

# Analyse et Conception avec UML

## Les diagrammes de classes

Université Nice Sophia Antipolis, IUT

Site web du module :  
<https://mbf-iut.i3s.unice.fr/>

[blay@unice.fr](mailto:blay@unice.fr)  
<http://mireilleblayfornarino.i3s.unice.fr/>

# Bibliographie

- Voir sur le site web les autres cours.
- UML Par la pratique (surtout dans sa dernière édition) (présent à la bibliothèque de l'IUT)
- Méthodologie en Ingénierie du logiciel, Modélisation Orientée objet, M.Grimaldi – janvier 2010



- <https://goo.gl/RrBtfD>

**Pour dans 15 jours**

## PROGRAMMATION OBJET - 1

Pr. Xavier Blanc  
Université de Bordeaux

0:45 / 34:29



### PROGRAMMATION D'OBJETS = LA CLASSE

- Une **classe** décrit un **ensemble d'objets** qui ont
  - Les mêmes types de données (nom, type, valeur de base)
  - Les même type de traitements (nom, paramètres)



<https://goo.gl/juA1Yd>



# Analyse du domaine

# Détermination des concepts du domaine

La détermination des concepts s'effectue sur la base des cas d'utilisation par simple **analyse grammaticale** de la description textuelle.

D'une manière générale,  
les **noms** représentent des **concepts** ou des **attributs**  
tandis  
que les **verbes** représentent des **comportements**  
(opérations, méthodes)



# Identifier les concepts



**Règle du cartographe** : Le modèle du domaine se construit de la même façon qu'un cartographe dessine une carte :

- ✓ En utilisant le **vocabulaire** du domaine étudié.
- ✓ En excluant les éléments **non pertinents**.
- ✓ En n'incluant pas d'éléments **inexistants** dans le domaine.

# Identifier les relations entre les concepts



- Déterminer les relations entre les concepts est essentiel.
- Choix entre concept et attribut : Si un élément du domaine étudié est autre chose qu'un nombre ou un simple texte, alors il s'agit probablement d'un **concept** et non d'un **attribut**.

# Concepts du domaine

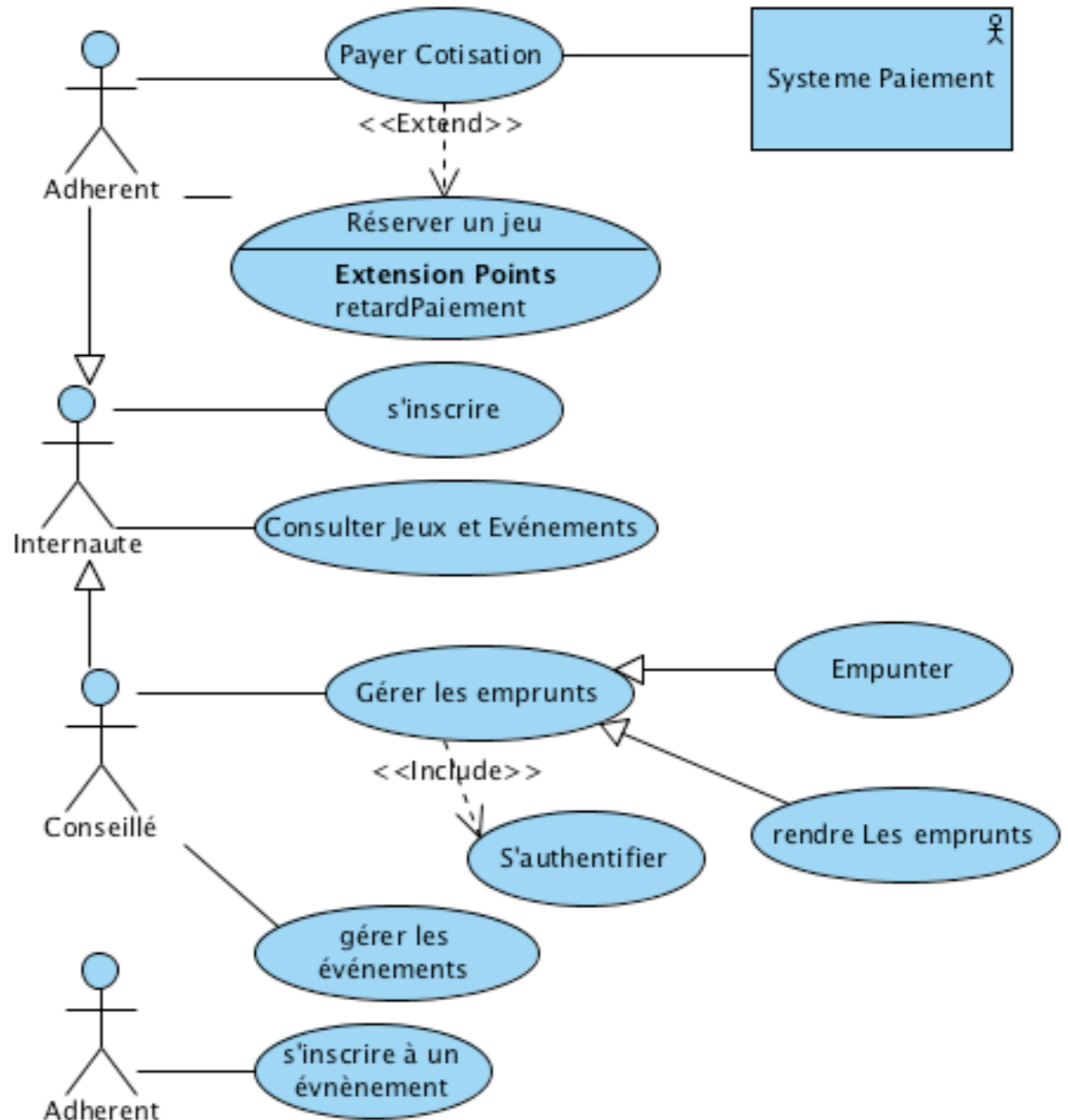
**Par l'exemple, découverte  
de la notation**



# Concepts de la ludothèque ?

- (1) Nous voulons informatiser une ludothèque pour favoriser la consultation des jeux proposés par la ludothèque.
- (2) Les adhérents peuvent emprunter des jeux en s'adressant à un conseiller qui enregistre l'emprunt.
- (3) Les jeux empruntés sont rendus à un conseiller....
- (4) Un adhérent peut réserver des jeux. Une réservation précise l'emprunteur, le jeu et la date de la demande de réservation. L'adhérent est averti quand le jeu revient en rayon.
- (5) Pour organiser un événement le conseiller spécialisé doit alors donner les informations suivantes : les jeux à tester, le nombre maximal et minimal de participants attendus, la date, et l'heure de début de l'événement.
- (6) Un adhérent peut s'inscrire pour participer à un événement à condition qu'il y ait encore de la place.
- (7) Un adhérent peut payer sa cotisation en ligne par un système de paiement externe

# Ludothèque



# Description d'un flot associé à un cas d'utilisation

*Un conseiller enregistre l'emprunt d'un jeu pour un adhérent*

- 1) Le conseiller s'authentifie;
- 2) Le conseiller saisit **l'identifiant** du jeu et de l'adhérent
- 3) Le système vérifie la **disponibilité** du jeu
- 4) Le système vérifie que la cotisation est bien payée
- 5) Le système vérifie que l'**adhérent n'a pas de pénalité impayée**
- 6) Le système enregistre **l'emprunt**.
- 7) Le système signale que l'emprunt est valide.

**Vous avez retenu ?**

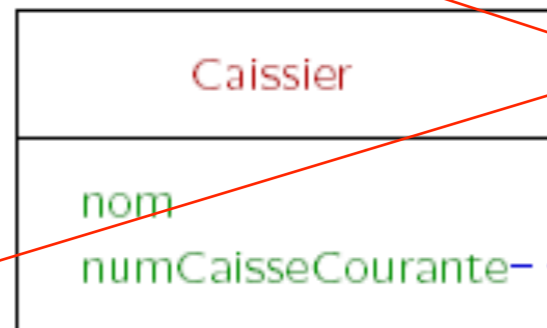


# Notes

- Un attribut ne peut représenter qu'une **valeur primitive** (entier, texte, date, identificateur, matricule, ...).
- Un attribut ne peut représenter que des données **relatives au concept auquel il est associé**.

Incorrect

Pas d'attributs complexes!!



pas un attribut simple!

Correct

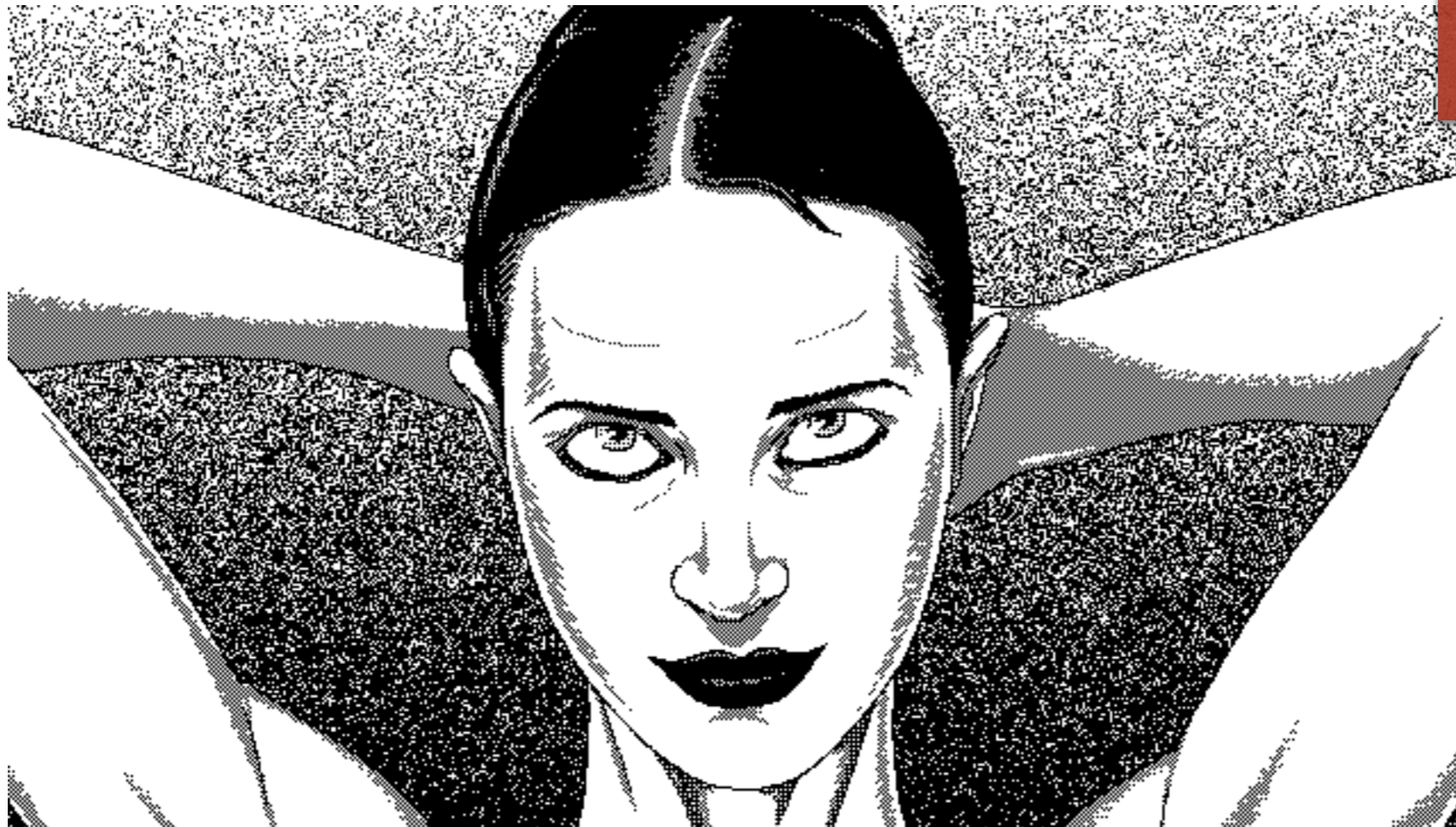


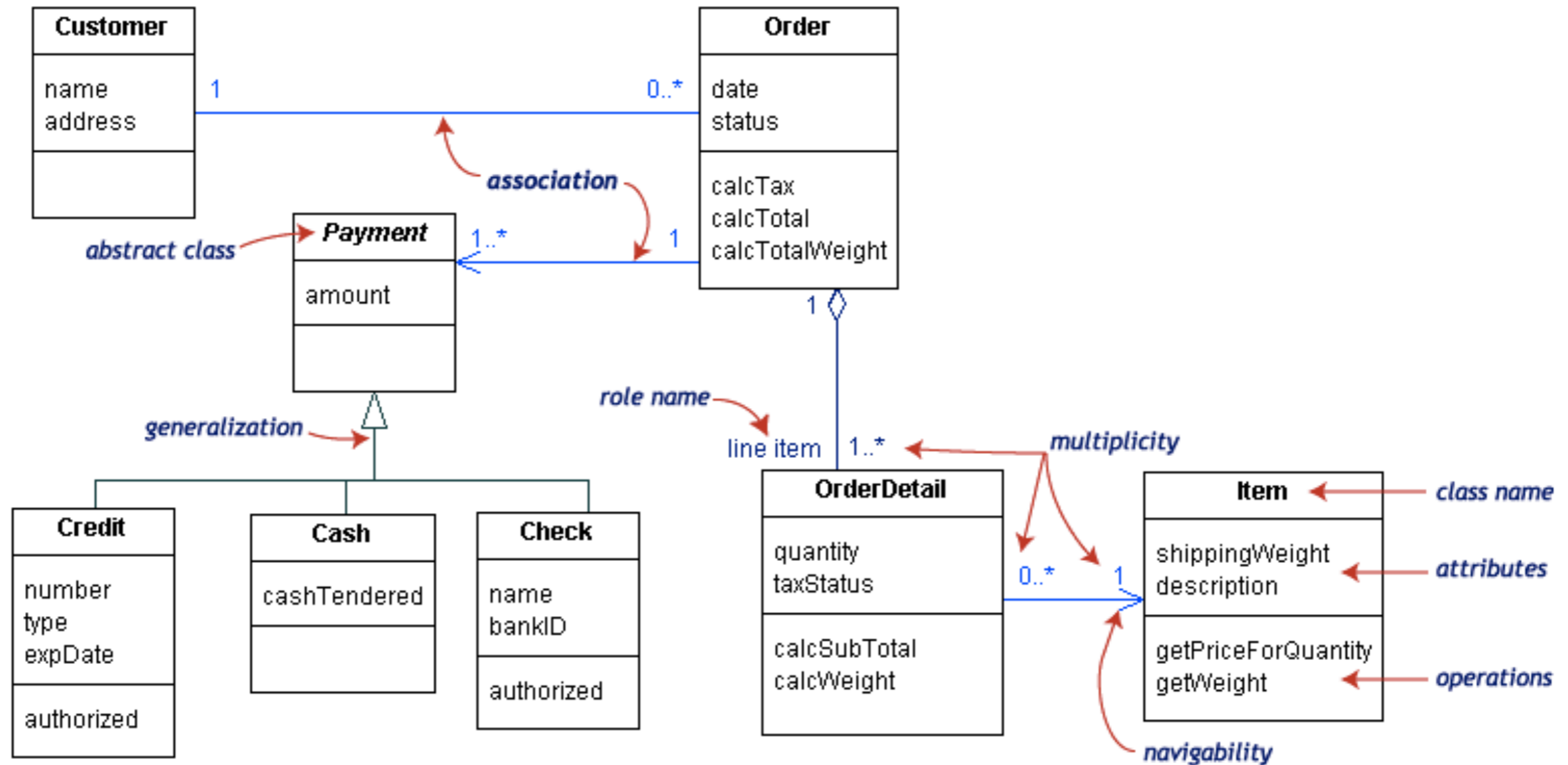


# Vous avez retenu ?

On représente les attributs complexes  
par des associations EXCLUSIVEMENT

Sinon



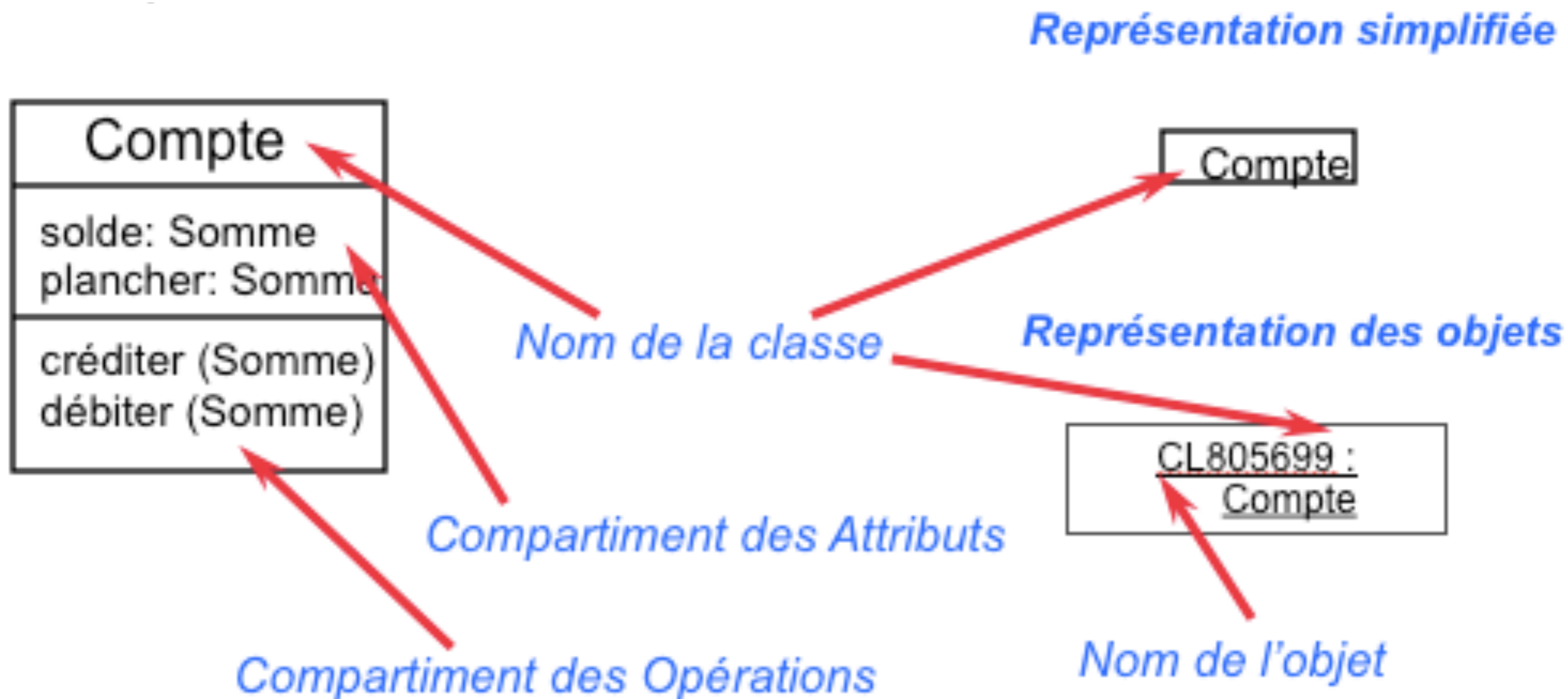


# Notation Synthèse

image : <http://edn.embarcadero.com/article/31863>

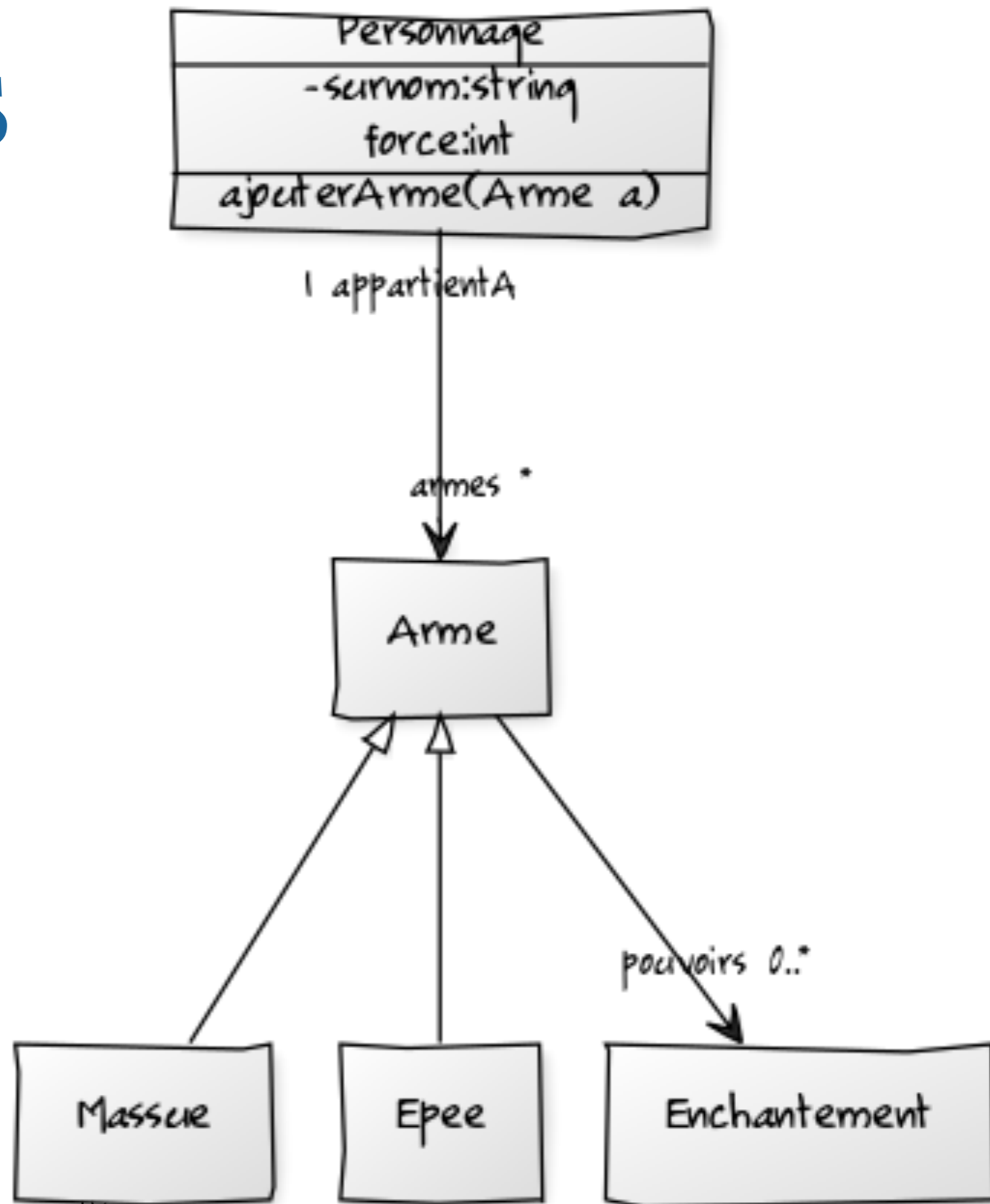


# Notations UML pour classes et objets



# Relations entre classes

Explicitation  
des notations  
déjà vues au  
tableau





# Associations

Les associations fournissent un chemin de communication entre objets - si deux objets ont besoin de se parler, il doit exister un lien entre eux

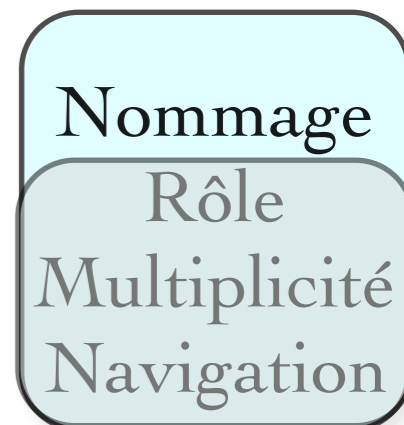
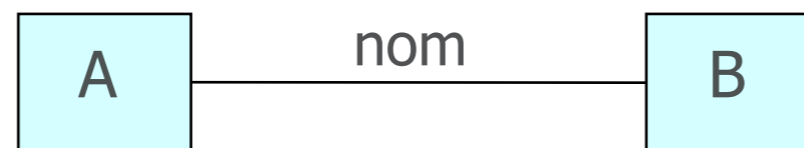
# Associations

Nommage  
Rôle  
Multiplicité  
Navigation

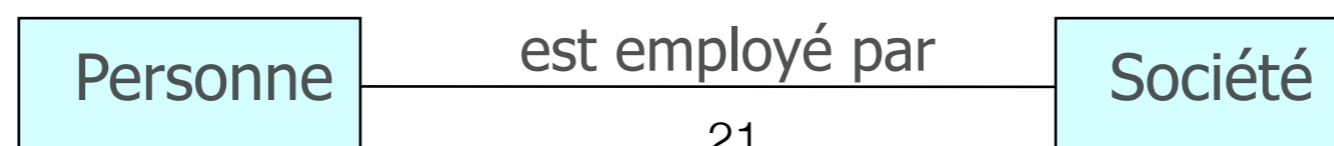
- Les associations peuvent avoir des étiquettes:
  - Il s'agit du **nom de l'association**.
- Les associations peuvent avoir des noms de **rôle**:
  - un nom de rôle identifie le rôle ou la responsabilité de l'objet dans l'association.
- Les associations peuvent indiquer la **navigation** avec une pointe de flèche ouverte:
  - Pas de flèche => bidirectionnelle
  - La plupart des associations sont unidirectionnelles en fin de conception.
- Les associations peuvent indiquer une **multiplicité**.

# Nommage des associations

- Une association peut être nommée afin de faciliter la compréhension des modèles. Dans ce cas le nom est indiqué au milieu du lien symbolisant l'association



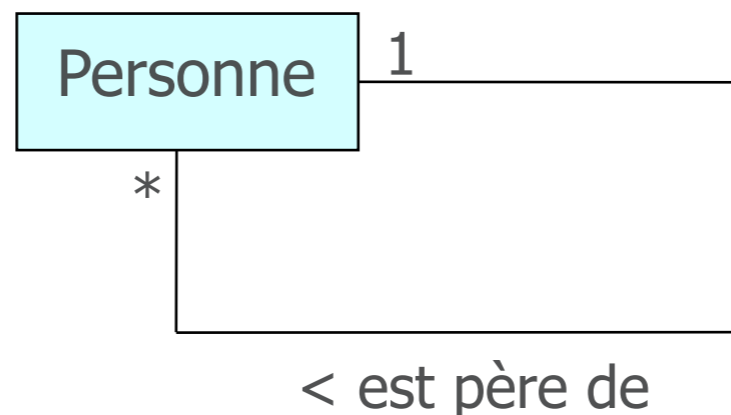
- L'usage recommande de choisir comme nom d'une association une forme verbale active (exemple : travaille pour) ou passive (exemple : est employé par)



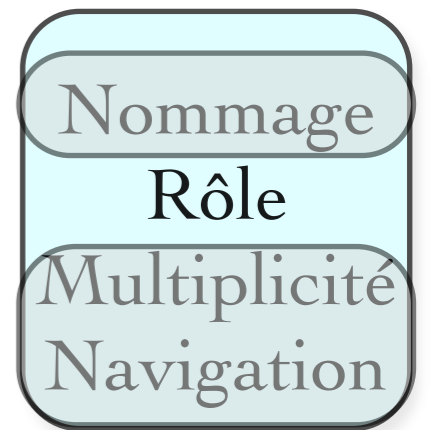
# Nommage des associations...

- Par défaut le sens de lecture du nom d'une association est de gauche à droite
- Dans le cas où la lecture du nom est ambiguë, on peut ajouter l'un des signes < ou > pour indiquer le sens de lecture

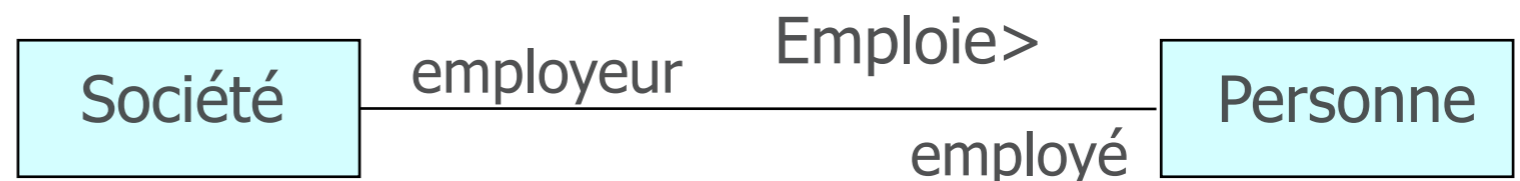
- **Exemples**



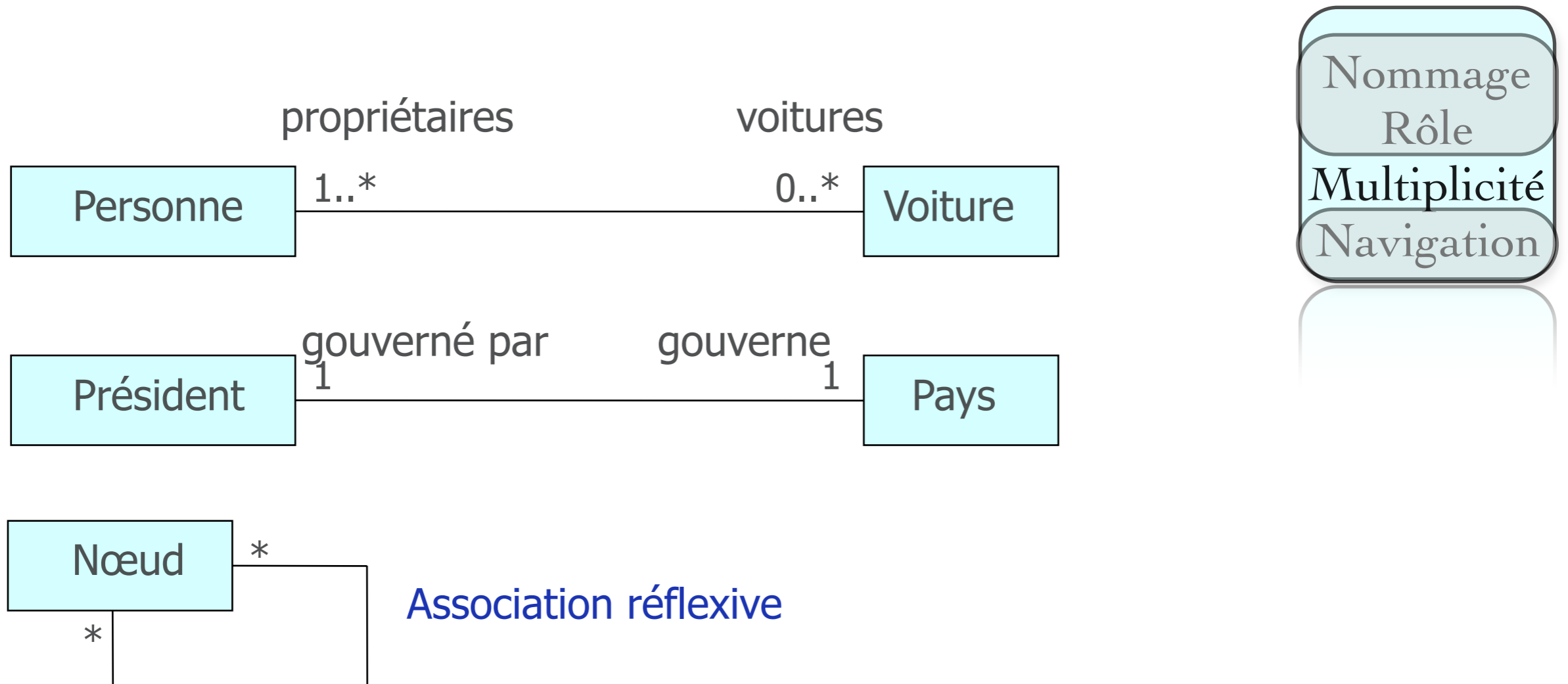
# Rôles des extrémités d'association



- On peut attribuer à une extrémité d'une association un nom appelé **rôle** qui décrit comment une classe source voit une classe destination au travers de l'association
- Le rôle est placé près de la fin de l'association et à côté de la classe à laquelle il est appliqué
- L'utilisation des rôles est optionnelle (on la préfère au nommage de l'association)
- Représentation et exemple



# Multiplicité : exemple



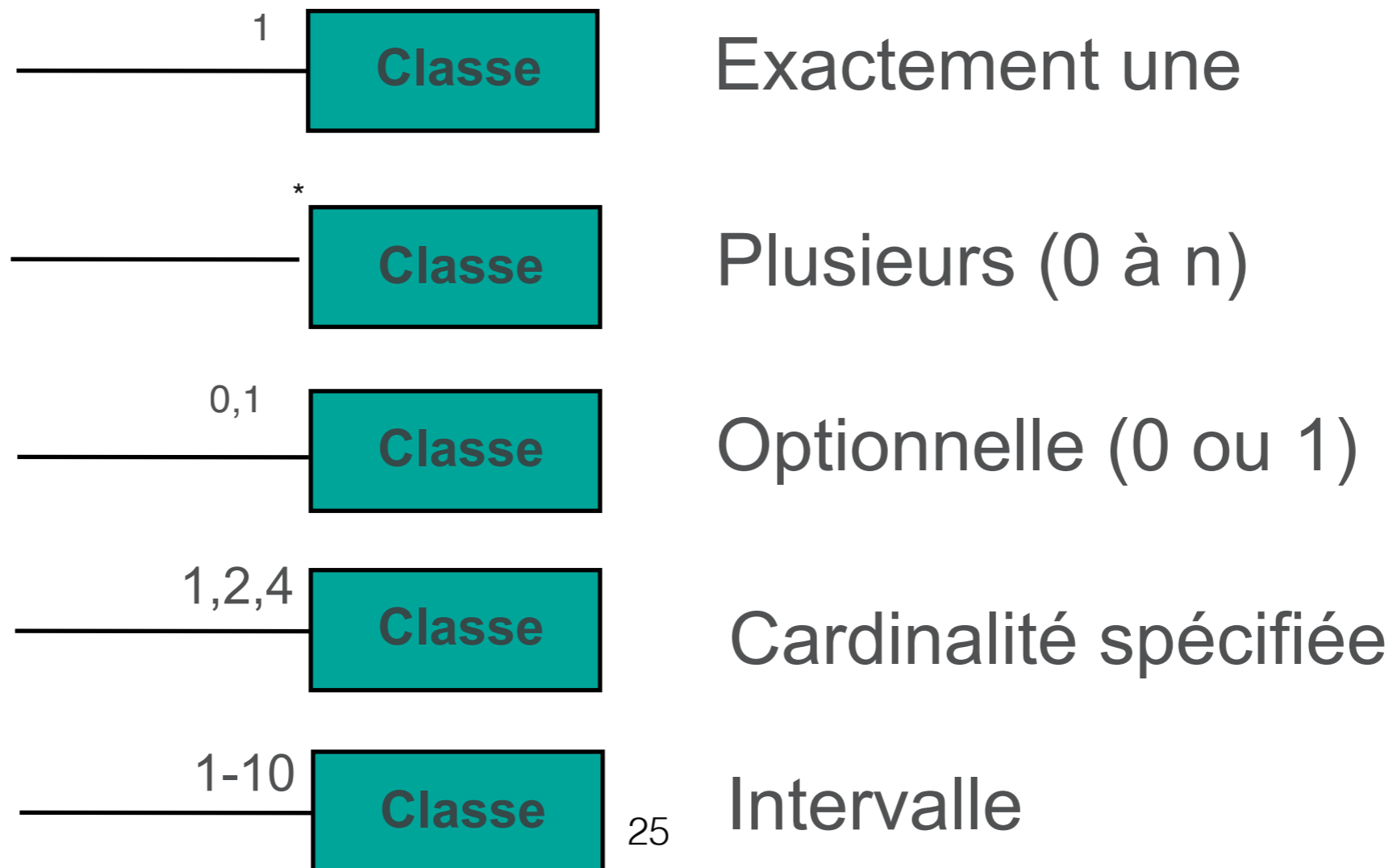
Un réseau informatique est composé de nœuds inter-connectés



# Multiplicité

La multiplicité est définie par le nombre d'objets qui participent à une relation

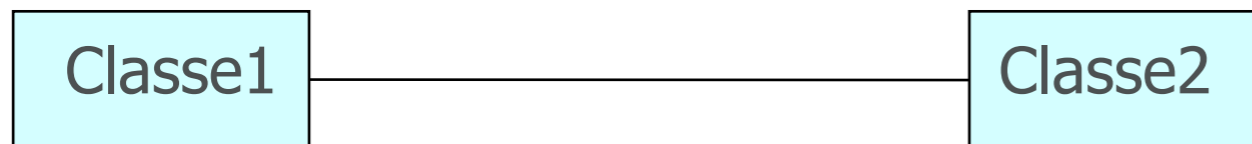
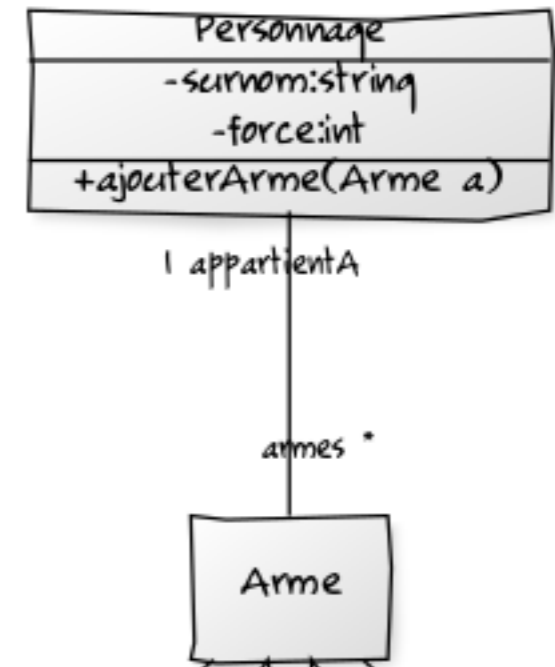
- La multiplicité est le nombre d'instances d'une classe reliées à UNE instance d'une autre classe
- Pour chaque association et agrégation, il y a deux multiplicités : une à chaque bout de la relation



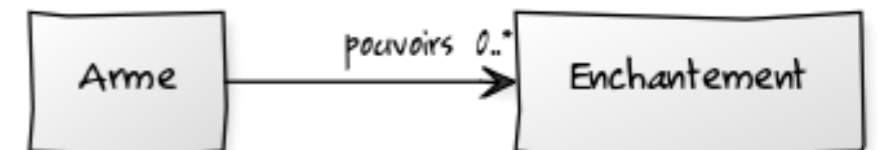
# Navigation

Nommage  
Rôle  
Multiplicité  
Navigation

- Bien que les associations soit bi-directionnelles par défaut, il peut être bon de limiter la navigation à un seul sens
  - Les objets de Classe2 sont accessibles à partir de ceux de Classe1 et vice-versa



- Si la navigation est restreinte, une flèche indique le sens de navigation : l'arme a accès à l'enchantelement, et pas l'inverse.

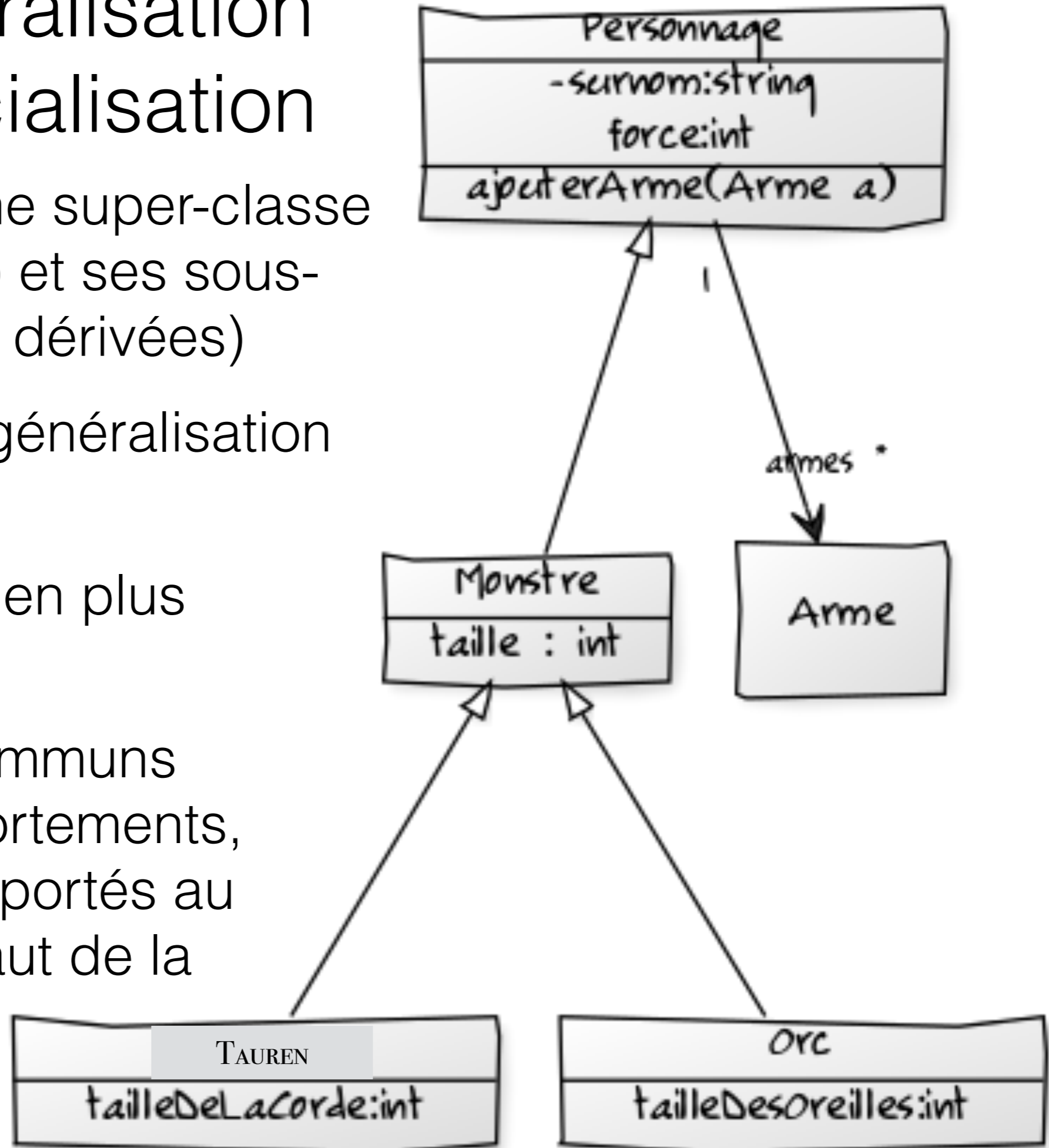




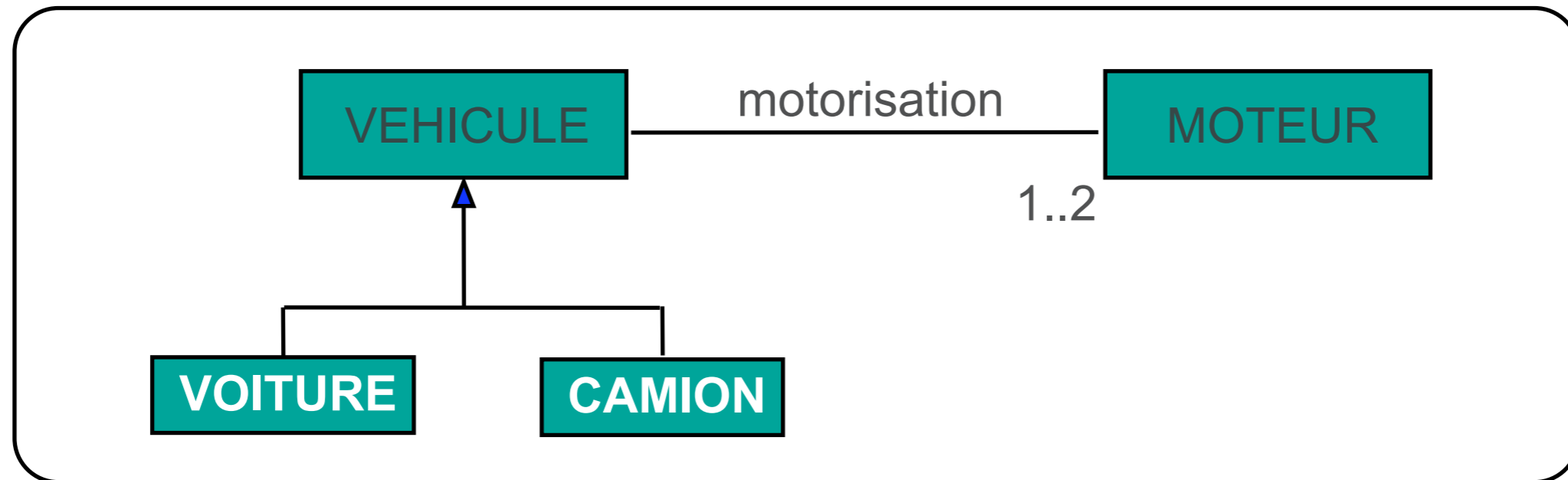
# Généralisation /Spécialisation

# Généralisation /Spécialisation

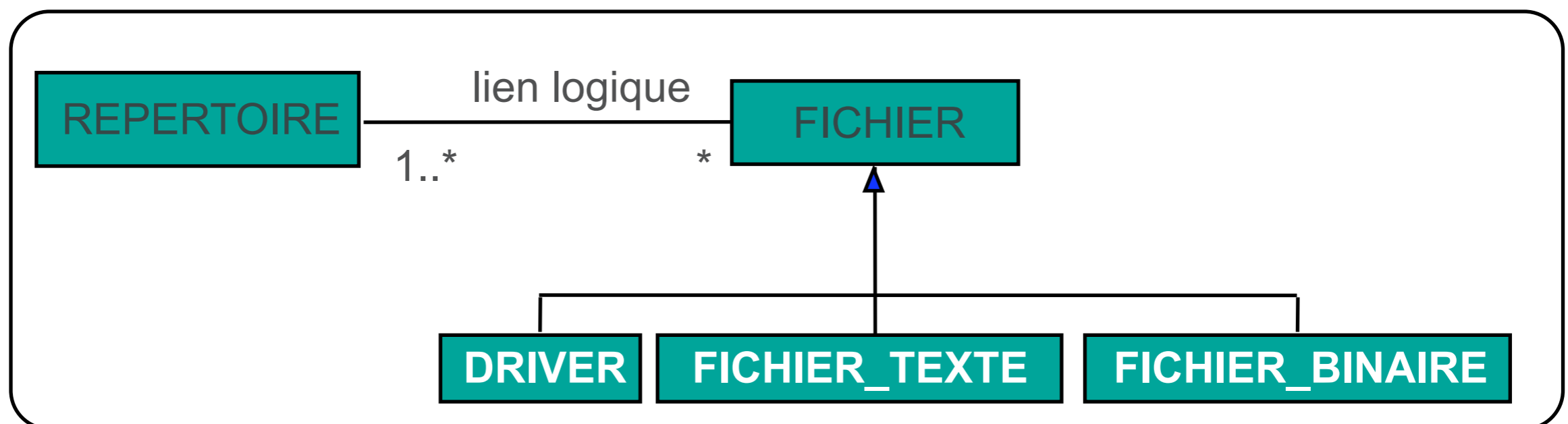
- Relation entre une super-classe (classe de base) et ses sous-classes (classes dérivées)
  - A est une généralisation de B
  - B est un A en plus spécialisé
- Les éléments communs (attributs, comportements, relations) sont reportés au niveau le plus haut de la hiérarchie



# Héritage des relations



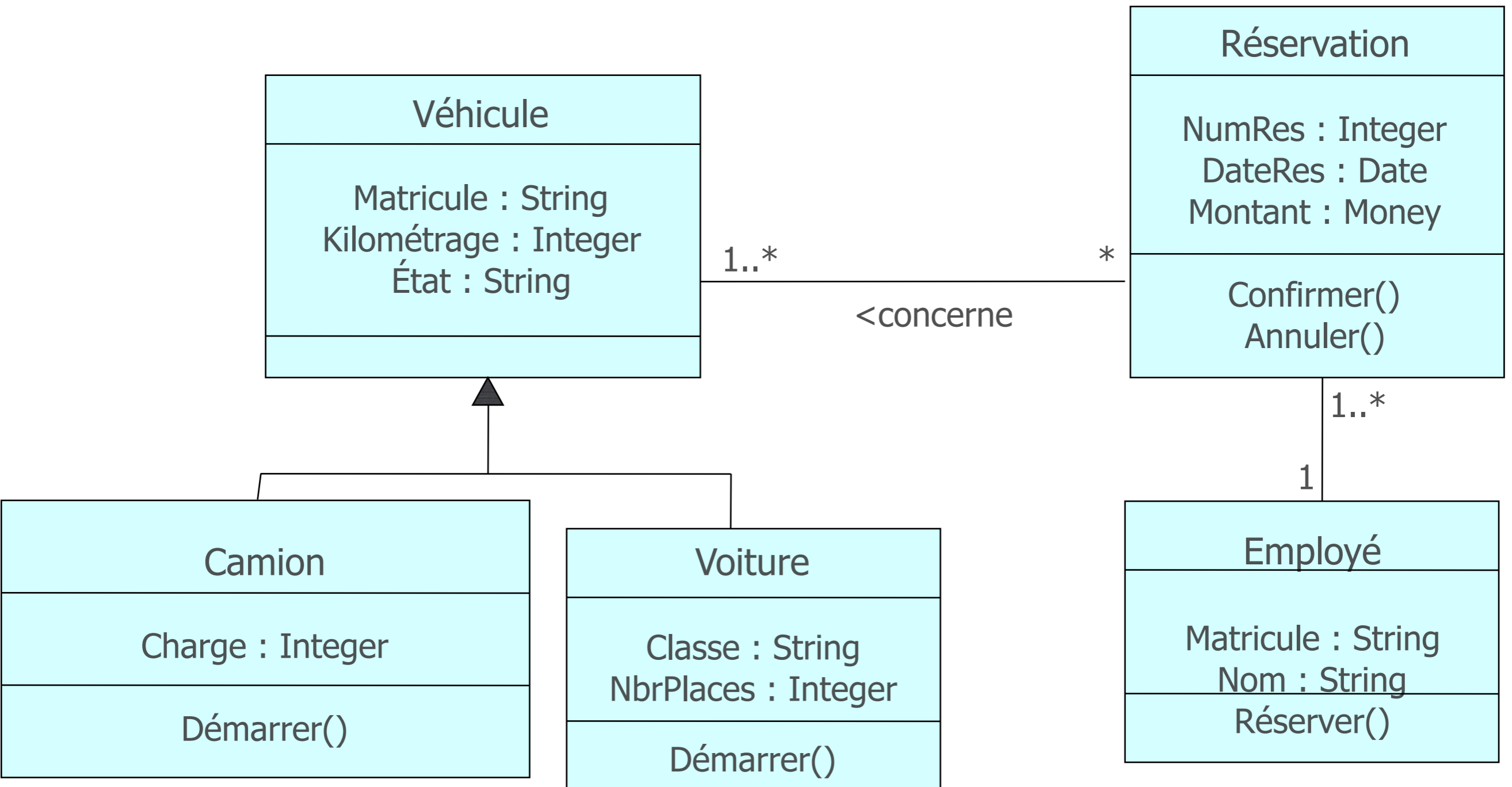
- Les relations sont héritées par les sous classes :



# Généralisation signifie :

- **Héritage**
  - L'enfant acquiert les propriétés du ou des parents (s) : les attributs, les relations et les opérations
- **Substituabilité**
  - Il est possible de substituer une instance d'une sous-classe à une instance de la classe.

# Diagramme de classes





# De la modélisation au code: Survol