

FORMATION UML



ANTHONY DI PERSIO

UML

Diagrammes de classes

1. Classes et associations

Processus de développement en V

Analyse

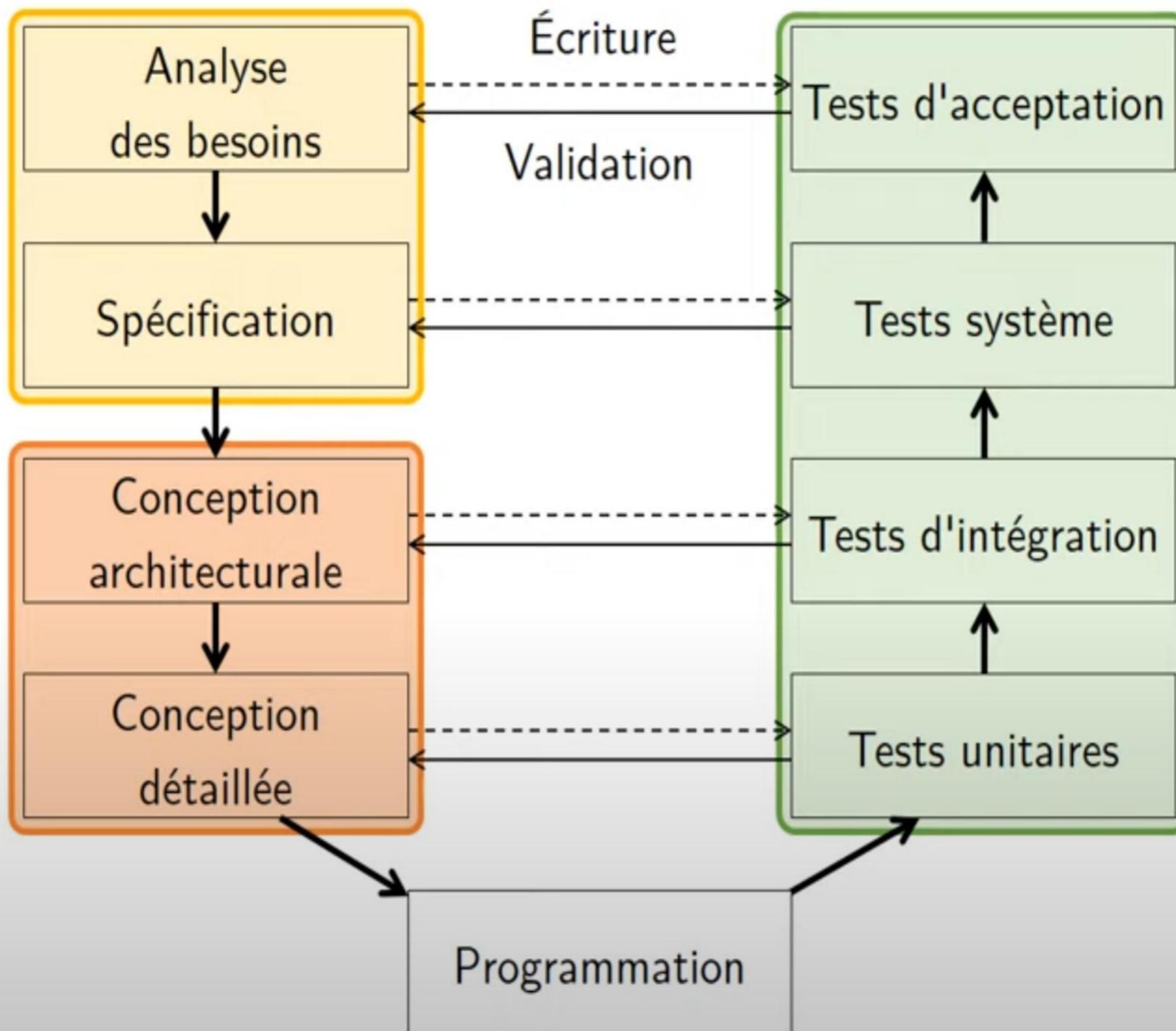
« que doit faire
le système ? »

Conception

« comment
va-t-il le
réaliser ? »

Validation
et vérification

« fait-il ce qui
était demandé
et le fait-il
correctement ? »



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

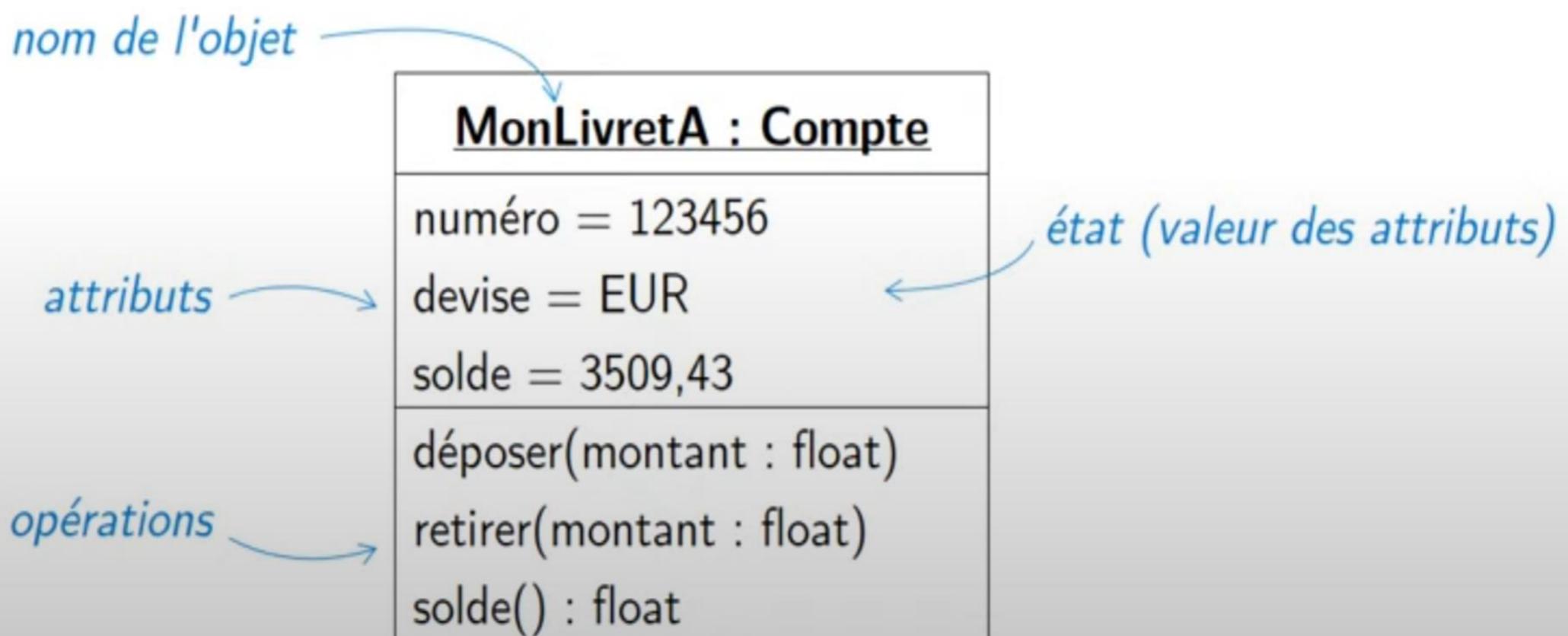
Diagramme d'objets

- 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)

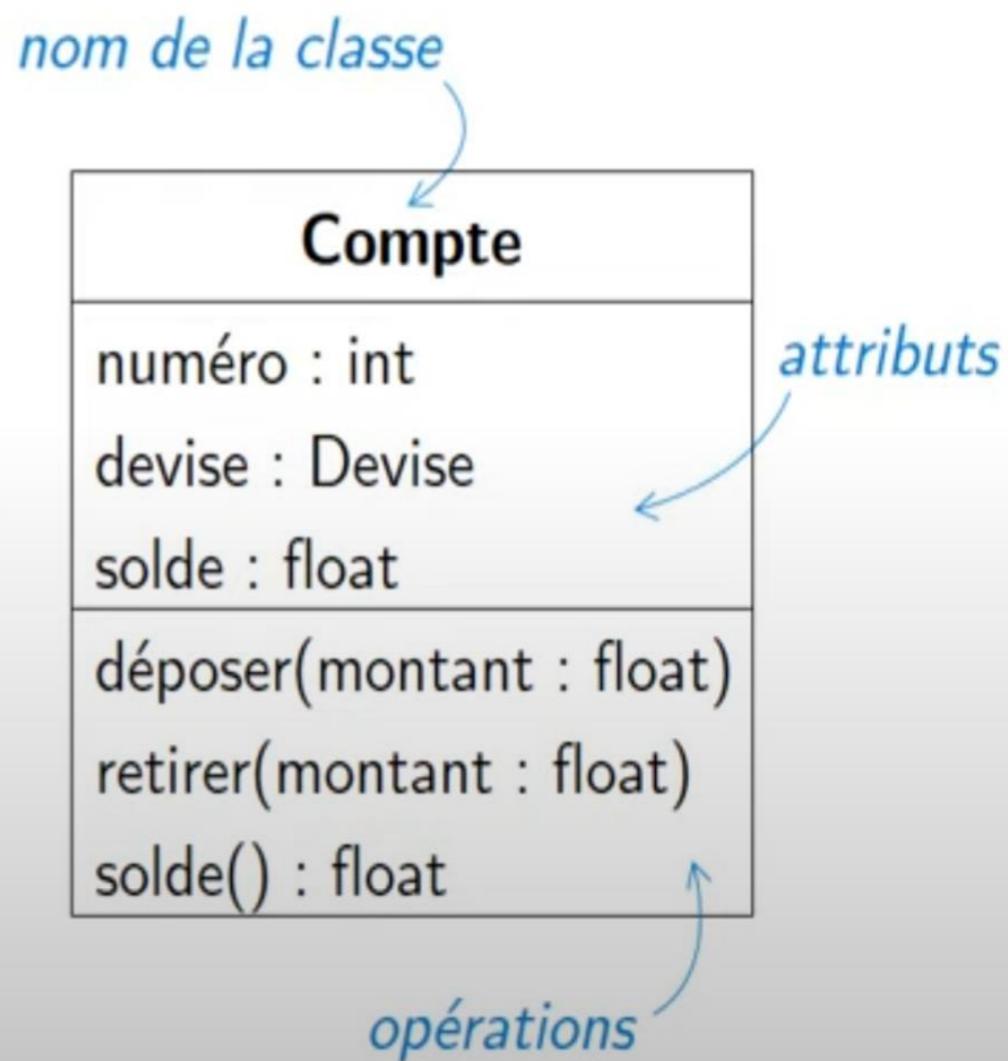
Objet = instance d'une classe

MonLivreA : Compte
numéro = 123456
devise = EUR
solde = 3509,43
déposer(montant : float)
retirer(montant : float)
solde() : float

MonCompteJoint : Compte
numéro = 854126
devise = EUR
solde = 2215,03
déposer(montant : float)
retirer(montant : float)
solde() : float

MonCompteSuisse : Compte
numéro = 70054568
devise = CHF
solde = 121000
déposer(montant : float)
retirer(montant : float)
solde() : float

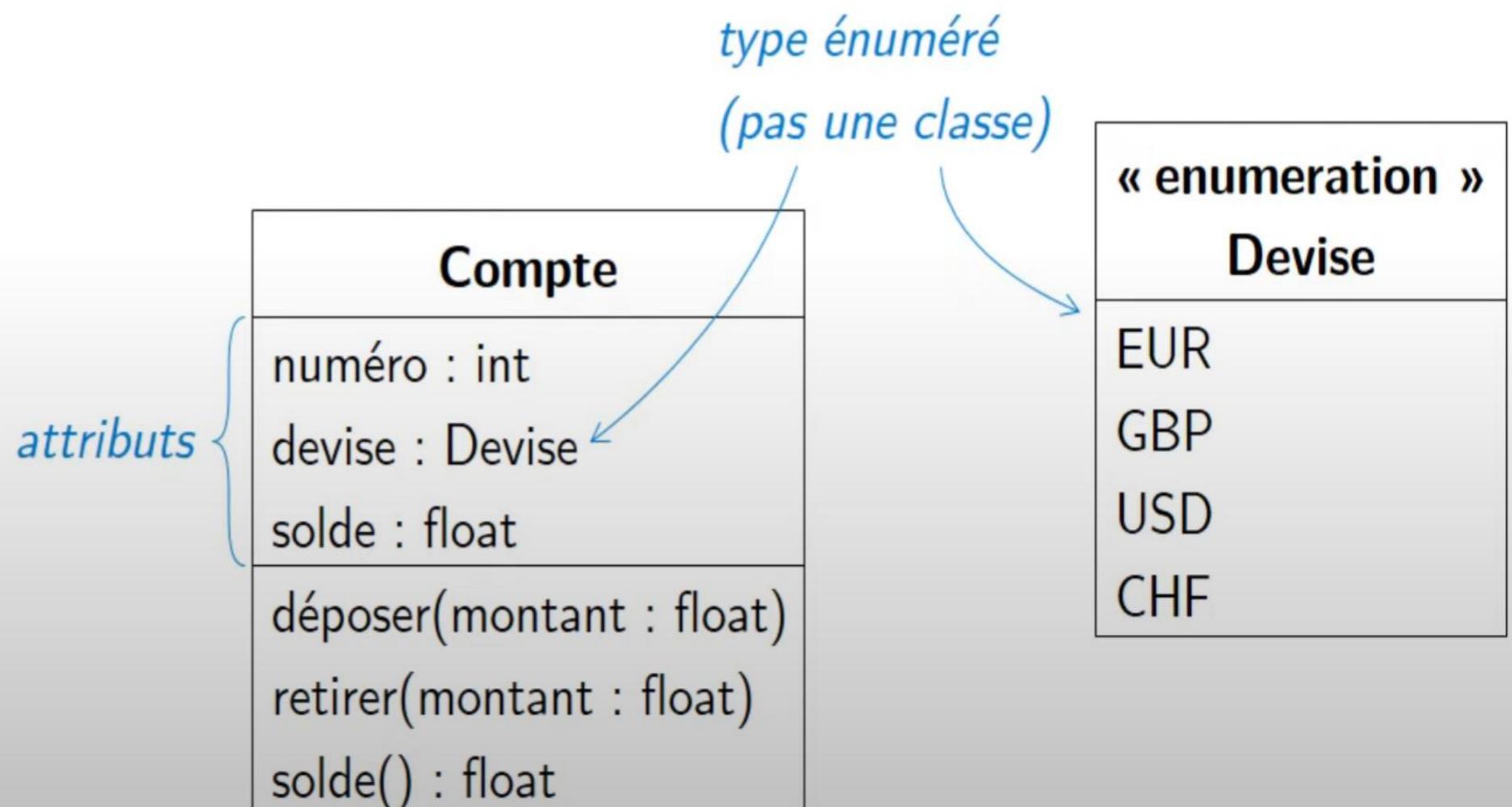
instances de



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



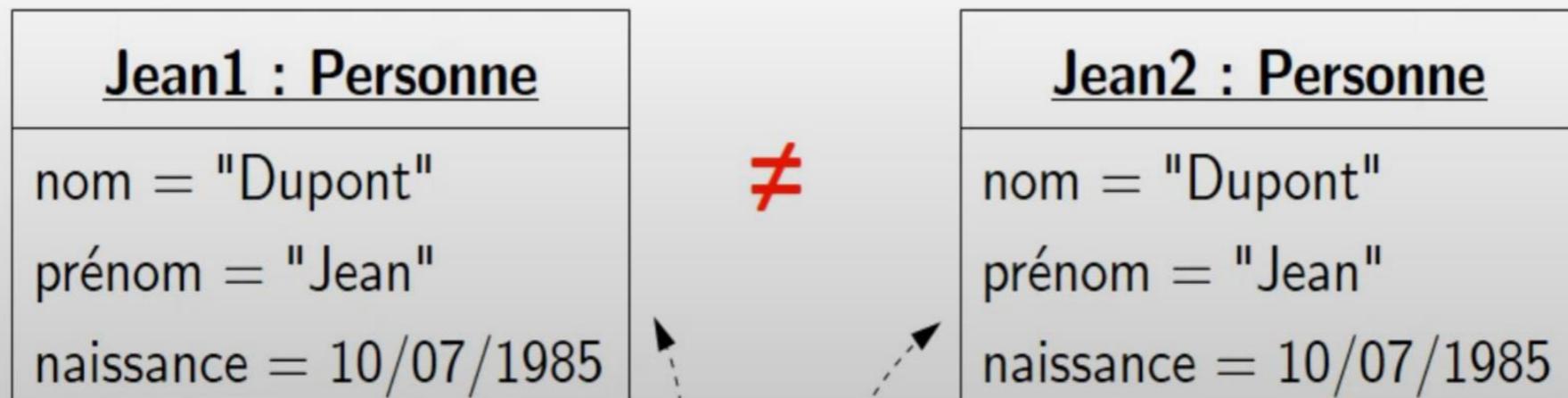
Classes

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

Compte	
	numéro : int
	devise : Devise
	solde : float
opérations {	déposer(montant : float)
	retirer(montant : float)
	solde() : float

Exemple de la bibliothèque

On cherche à développer un système qui gère les emprunts et les retours dans une bibliothèque.

La bibliothèque gère des livres et des revues. Un livre est caractérisé par son titre, son auteur et son code ISBN. Un numéro de revue est caractérisé par le titre de la revue, un numéro de volume et sa date de parution. Chaque exemplaire d'une ressource est caractérisé par un code barre au sein de la bibliothèque.

Pour emprunter un ouvrage, un utilisateur doit être enregistré. Il s'enregistre auprès du bibliothécaire en donnant son nom et une caution. Chaque ouvrage a une caution. Un utilisateur ne peut emprunter un ouvrage que si la caution qui lui reste sur son compte est supérieure à la caution de l'ouvrage. La durée de l'emprunt est fixée à 15 jours.

On ne peut pas emprunter plus d'un exemplaire d'une même ressource, ni emprunter une nouvelle ressource si on est en retard pour rendre une ressource.

L'emplacement de stockage d'un ouvrage dans la bibliothèque est représenté par un numéro de travée, un numéro d'étagère dans la travée, et un niveau. Différentes ressources peuvent être rangées au même emplacement, mais tous les exemplaires d'une même ressource sont stockés au même endroit.

Exemple de la bibliothèque (1)

Utilisateur
nom : string
caution : int

Livre
titre : string
auteur : string
ISBN : int
caution : int

Revue
titre : string
volume : int
parution : Date
caution : int

Emplacement
travée : int
étagère : int
niveau : int

Exemplaire
code_barre : int
retour : Date

Note : si un exemplaire n'est pas emprunté, retour à la valeur *null*

Relations entre objets

Lien entre objets

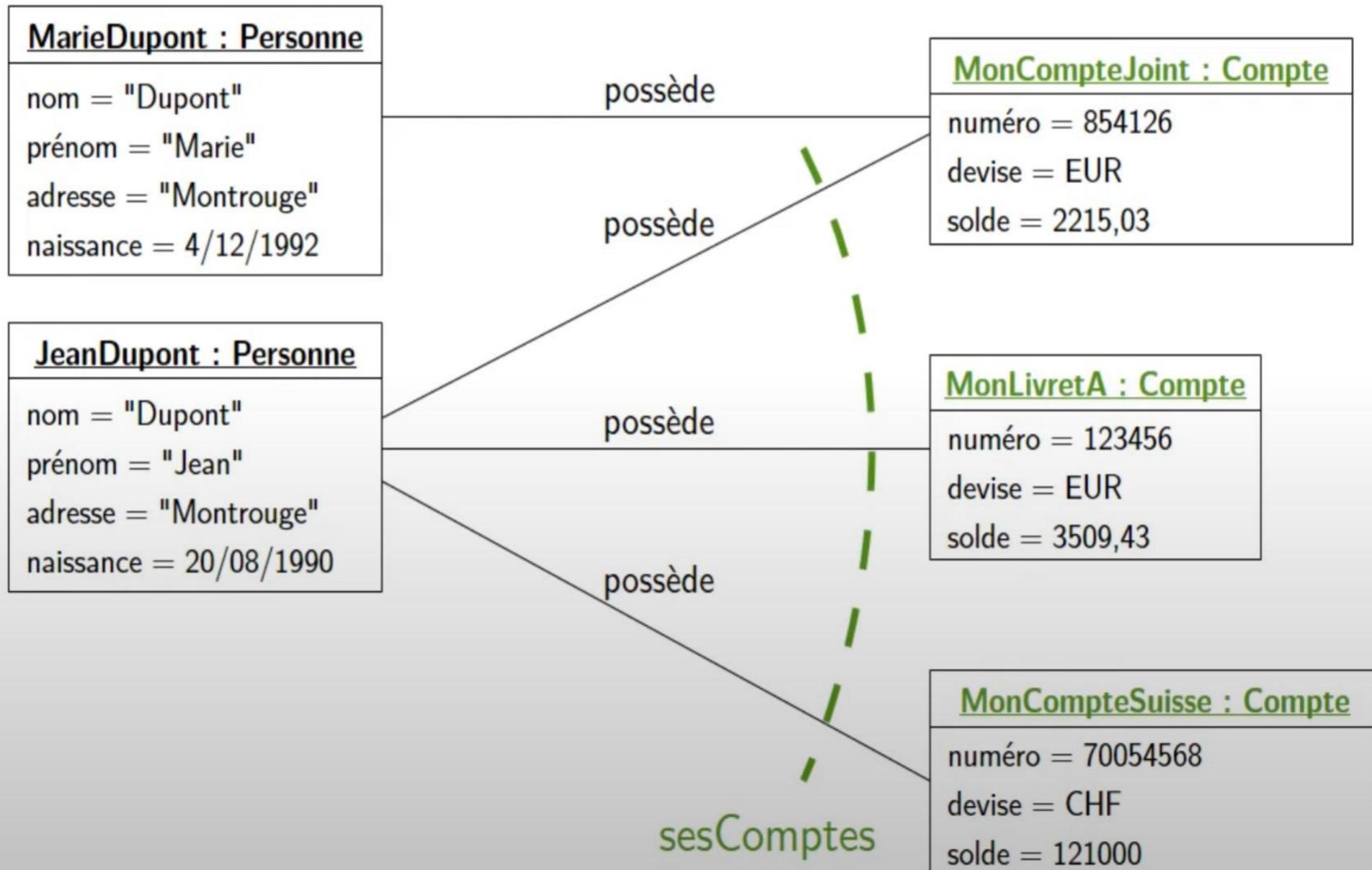
- relation binaire (en général)
- au plus un lien entre deux objets (pour une association)



Relations entre objets

Lien entre objets

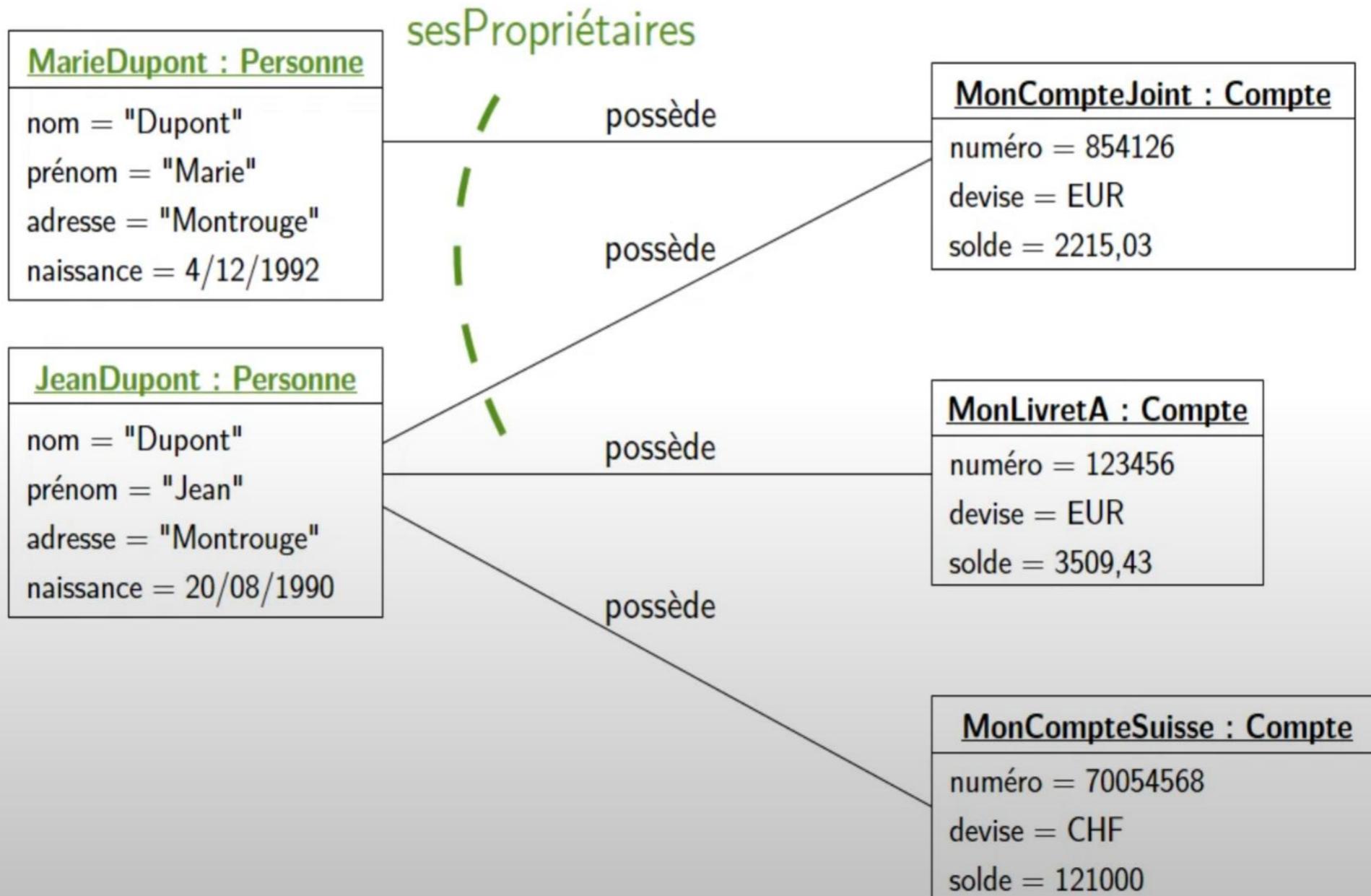
- relation binaire (en général)
- au plus un lien entre deux objets (pour une association)



Relations entre objets

Lien entre objets

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

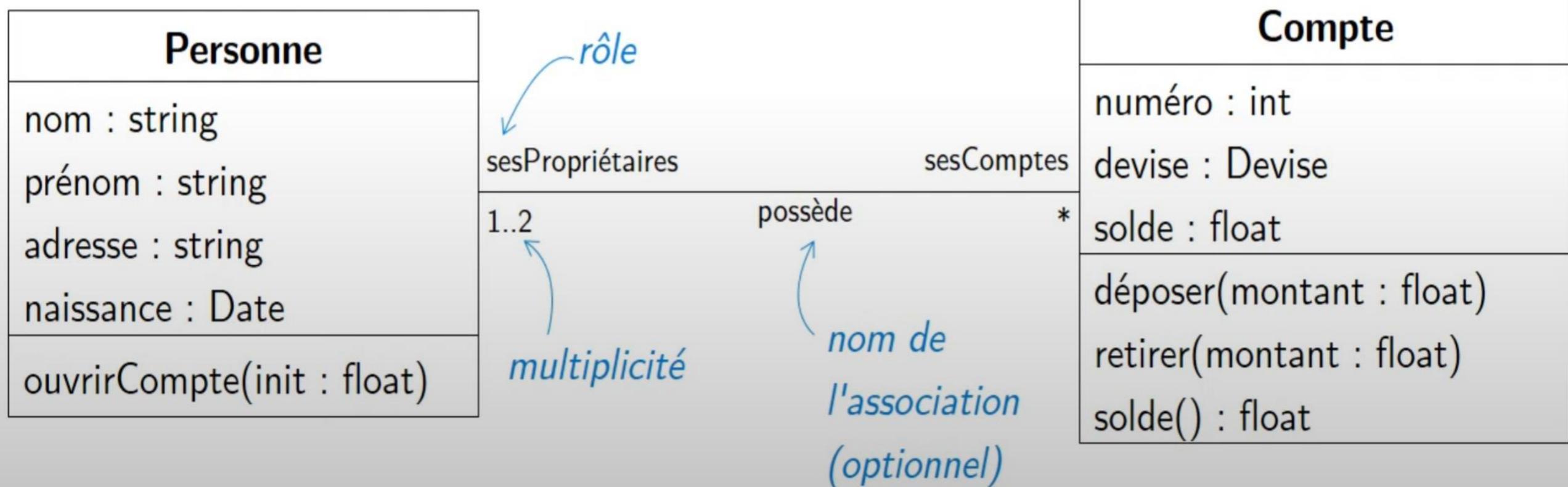


Relations entre classes

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



Attribut et association

Rappel : Types des attributs simple, primitif ou énuméré

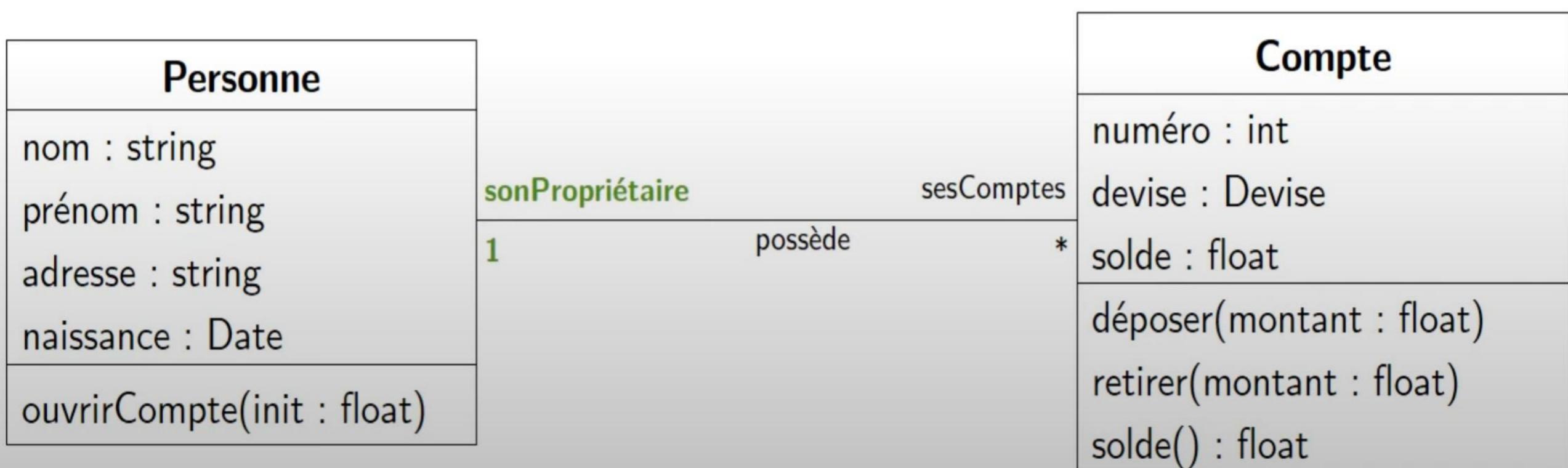
En particulier, pas d'attribut dont le type est une classe du diagramme

Compte
numéro : int
devise : Devise
solde : float
propriétaire  Personne
déposer(montant : float)
retirer(montant : float)
solde() : float

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

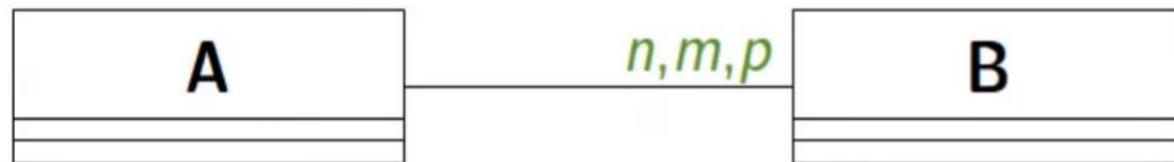


Multiplicités

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)

Multiplicités en pratique

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



Exactement 1



Au plus 1 (0 ou 1)



Au moins 1 (jamais 0)



0 ou plus

Exemple de la bibliothèque (2)

Utilisateur
nom : string
caution : int

Exemplaire
code_barre : int
retour : Date

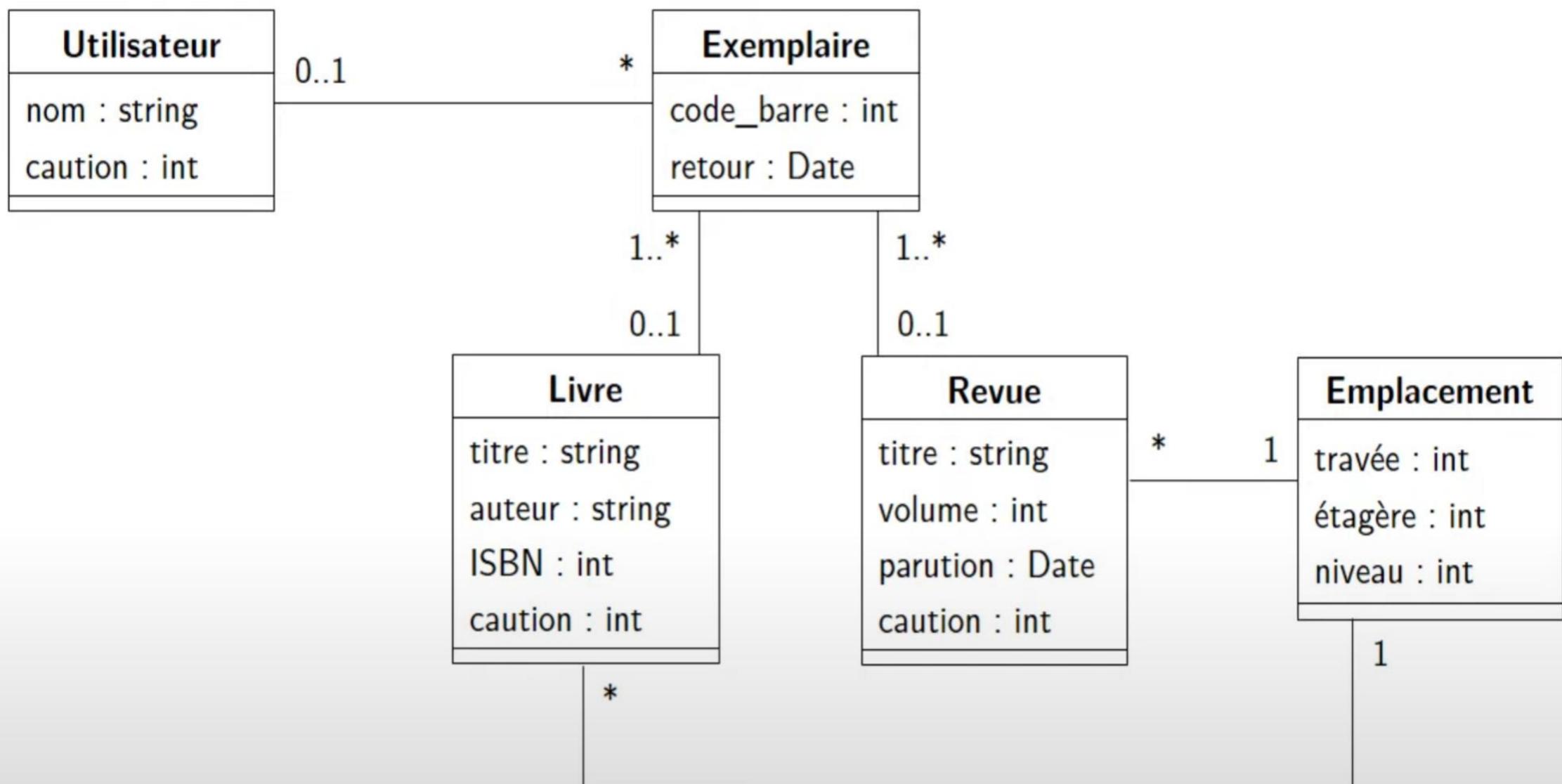
Livre
titre : string
auteur : string
ISBN : int
caution : int

Revue
titre : string
volume : int
parution : Date
caution : int

Emplacement
travée : int
étagère : int
niveau : int

Notes : Si un exemplaire n'est pas emprunté, retour à la valeur *null*

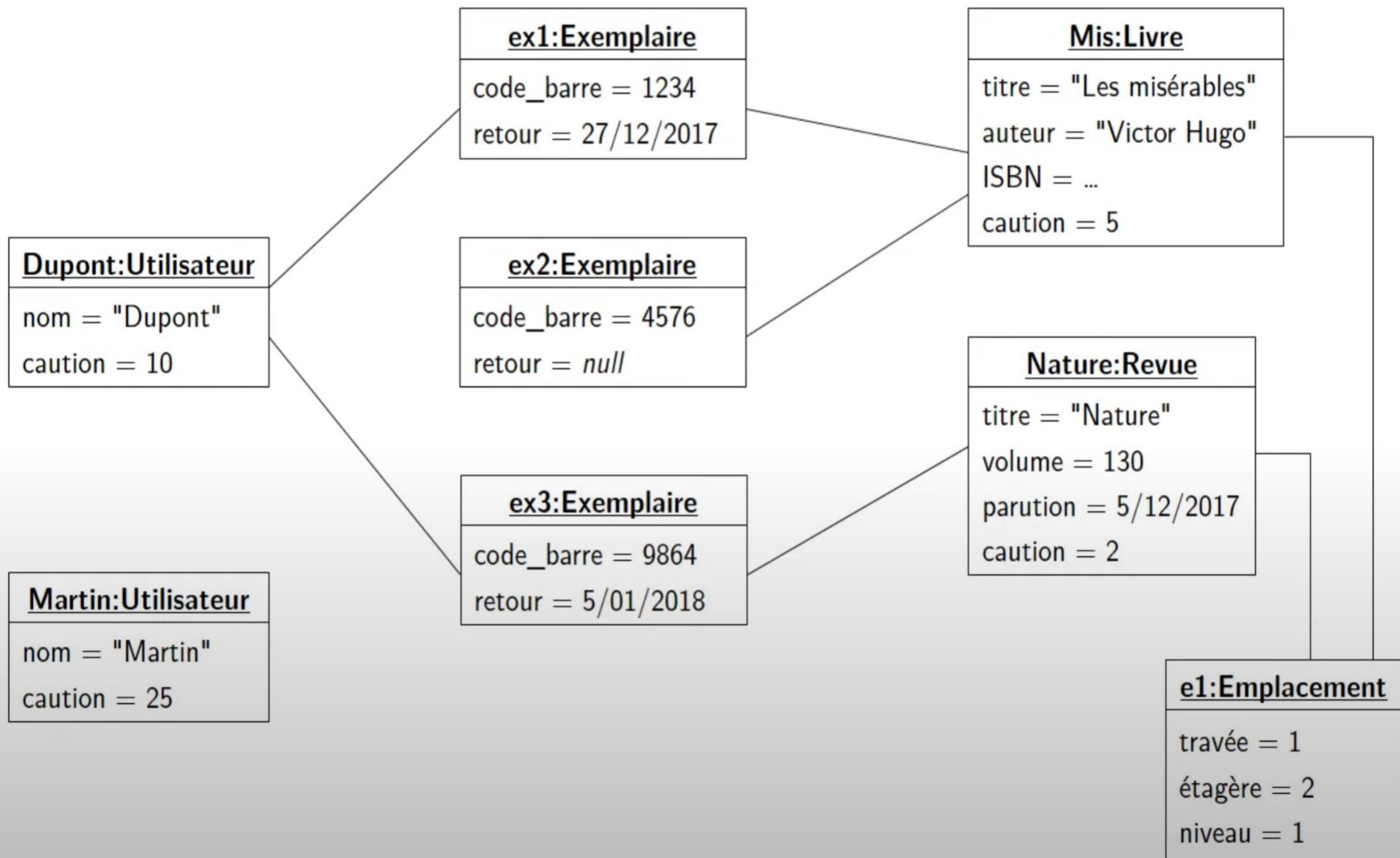
Exemple de la bibliothèque (2)



Notes : Si un exemplaire n'est pas emprunté, retour à la valeur *null*
Un exemplaire est un exemplaire d'un livre ou d'une revue

Exemple de la bibliothèque (2)

Exemple de diagramme d'objets



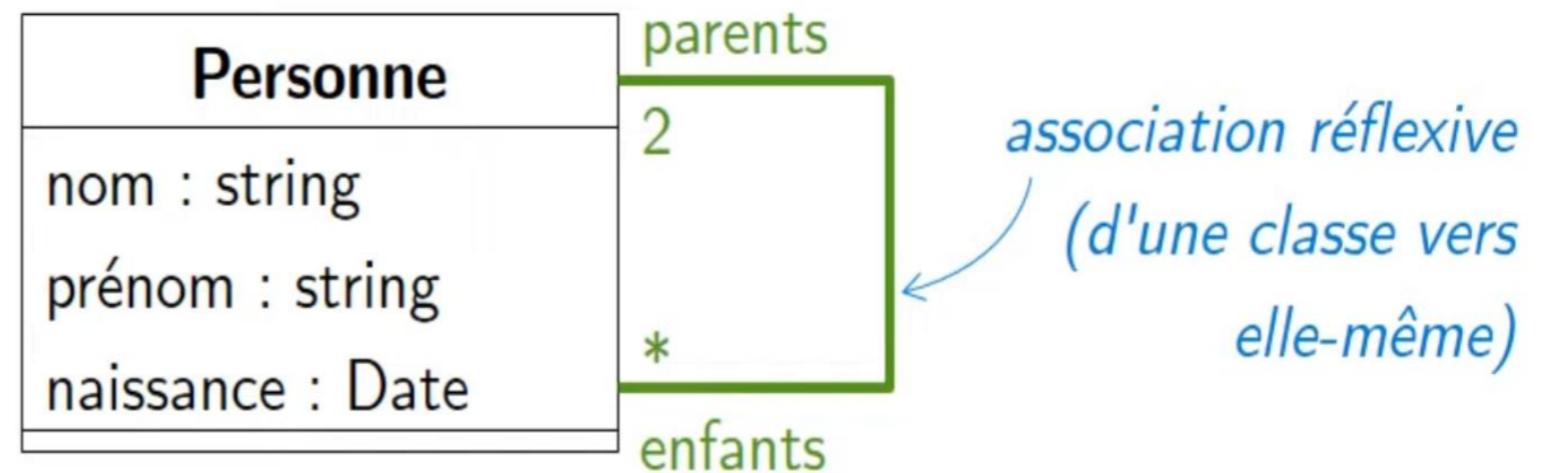
UML

Diagrammes de classes

2. Associations particulières, héritage

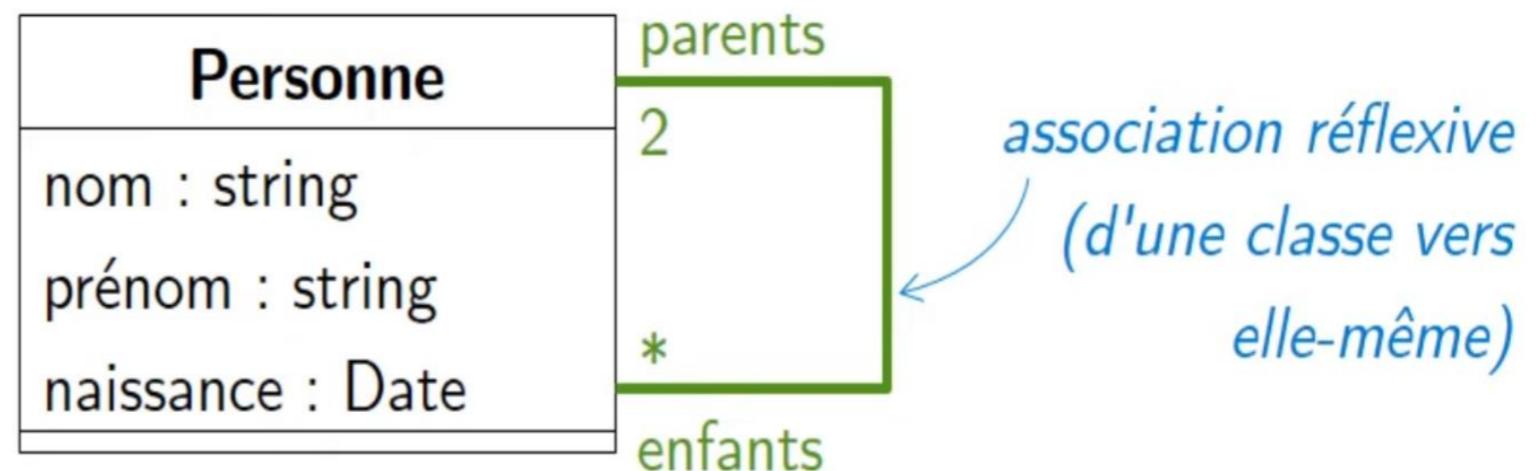
Association réflexive

Diagramme de classes

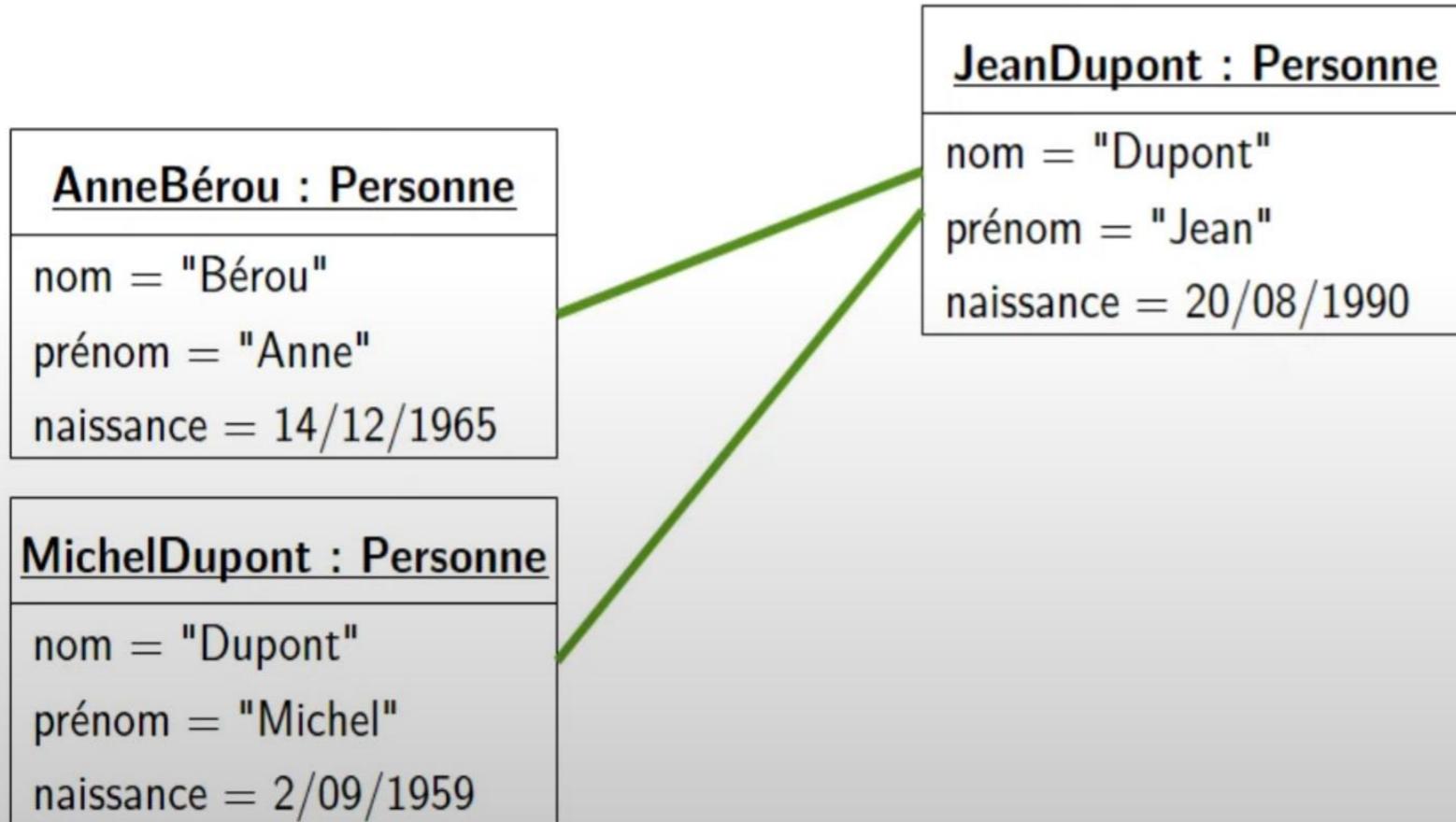


Association réflexive

Diagramme de classes

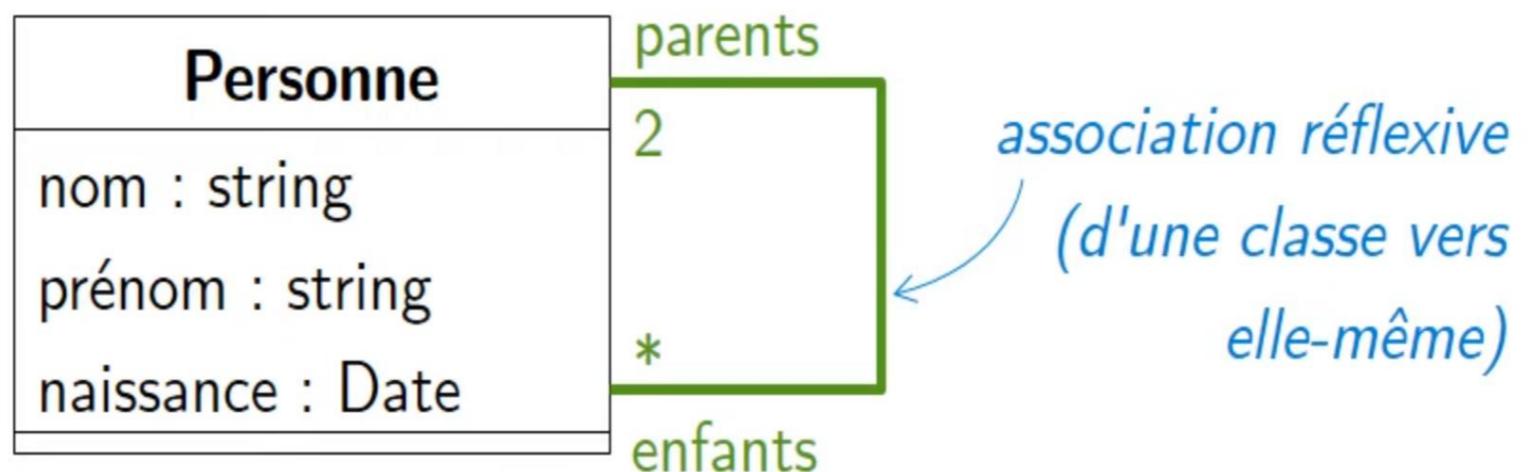


Exemple de diagramme d'objets

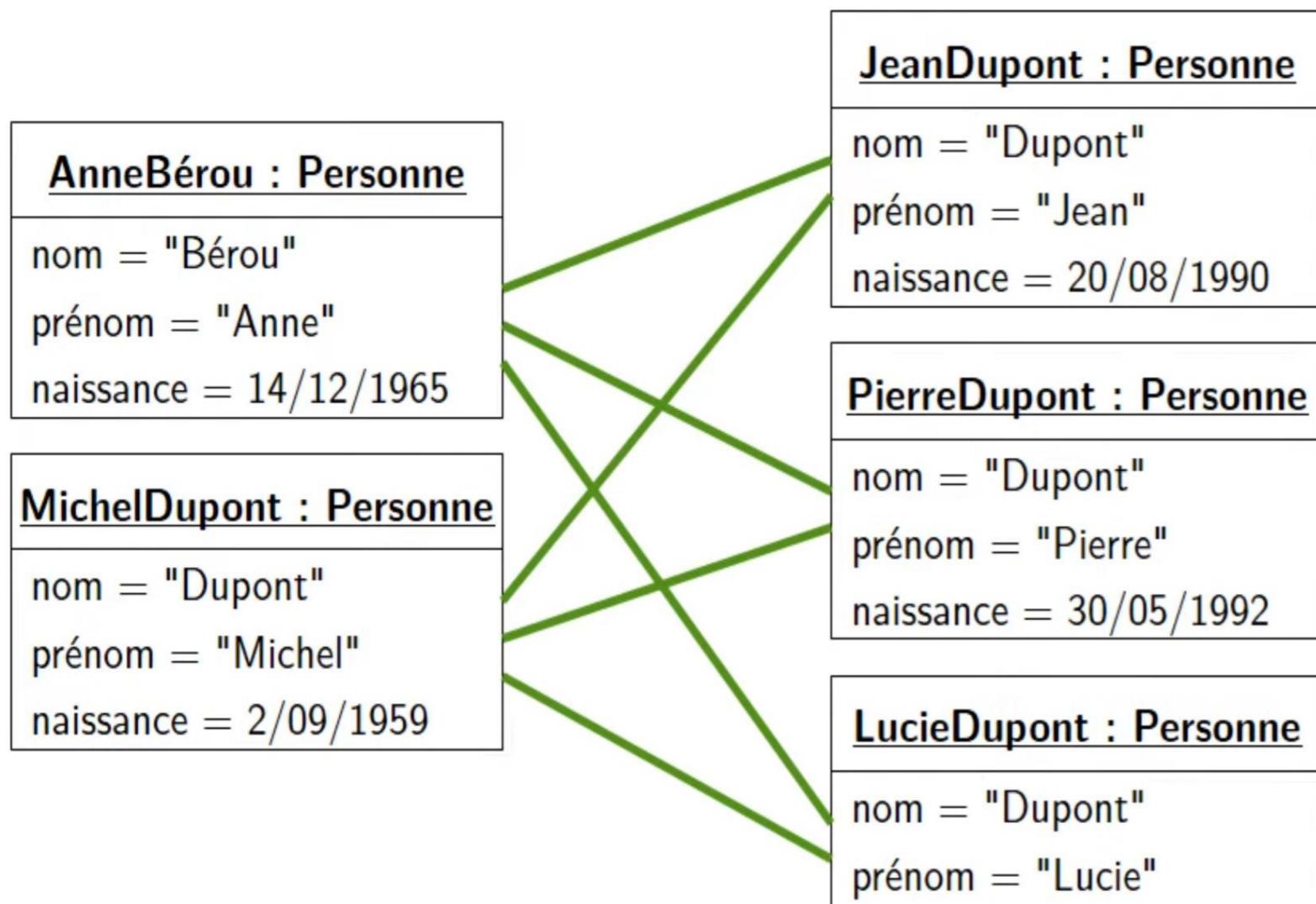


Association réflexive

Diagramme de classes

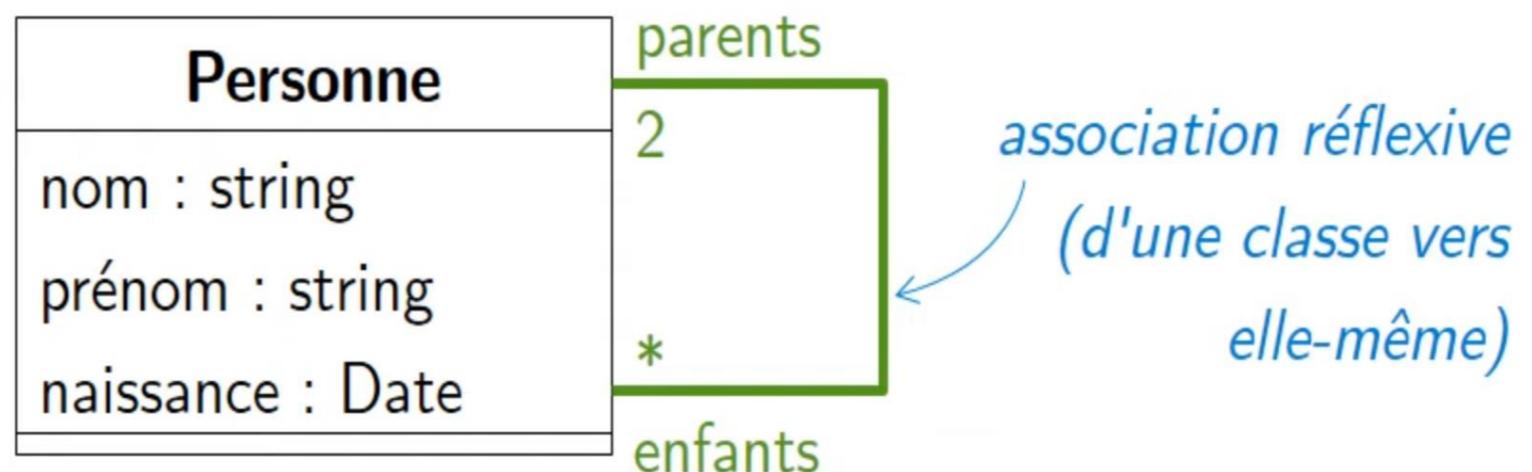


Exemple de diagramme d'objets

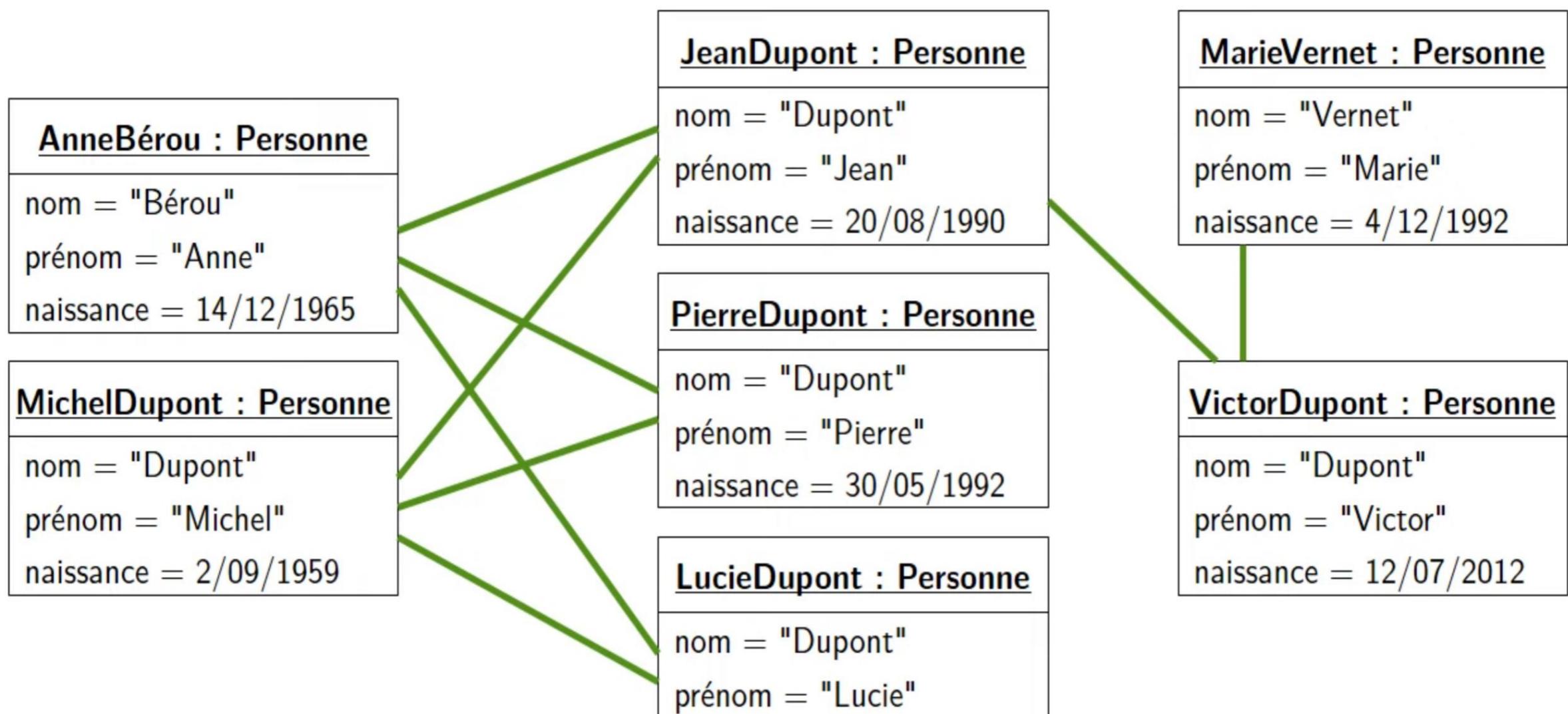


Association réflexive

Diagramme de classes

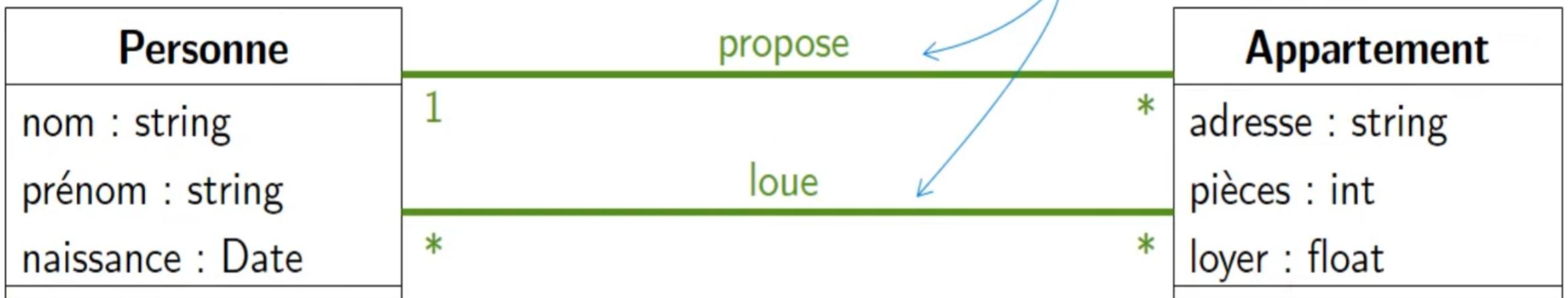


Exemple de diagramme d'objets



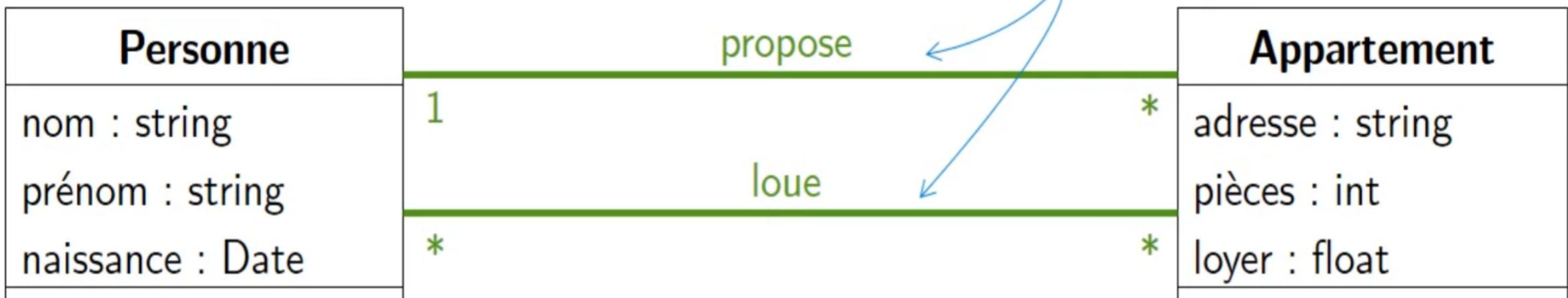
Associations multiples

Diagramme de classes

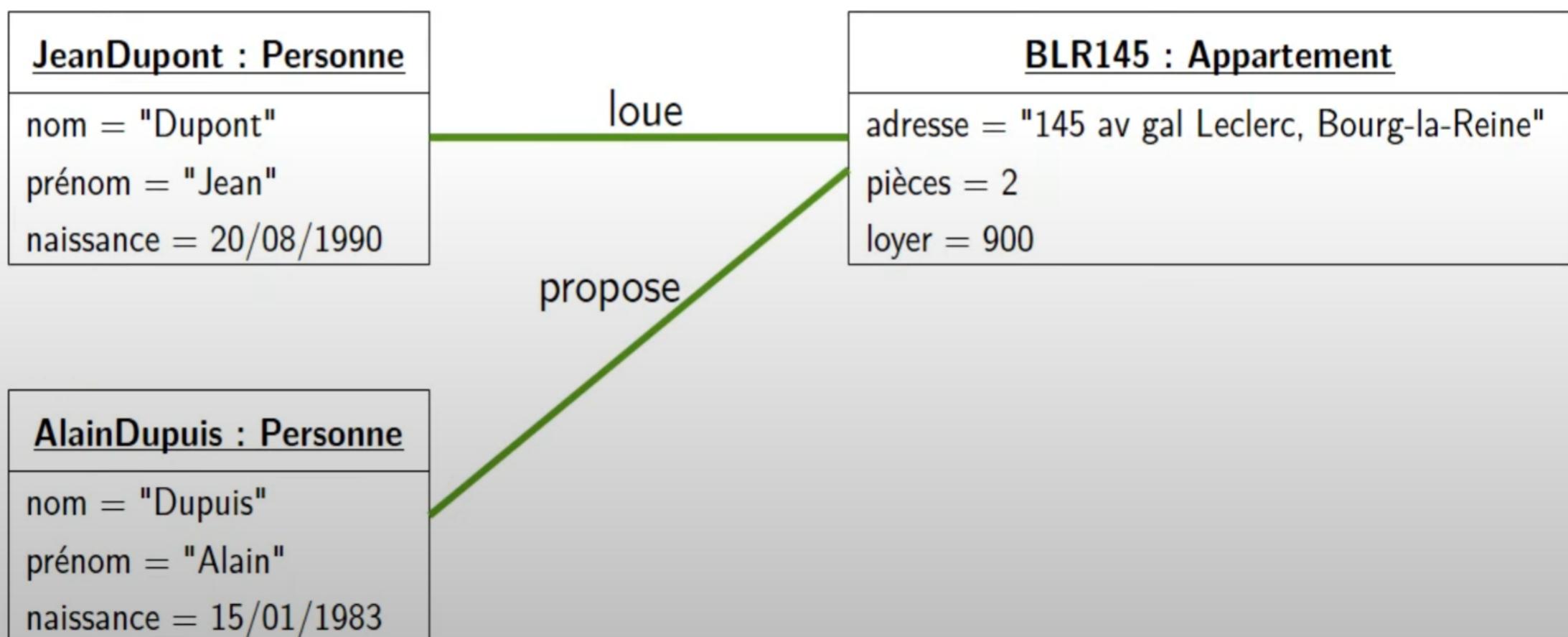


Associations multiples

Diagramme de classes

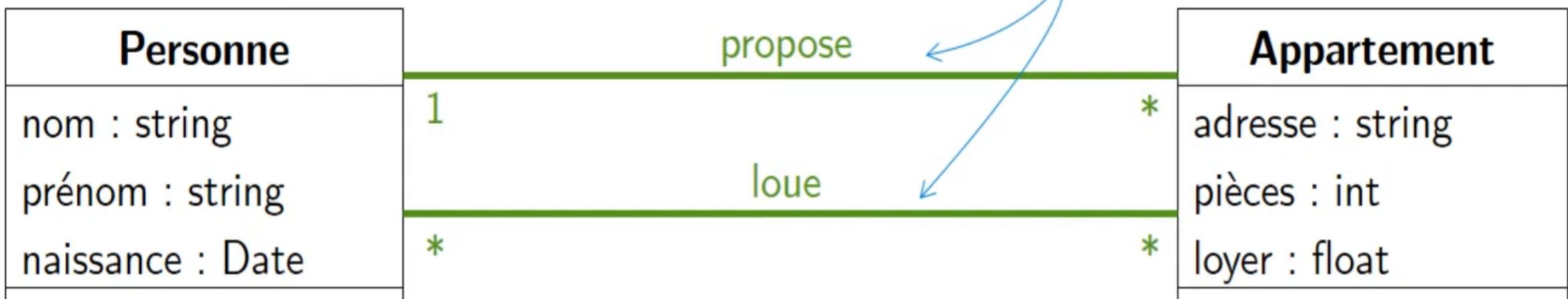


Exemple de diagramme d'objets

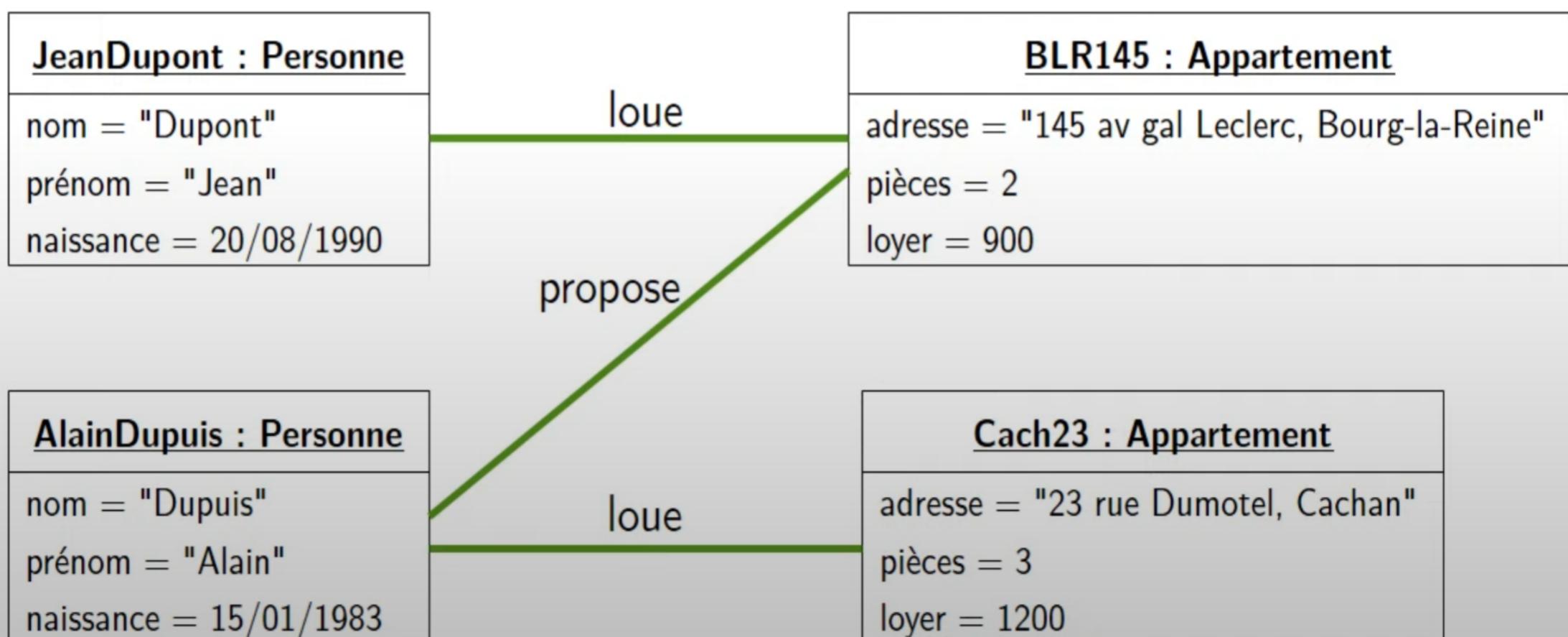


Associations multiples

Diagramme de classes



Exemple de diagramme d'objets



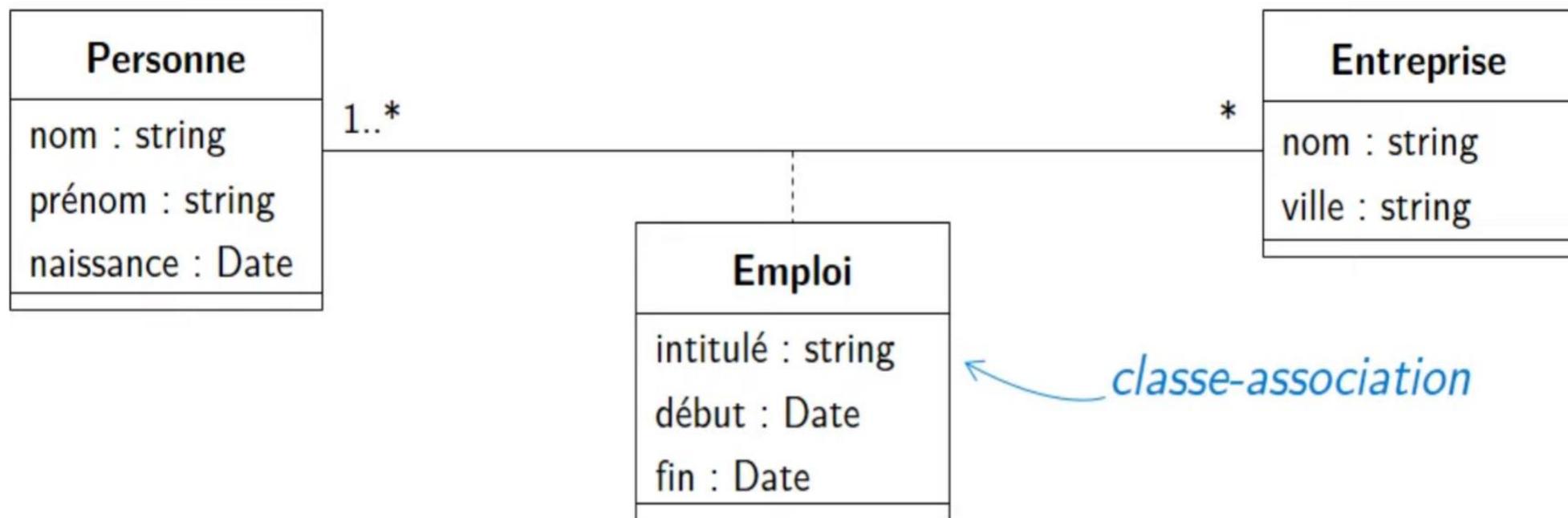
Classe-association

Permet de paramétrer une association entre deux classes par une classe



Classe-association

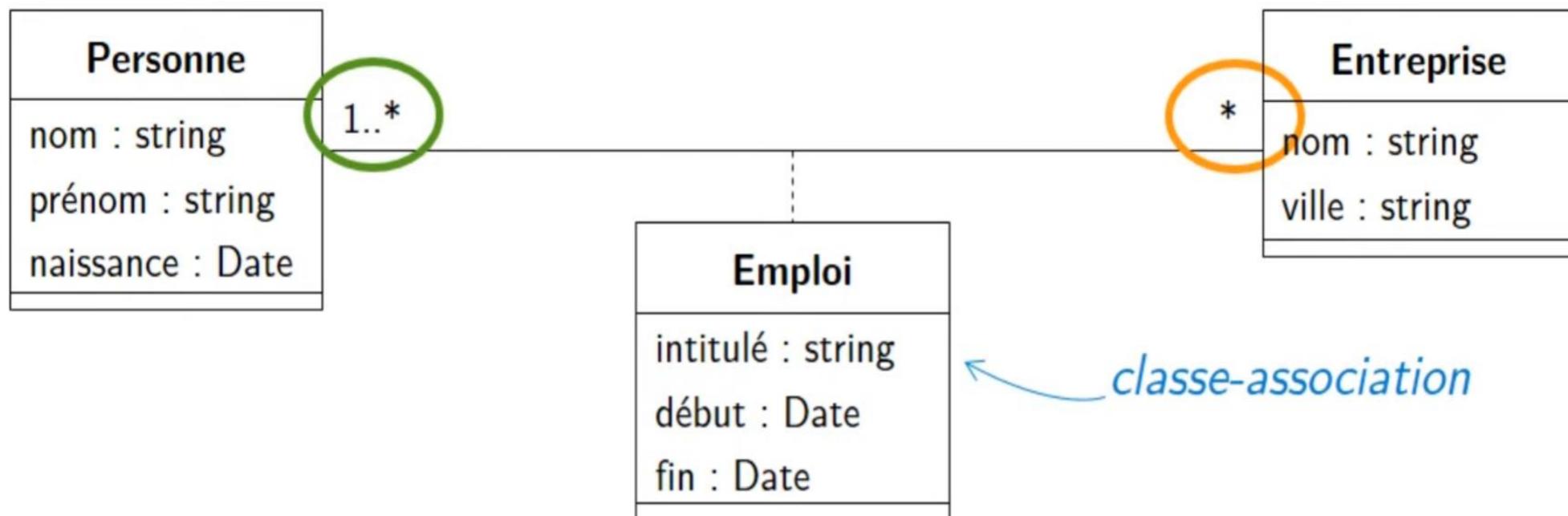
Permet de paramétrer une association entre deux classes par une classe



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

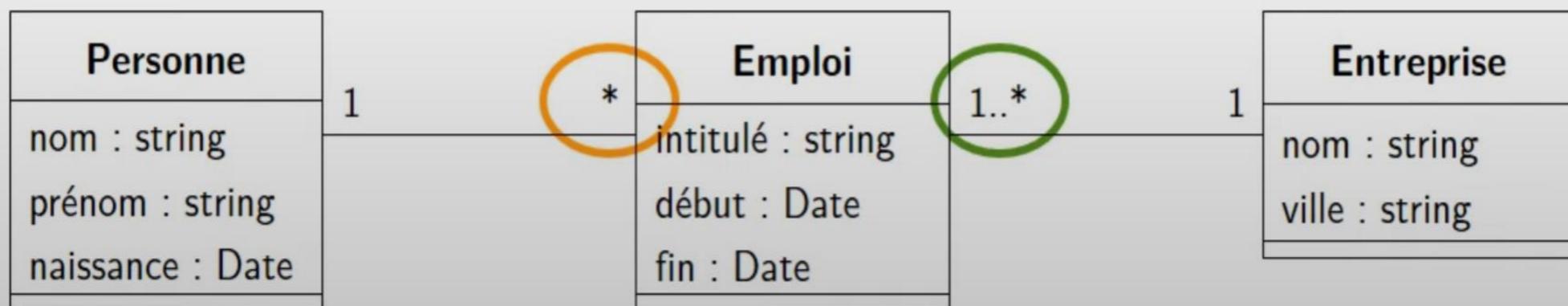
Classe-association

Permet de paramétrer une association entre deux classes par une classe



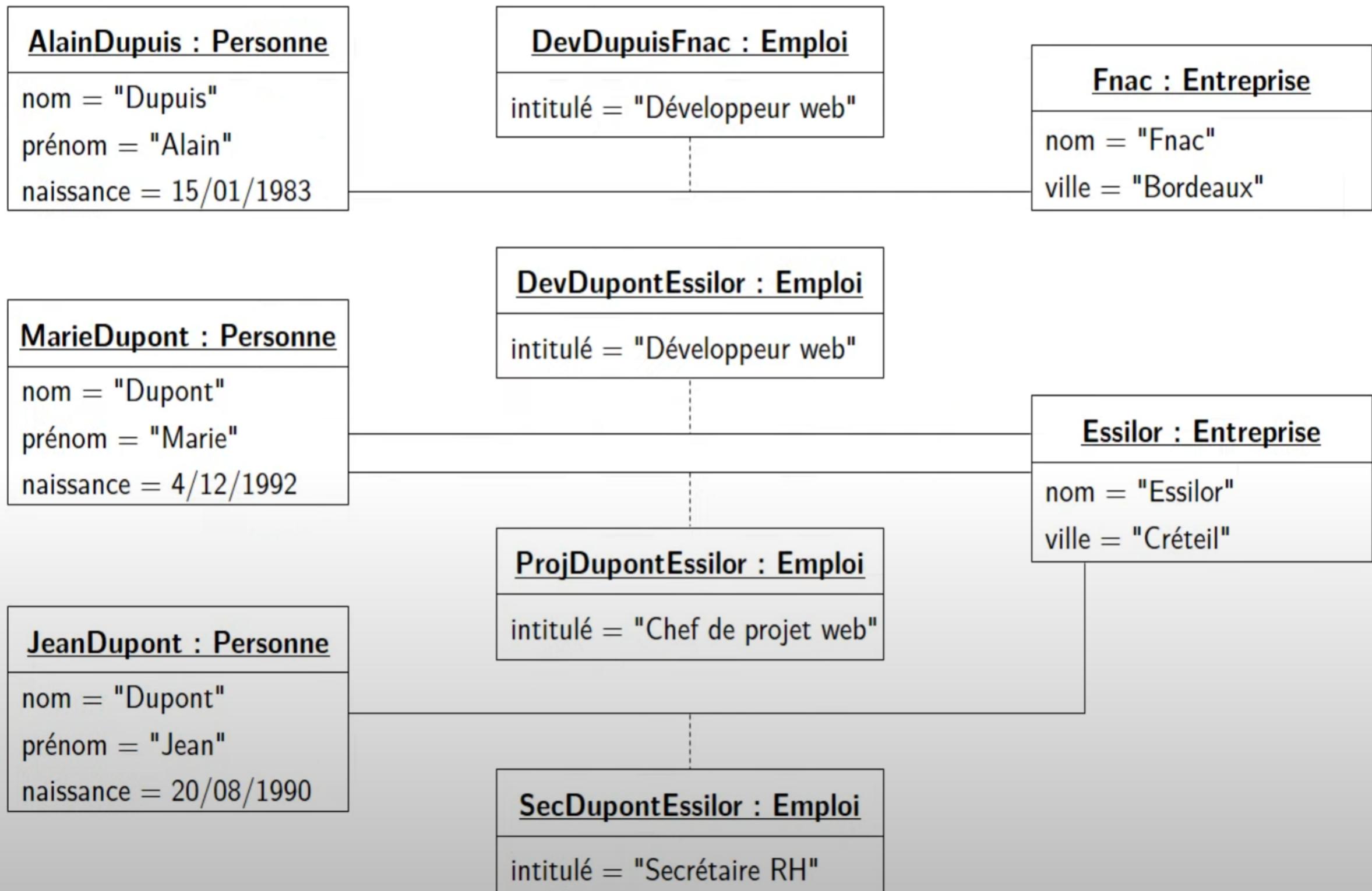
Instance unique de la classe-association pour chaque lien entre objets

Équivalence en termes de classes et d'associations :



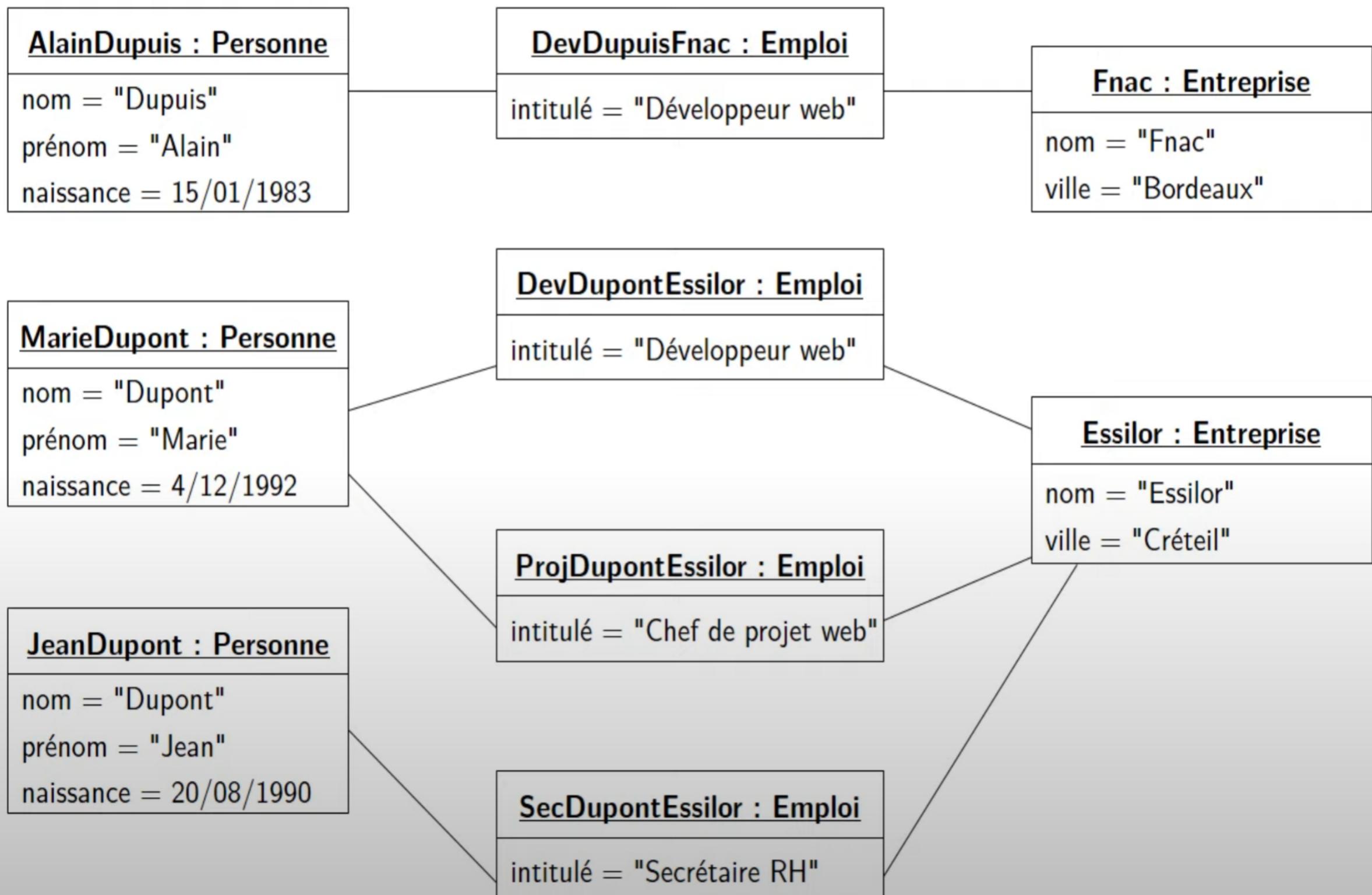
Classe-association

Exemple de diagramme d'objets

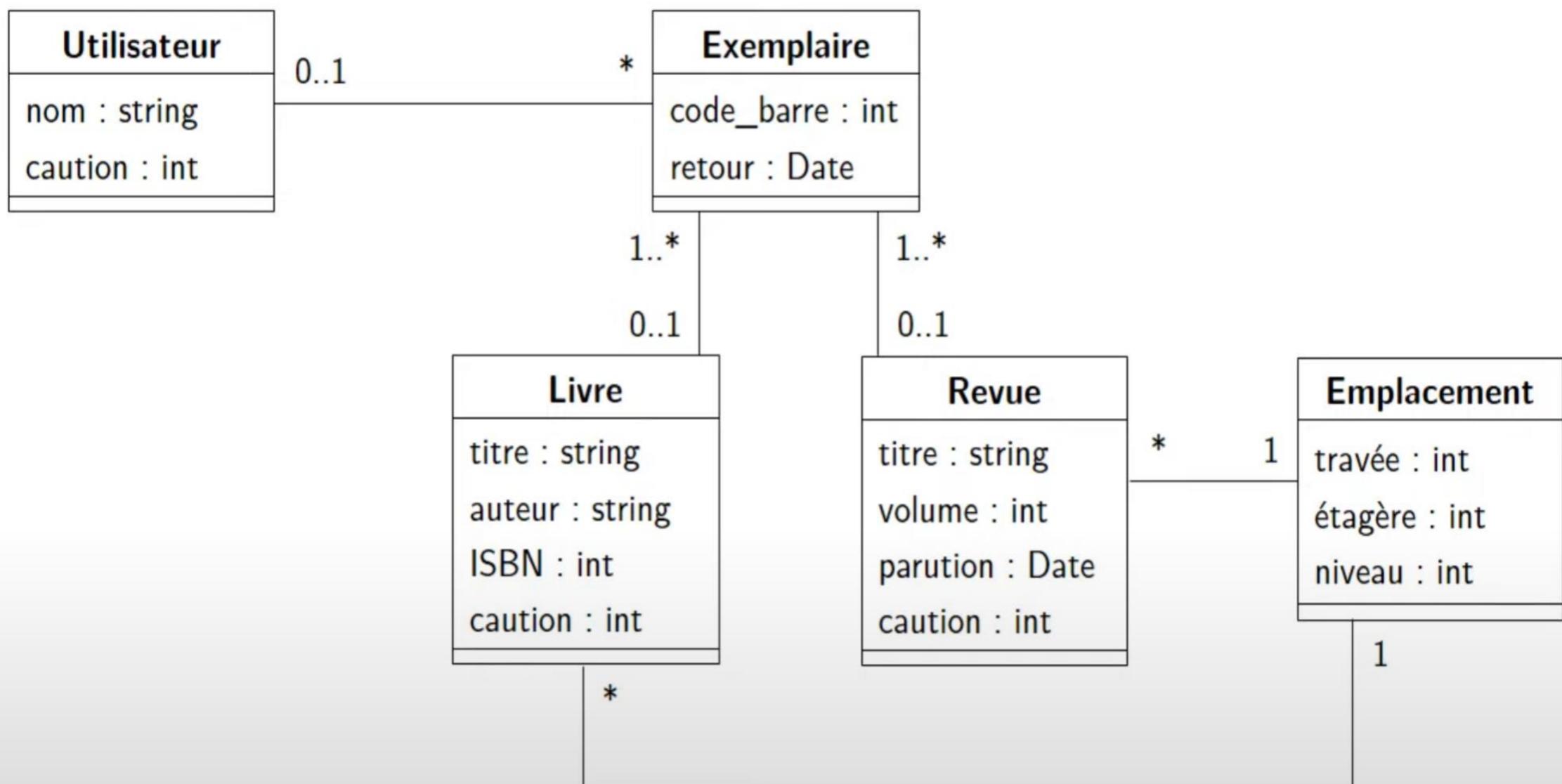


Classe-association

Diagramme d'objets du diagramme de classes équivalent

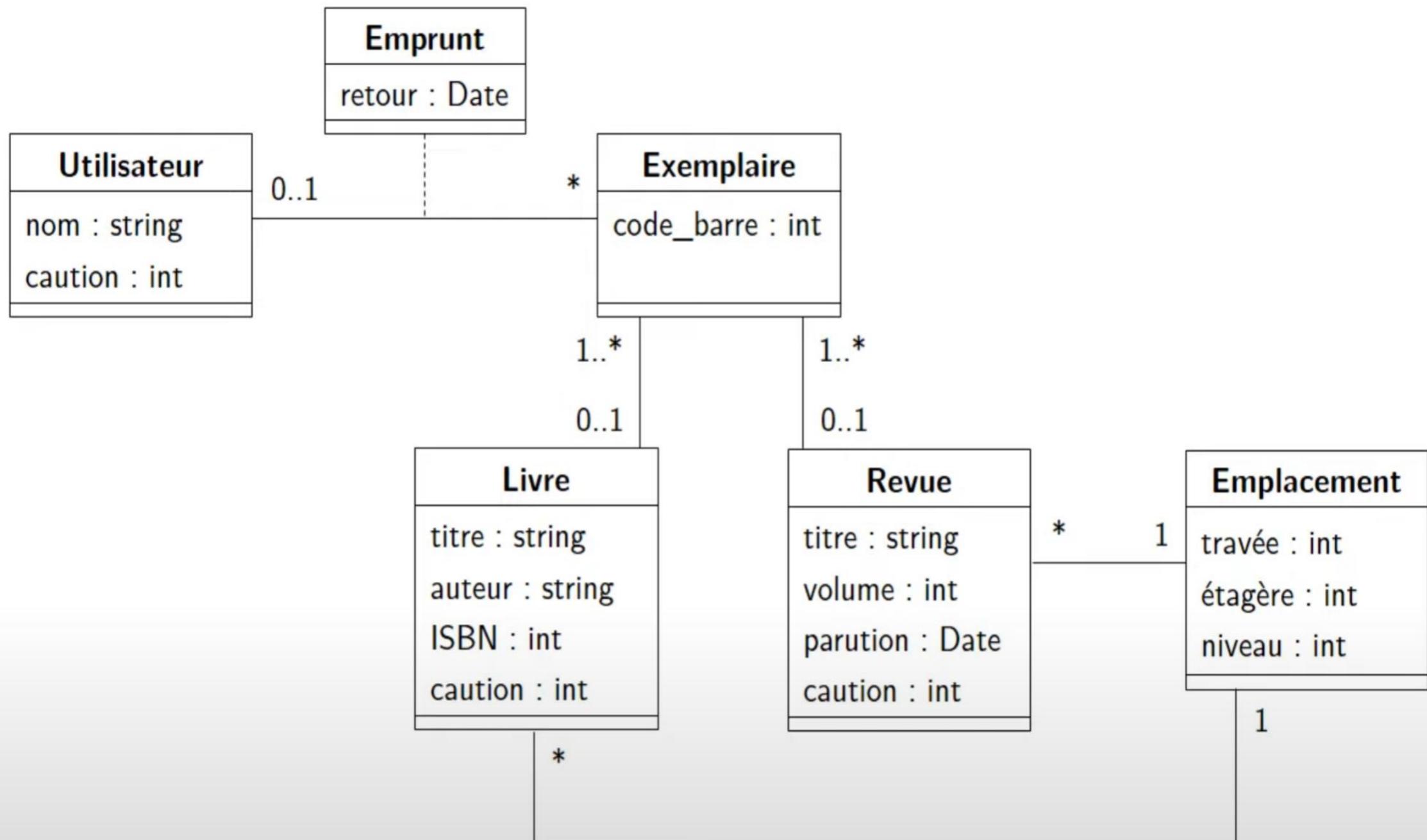


Exemple de la bibliothèque (3)



Note : Un exemplaire est un exemplaire d'un livre ou d'une revue
Si un exemplaire n'est pas emprunté, retour à la valeur *null*

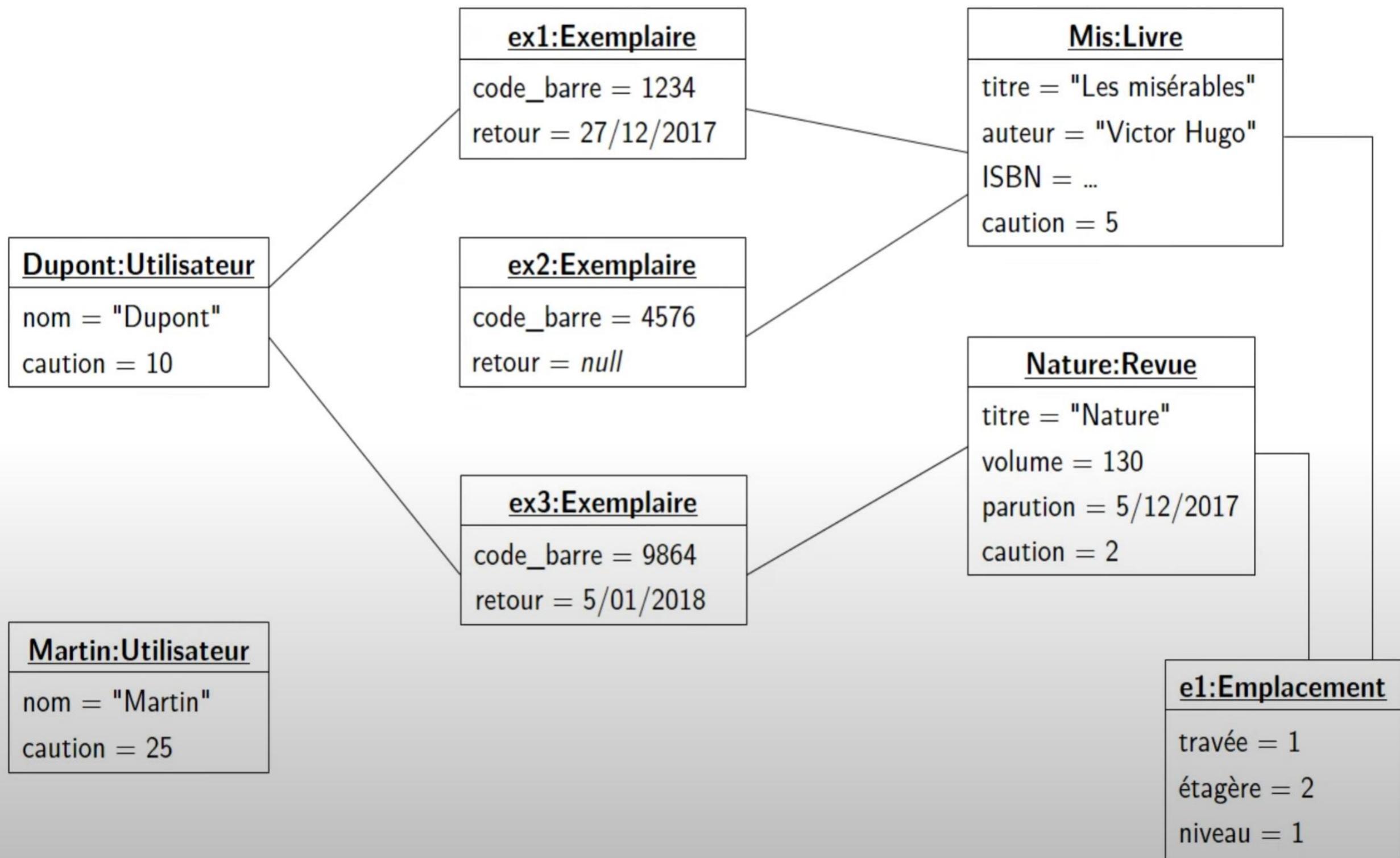
Exemple de la bibliothèque (3)



Note : Un exemplaire est un exemplaire d'un livre ou d'une revue

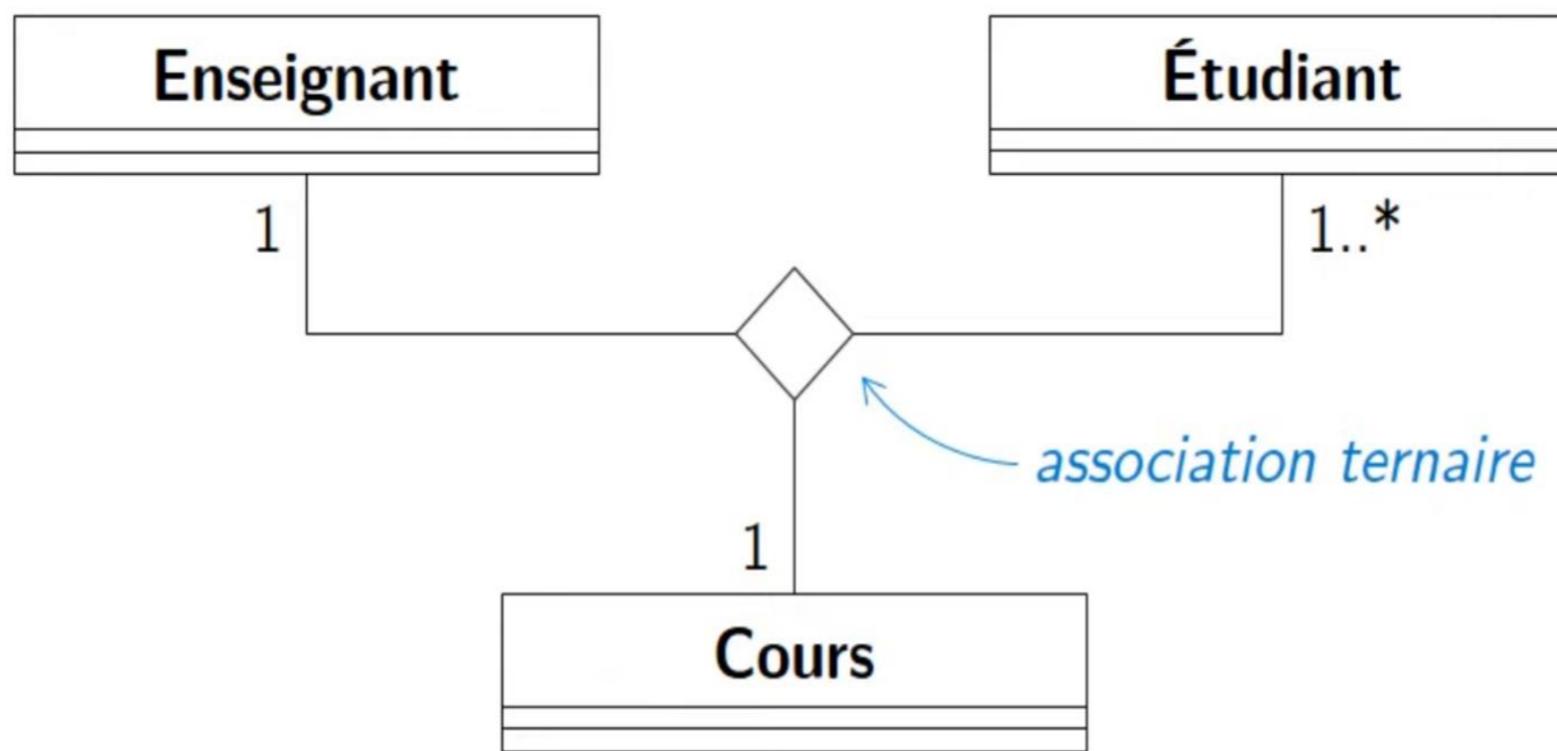
Exemple de la bibliothèque (3)

Exemple de diagramme d'objets

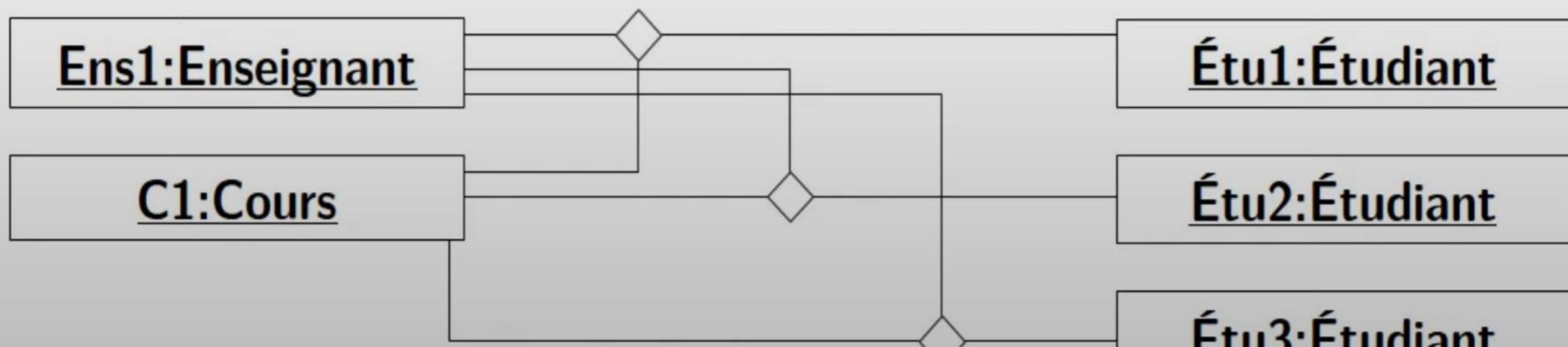


Association *n*-aire

Association reliant plus de deux classes

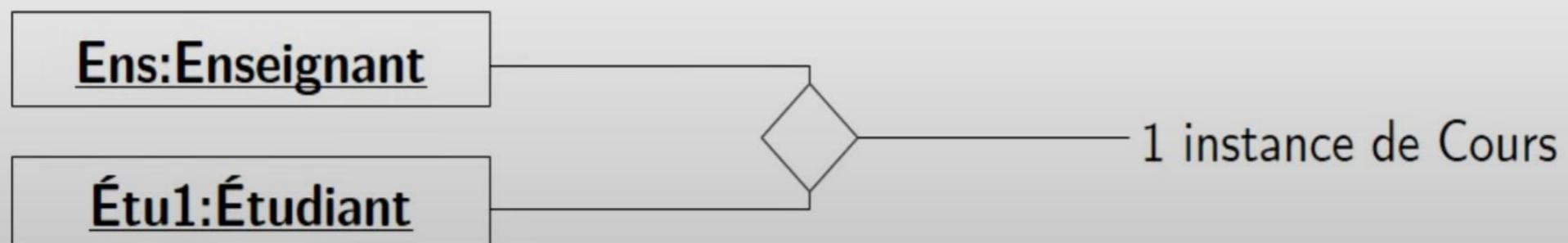
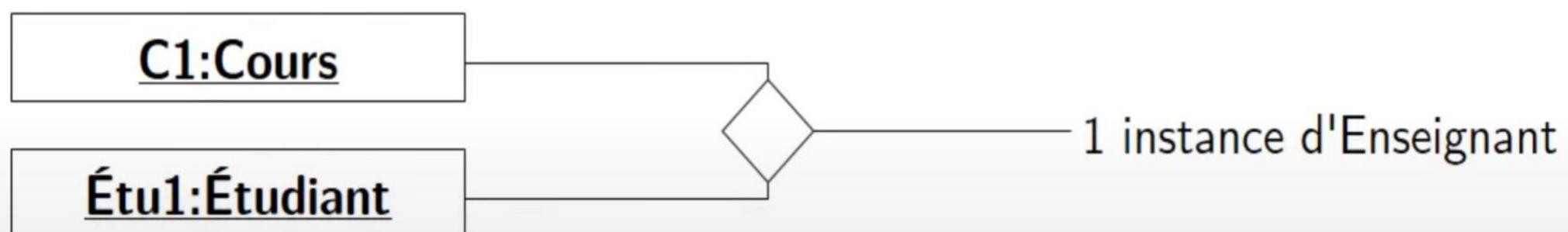
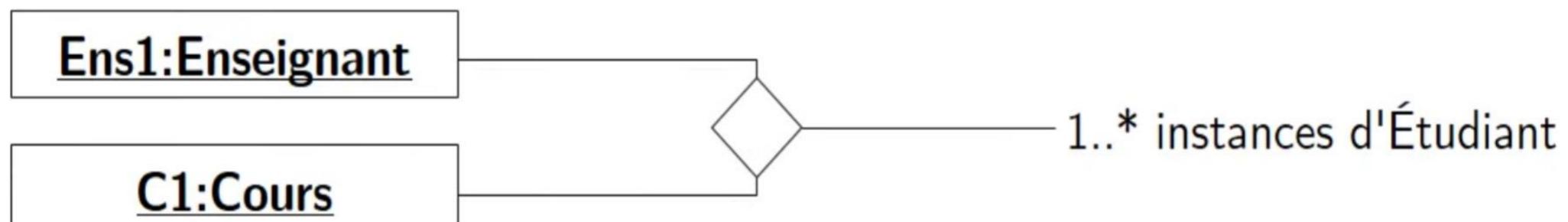


Instance d'une association *n*-aire = lien entre *n* objets



Association *n*-aire

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



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

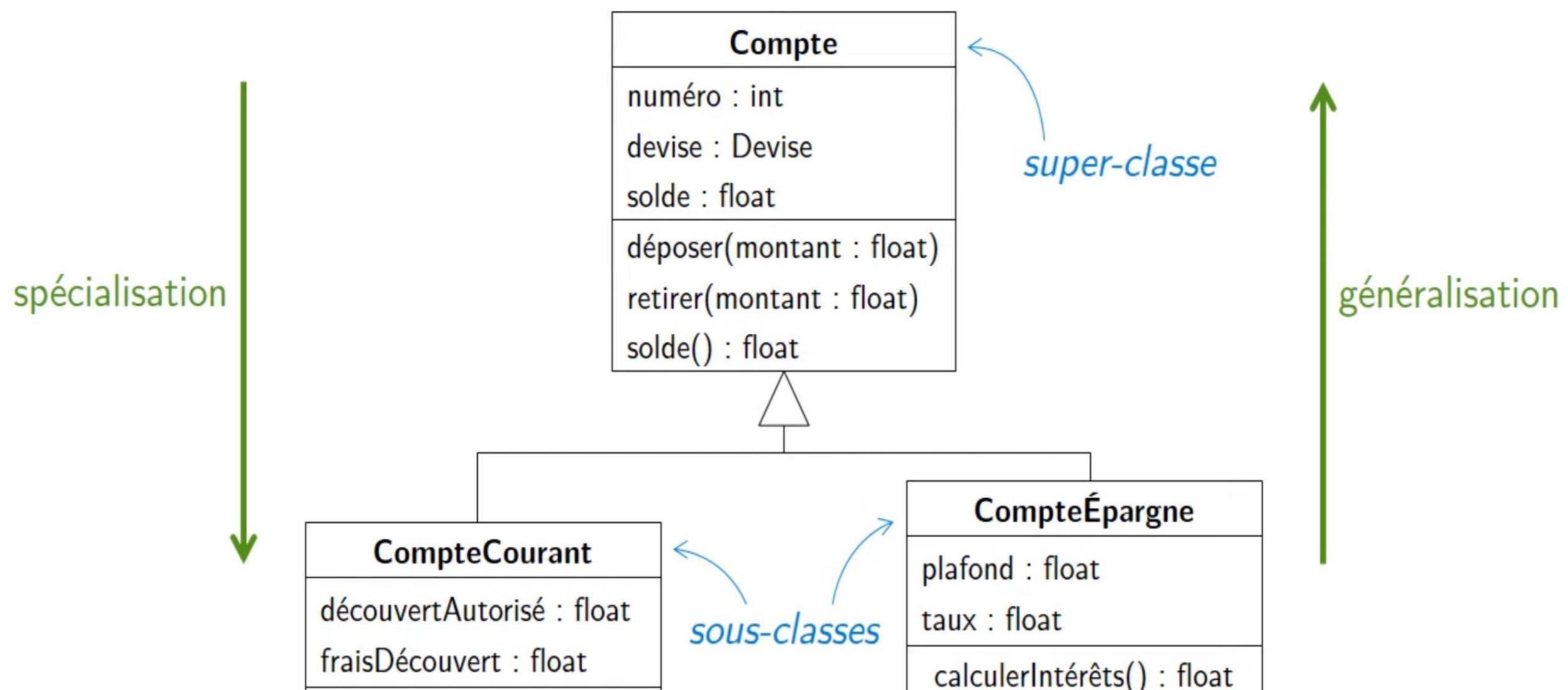
CompteÉpargne
numéro : int
devise : Devise
solde : float
plafond : float
taux : float
déposer (montant : float)
retirer(montant : float)
solde() : float
calculerIntérêts() : float

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

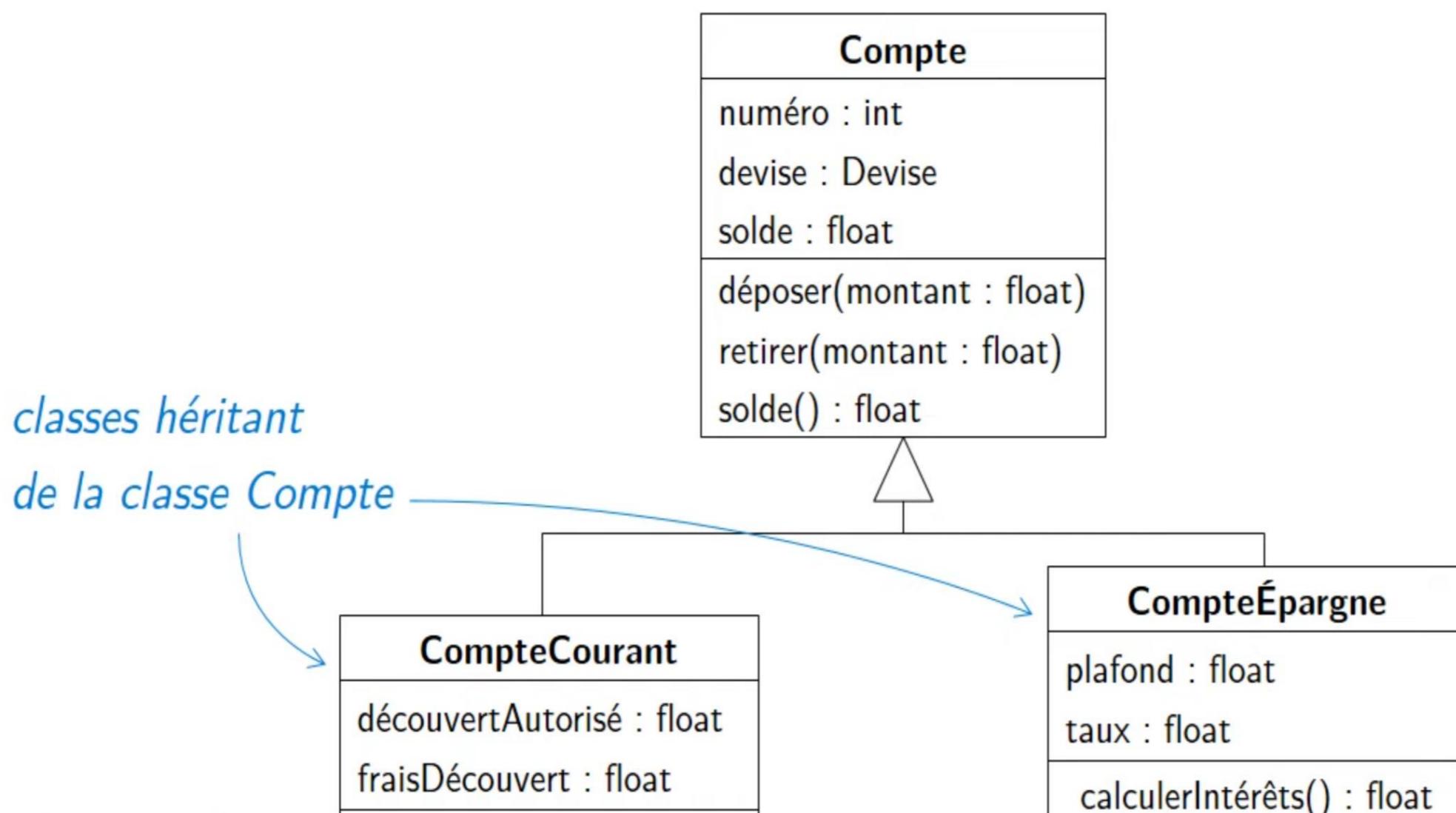
Généralisation : abstraction d'un ensemble de classes en super-classe



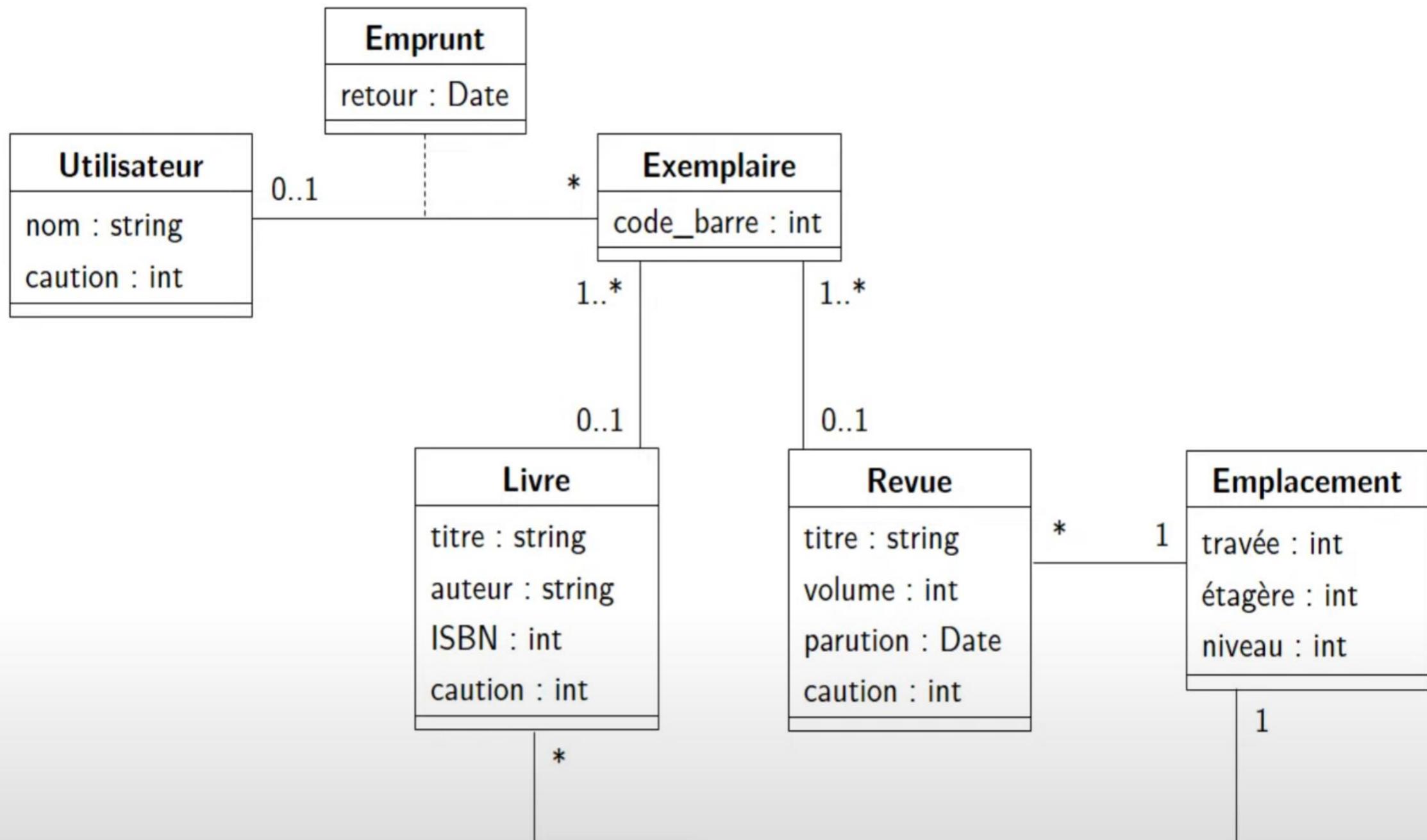
Hiérarchie de classes

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...)

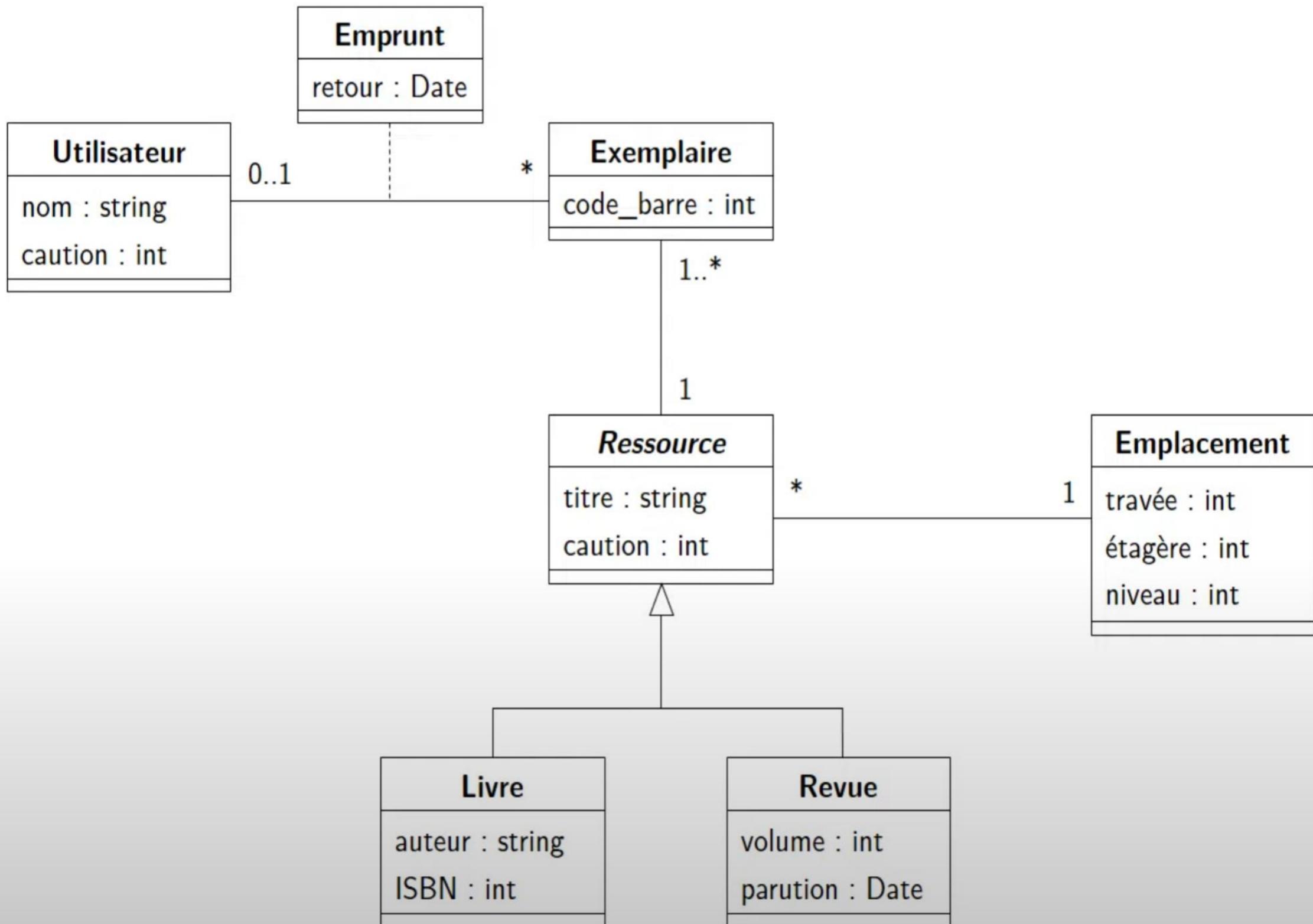


Exemple de la bibliothèque (4)



Note : Un exemplaire est un exemplaire d'un livre ou d'une revue

Exemple de la bibliothèque (4)



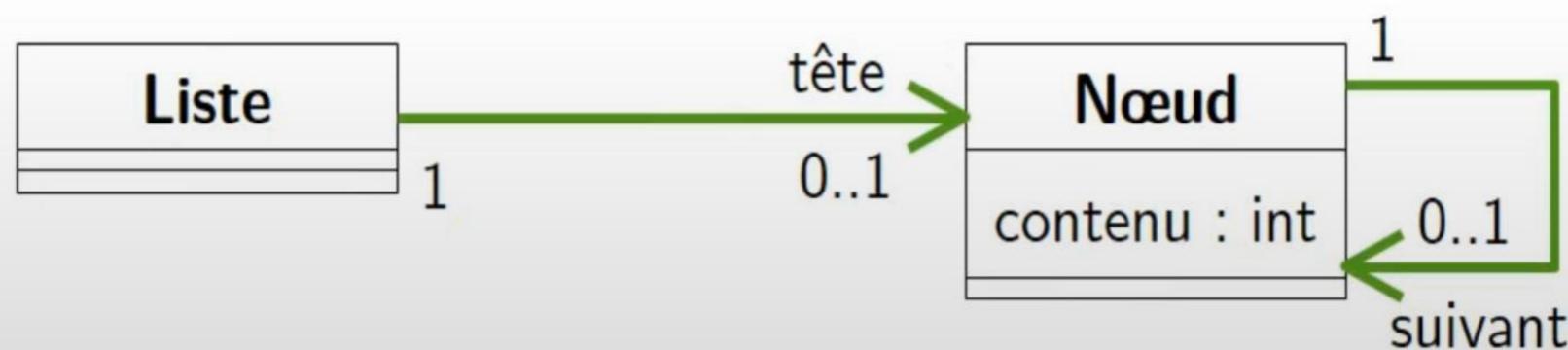
Navigabilité

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)

Agrégation

Association particulière entre classes

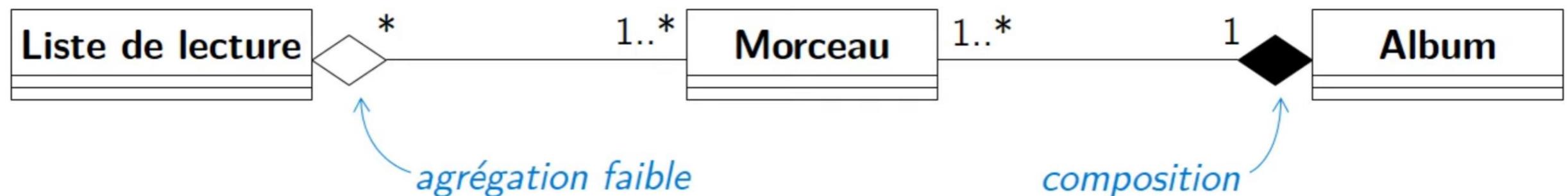
- Dissymétrique : une classe prédominante sur l'autre
- Relation de type composant-composite

Deux types d'agrégation

- Agrégation faible
- Composition

Exemple

Lecteur de contenu audio permettant de créer des listes de lecture



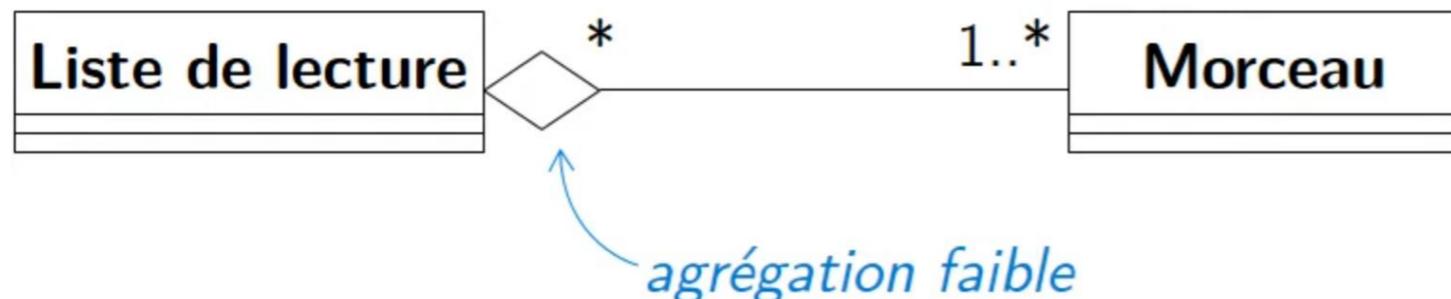
Agrégation faible

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



Exemple de la bibliothèque (5)

