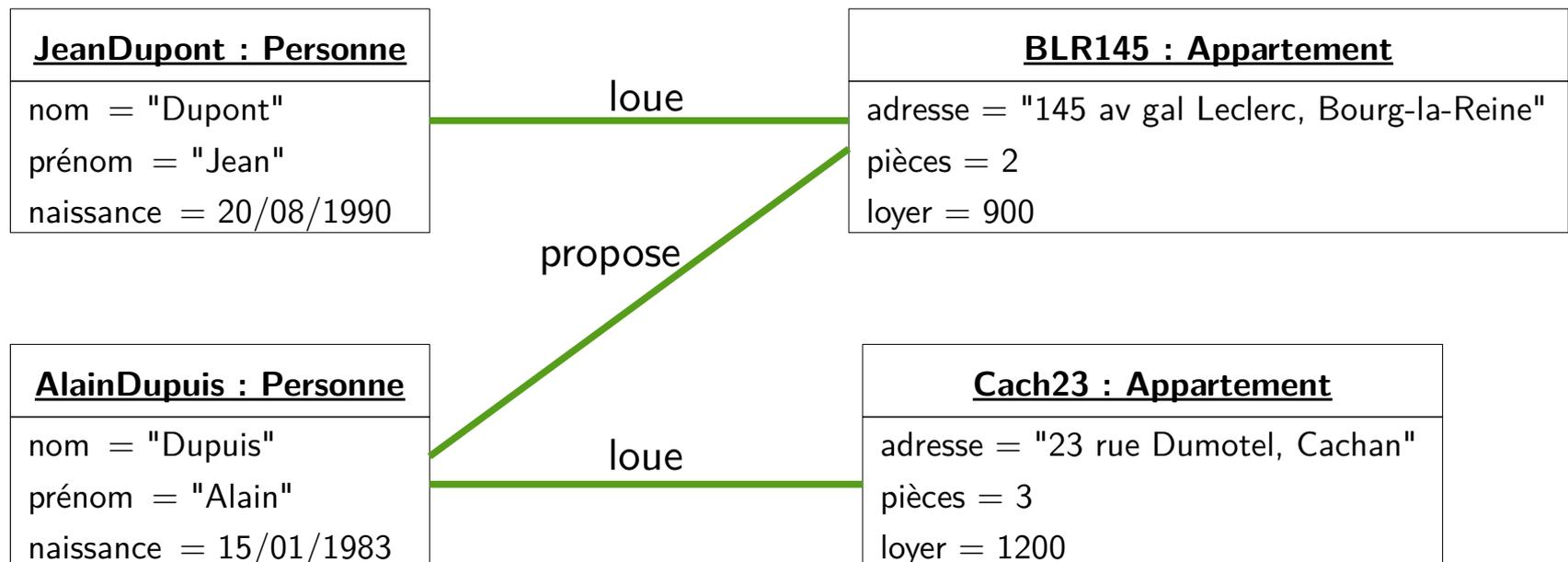
Association multiple

Diagramme de classes

*plusieurs associations
entre deux classes*



Exemple de diagramme d'objets



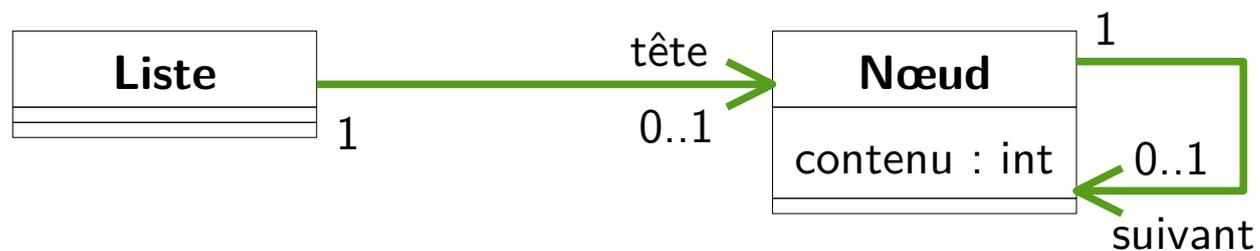
Navigabilité

Orientation d'une association

- Restreint l'accessibilité des objets
- Depuis un A, on a accès aux objets de B qui lui sont associés, mais pas l'inverse



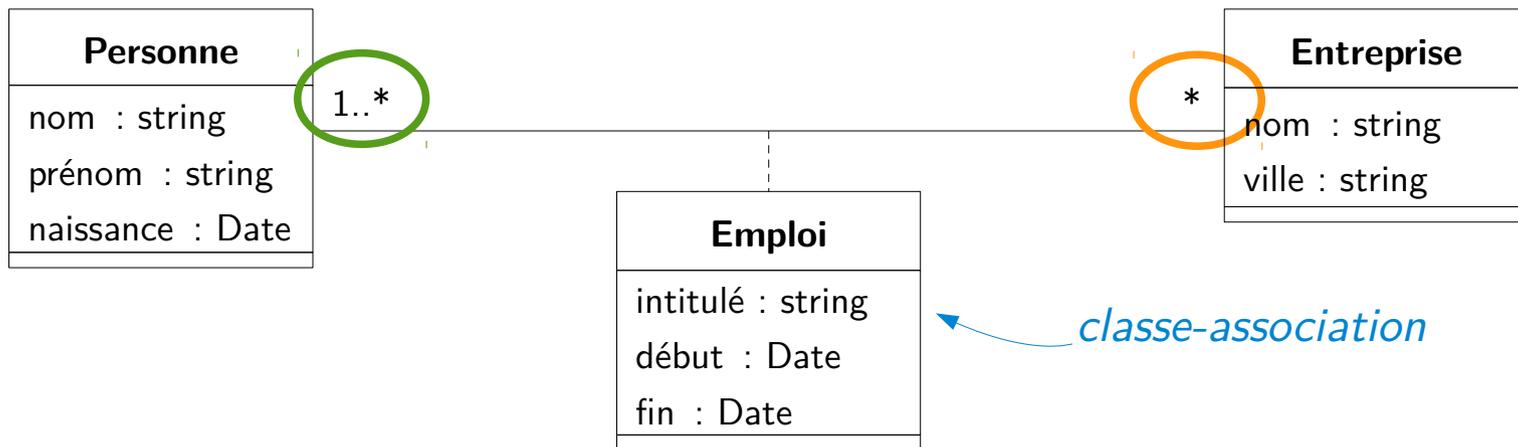
Exemple (listes chaînées)



Par défaut, associations navigables dans les deux sens (pas de flèche)

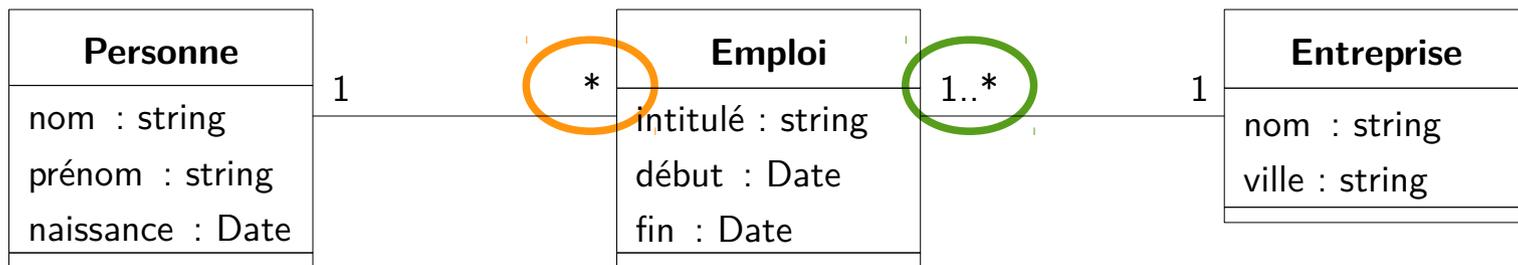
Classe-association

Permet de paramétrer une association entre deux classes par une classe



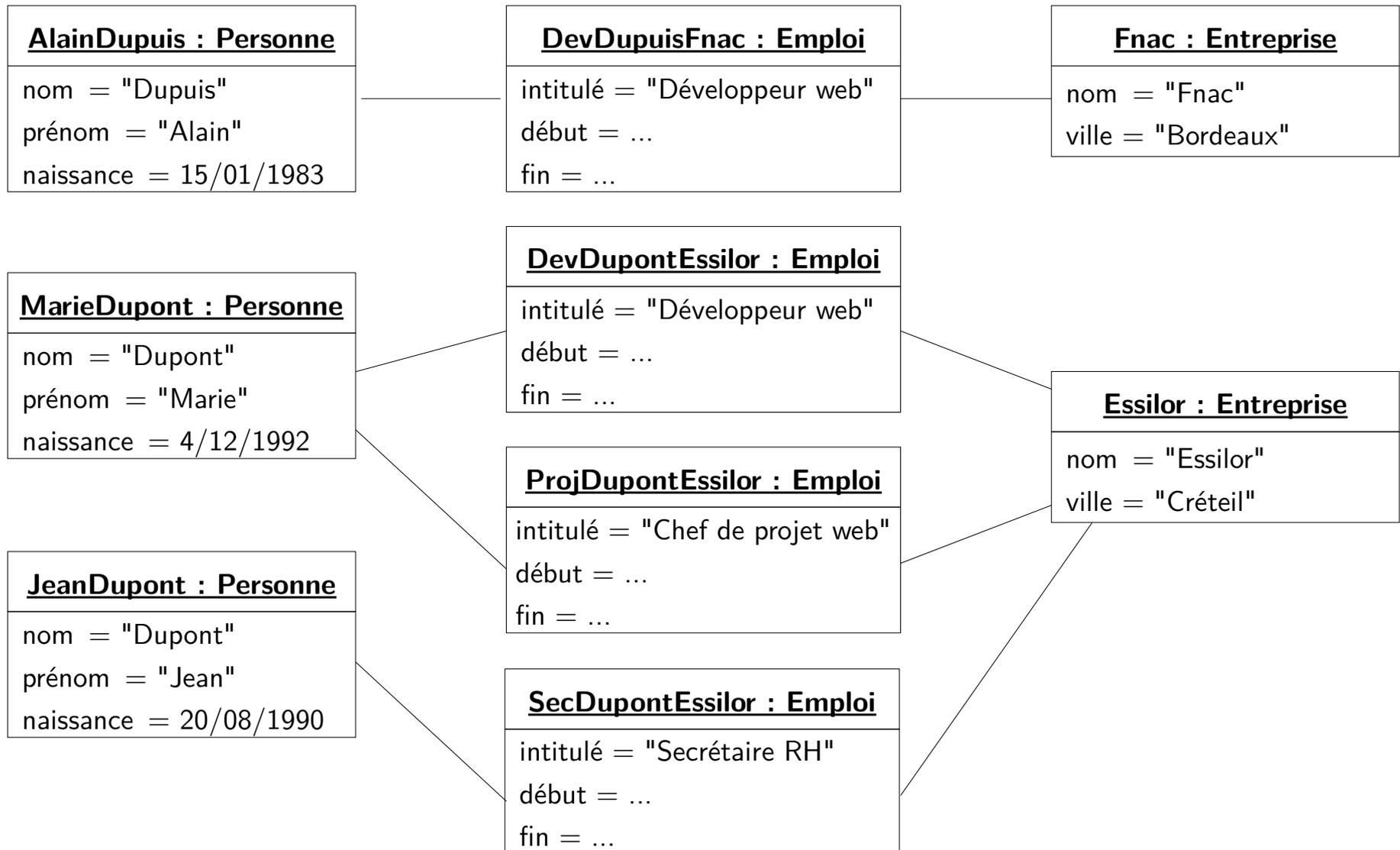
Instance unique de la classe-association pour chaque lien entre objets

Équivalence en termes de classes et d'associations :



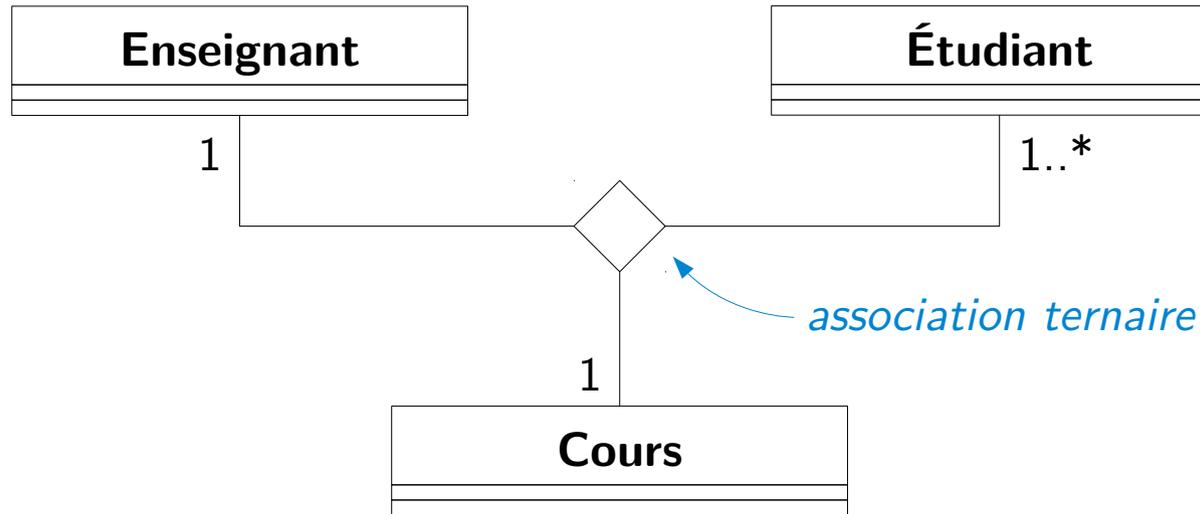
Classe-association

Exemple de diagramme d'objets

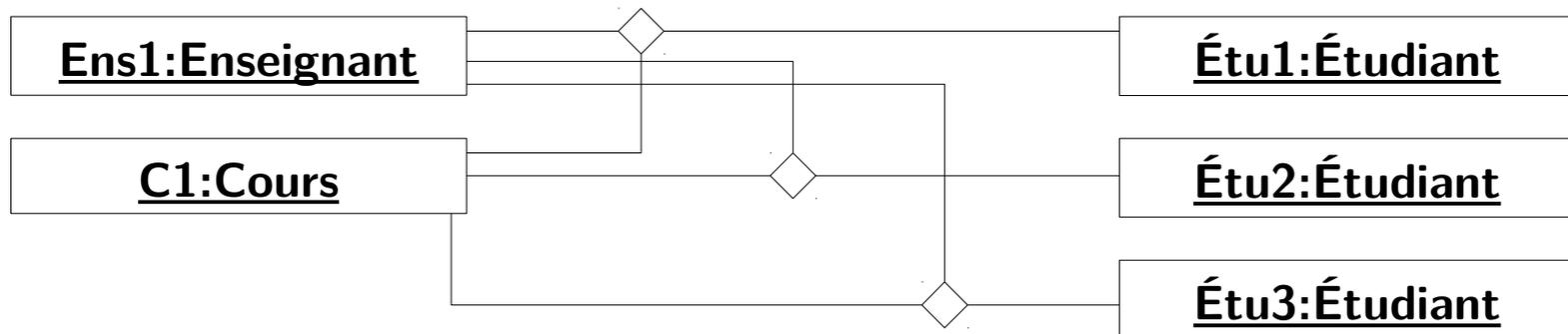


Association n -aire

Association reliant plus de deux classes

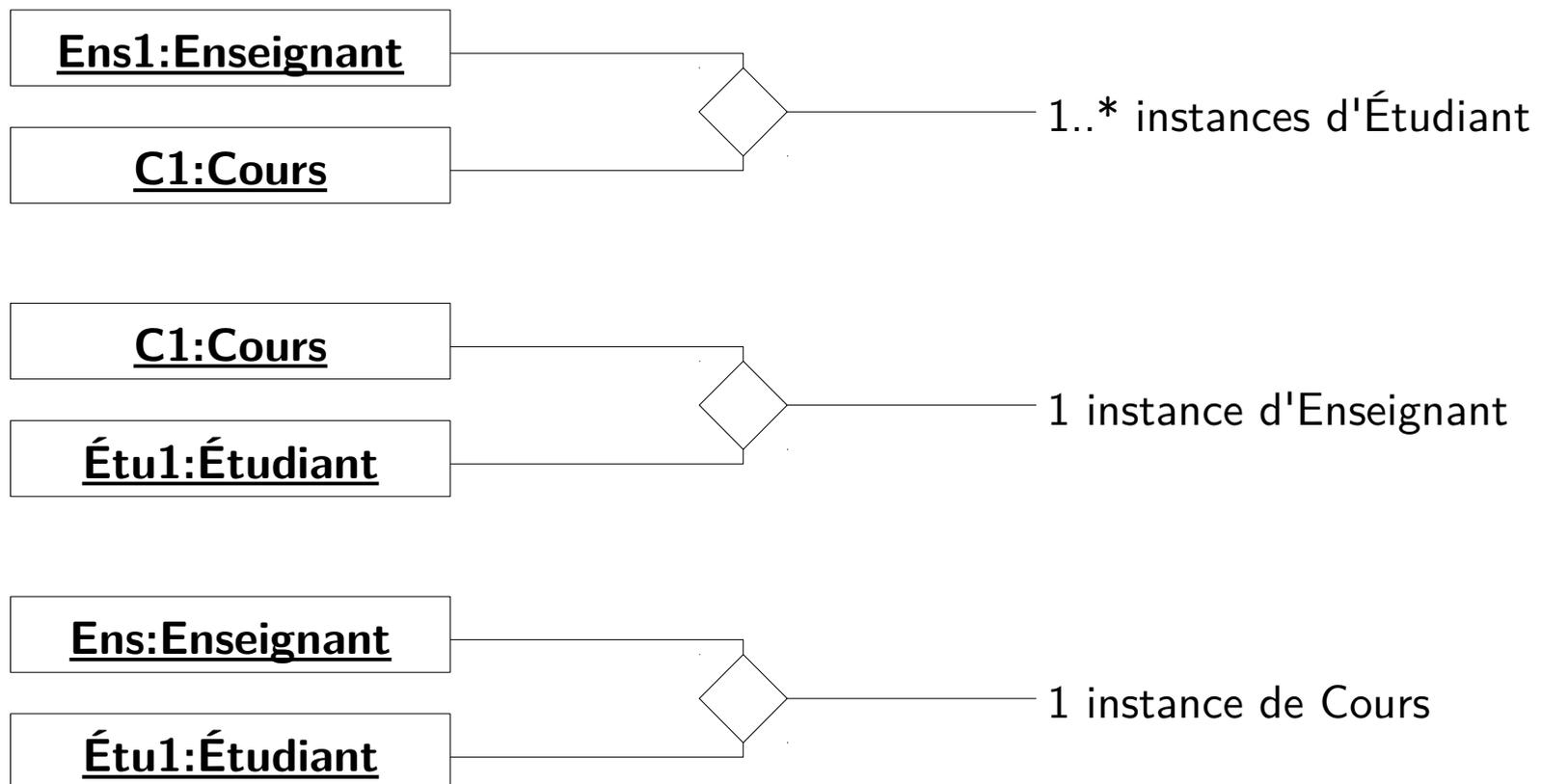


Instance d'une association n -aire = lien entre n objets



Association n -aire

Multiplicités : pour chaque $(n-1)$ -uplet d'objets, contraint le nombre d'objets qui lui sont associés



Agrégation

Association particulière entre classes

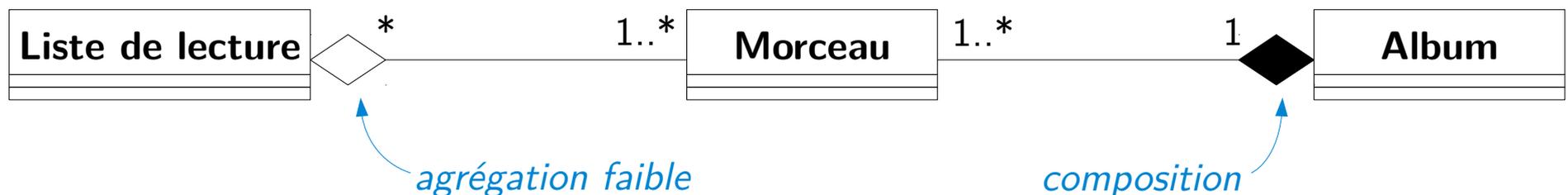
- **Dissymétrique** : une classe prédominante sur l'autre
- Relation de type **composant-composite**

Deux types d'agrégation

- Agrégation faible
- Composition

Exemple

Lecteur de contenu audio permettant de créer des listes de lecture



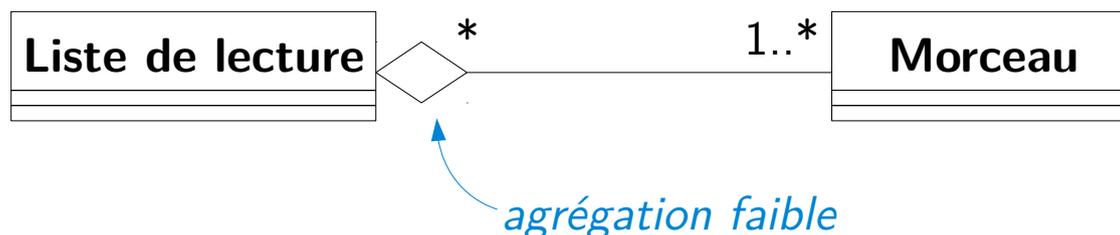
Agrégation faible

Agrégation par référence

- Le composite **fait référence** à ses composants
- La création ou destruction du composite est **indépendante** de la création ou destruction de ses composants
- Un objet peut **faire partie de plusieurs composites** à la fois

Exemple

- Une liste de lecture est composée d'un ensemble de morceaux
- Un morceau peut appartenir à plusieurs listes de lecture
- Supprimer la liste ne supprime pas les morceaux



Composition

Agrégation par valeur

- Le composite **contient** ses composants
- La création ou destruction du composite **entraîne** la création ou destruction de ses composants
- Un objet ne **fait partie que d'un composite** à la fois

Exemple

- Un morceau n'appartient qu'à un album
- La suppression de l'album entraîne la suppression de tous ses morceaux

