



UNIVERSITÉ
DE MONTPELLIER



Les concepts OO

Ressource R3.03 — Analyse

Nadjib Lazaar (nadjib.lazaar@umontpellier.fr)

Concepts de Base de l'OO

Concept n°1 : L'objet

Concepts de Base de l'OO

Concept n°1 : L'objet

- **Entité identifiable du monde réel**
 - Objet matériel : Voiture, Appartement, ...
 - Objet immatériel : Compte bancaire, Logarithme,...

Concepts de Base de l'OO

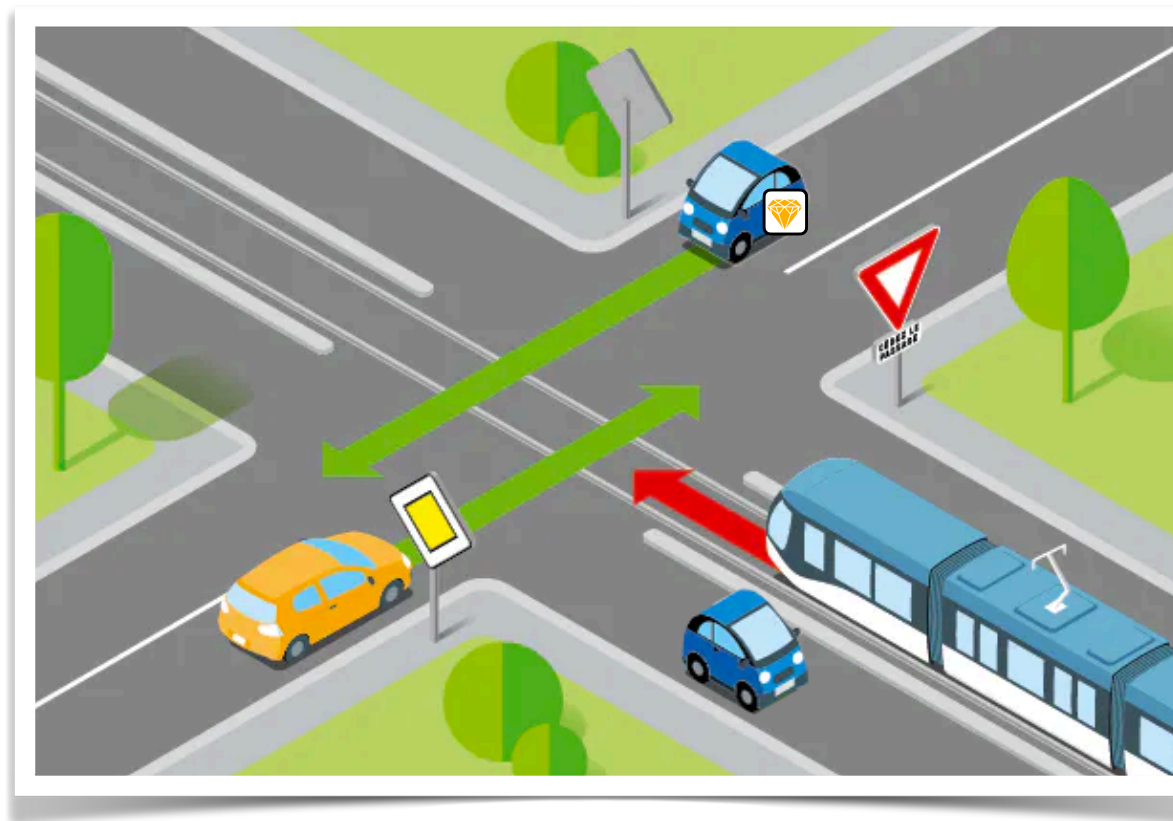
Concept n°1 : L'objet

- **Entité identifiable du monde réel**
 - Objet matériel : Voiture, Appartement, ...
 - Objet immatériel : Compte bancaire, Logarithme,...
- **Un objet possède :**
 - **Une identité** (mêmes attributs \nRightarrow même objet)
 - **Un ensemble d'attributs** qui décrivent sa structure, son état
 - **Un ensemble de méthodes** qui décrivent ses comportements

Concepts de Base de l'OO

Concept n°1 : L'objet (exemple)

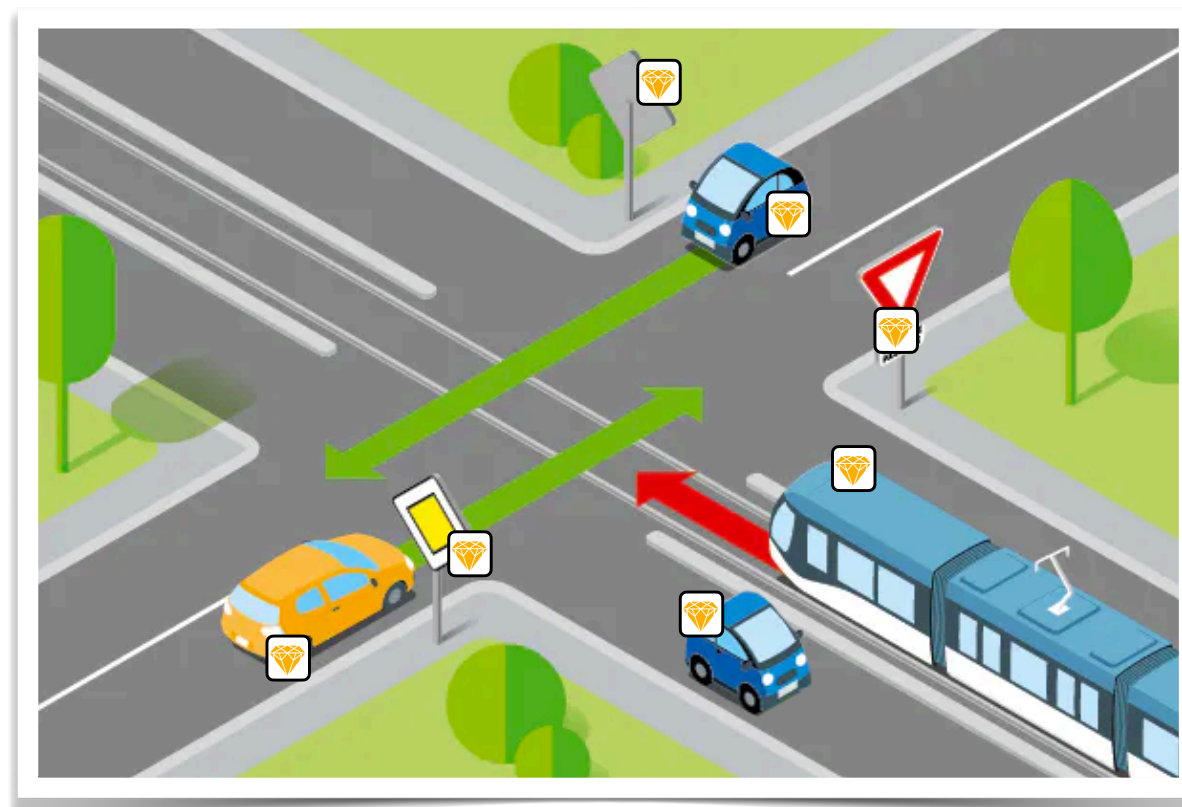
Systeme



Concepts de Base de l'OO

Concept n°1 : L'objet (exemple)

Systeme



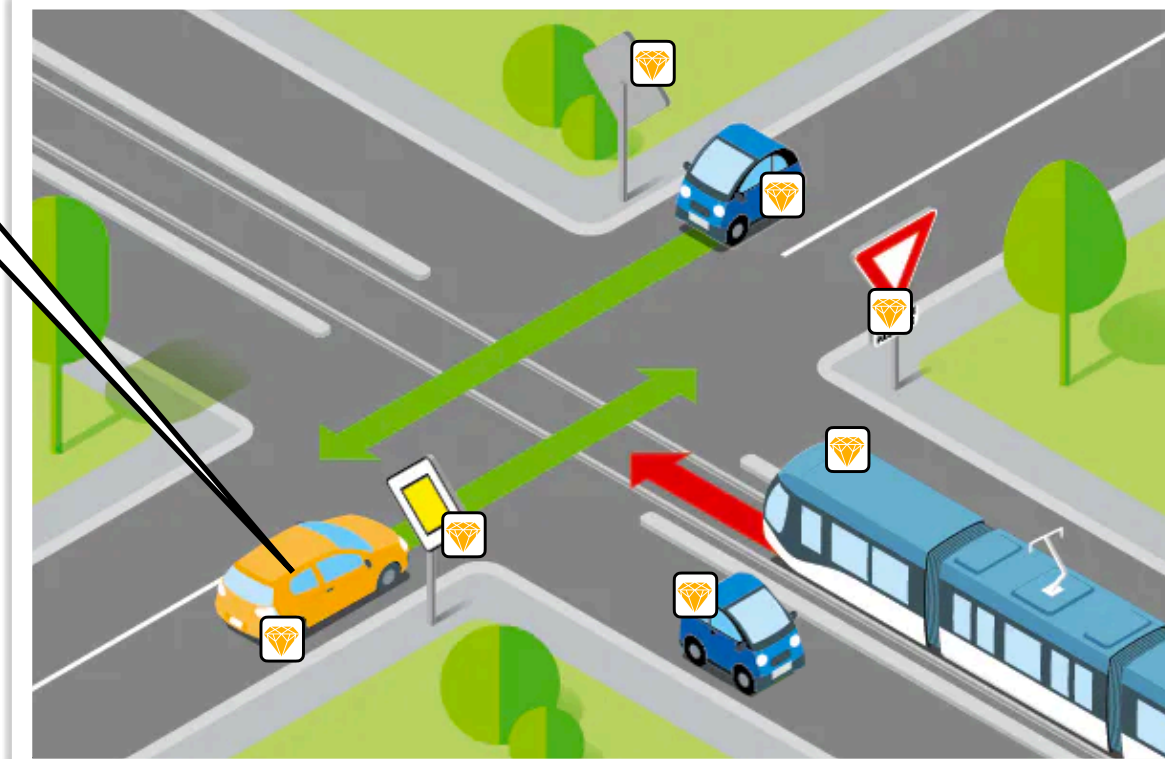
Les objets du systèmes

Concepts de Base de l'OO

Concept n°1 : L'objet (exemple)

Systeme

identité:
iut-100-ms



Les objets du systèmes

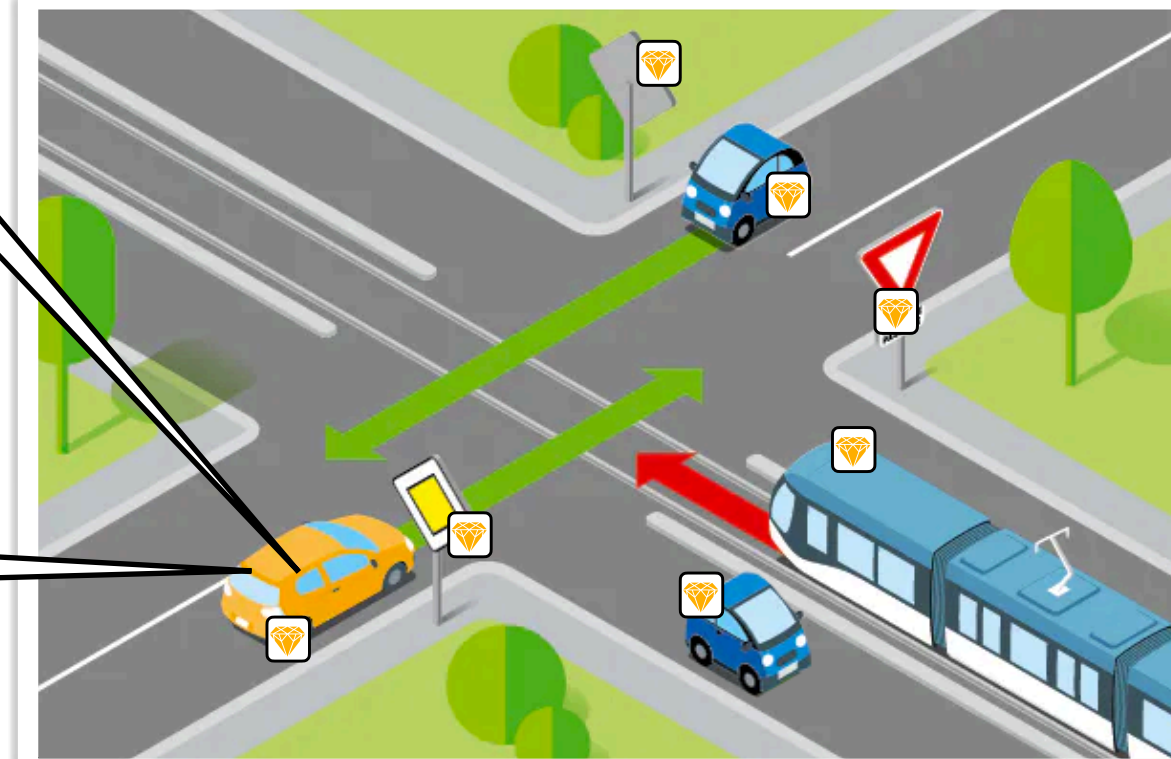
Concepts de Base de l'OO

Concept n°1 : L'objet (exemple)

Systeme

identité:
iut-100-ms

Attributs
Couleur= jaune
#portes= 5
Moteur= diesel
Position= (5,10)
Vitesse= 30km
direction= N



Les objets du systèmes

Concepts de Base de l'OO

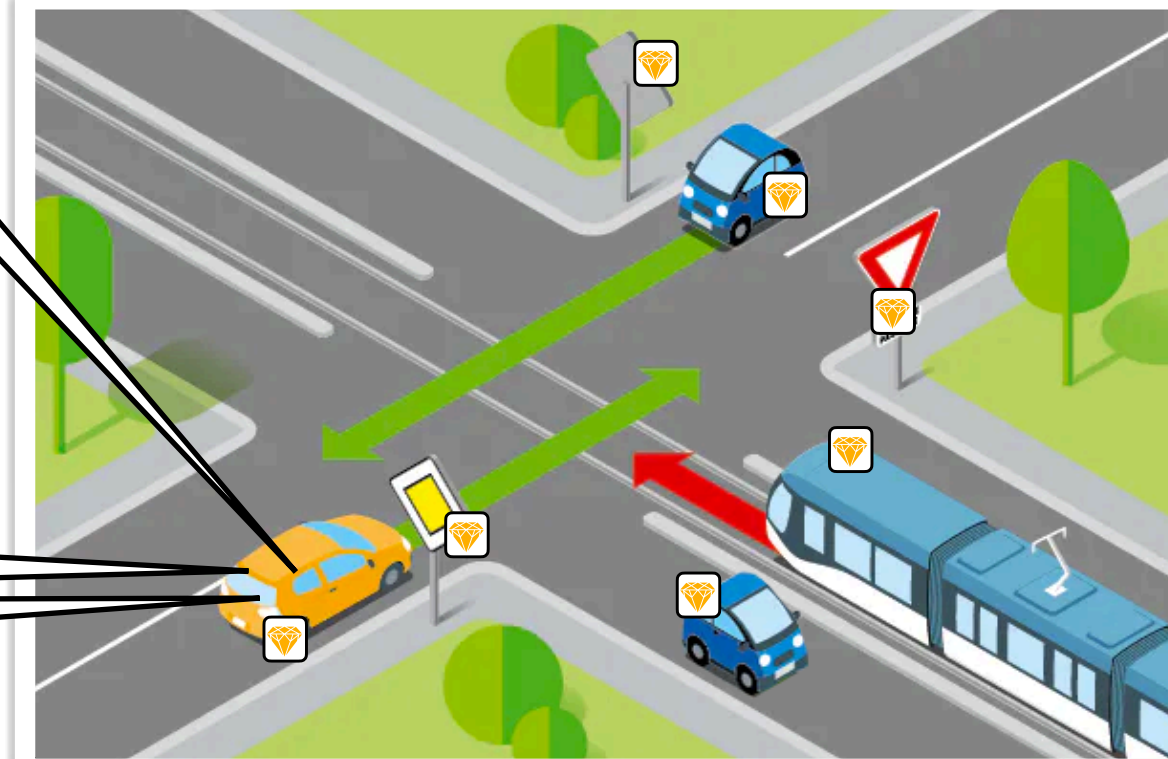
Concept n°1 : L'objet (exemple)

Système

identité:
iut-100-ms

Attributs
Couleur= jaune
#portes= 5
Moteur= diesel
Position= (5,10)
Vitesse= 30km
direction= N

méthodes
Rouler
Arret
tourner(G/D)
Klaxonner



 Les objets du systèmes

Concepts de Base de l'OO

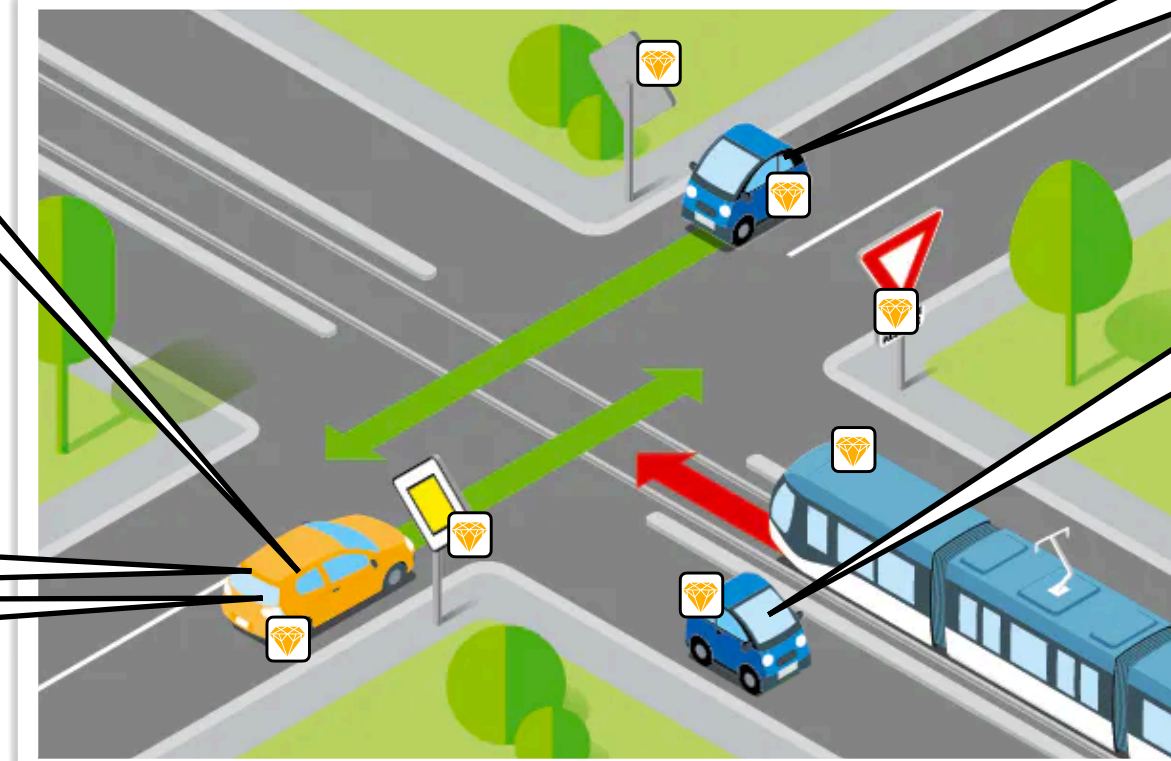
Concept n°1 : L'objet (exemple)

Systeme

identité:
iut-100-ms

Attributs
Couleur= jaune
#portes= 5
Moteur= diesel
Position= (5,10)
Vitesse= 30km
direction= N

méthodes
Rouler
Arret
tourner(G/D)
Klaxonner



identité:
iut-101-ms

≠

identité:
iut-102-ms



Les objets du systèmes

Concepts de Base de l'OO

Concept n°2 : L'abstraction

Concepts de Base de l'OO

Concept n°2 : L'abstraction

- Faire abstraction du superflu

Concepts de Base de l'OO

Concept n°2 : L'abstraction

- Faire abstraction du superflu

Concepts de Base de l'OO

Concept n°2 : L'abstraction

- Faire abstraction du superflu
- Retenir uniquement les attributs et les comportements nécessaires au bon fonctionnement du logiciel

Concepts de Base de l'OO

Concept n°2 : L'abstraction

- Faire abstraction du superflu
- Retenir uniquement les attributs et les comportements nécessaires au bon fonctionnement du logiciel

Concepts de Base de l'OO

Concept n°2 : L'abstraction

- Faire abstraction du superflu
- Retenir uniquement les attributs et les comportements nécessaires au bon fonctionnement du logiciel
- Un objet UML est une simplification d'un objet du monde réel par rapport aux besoins et aux objectifs du logiciels

Concepts de Base de l'OO

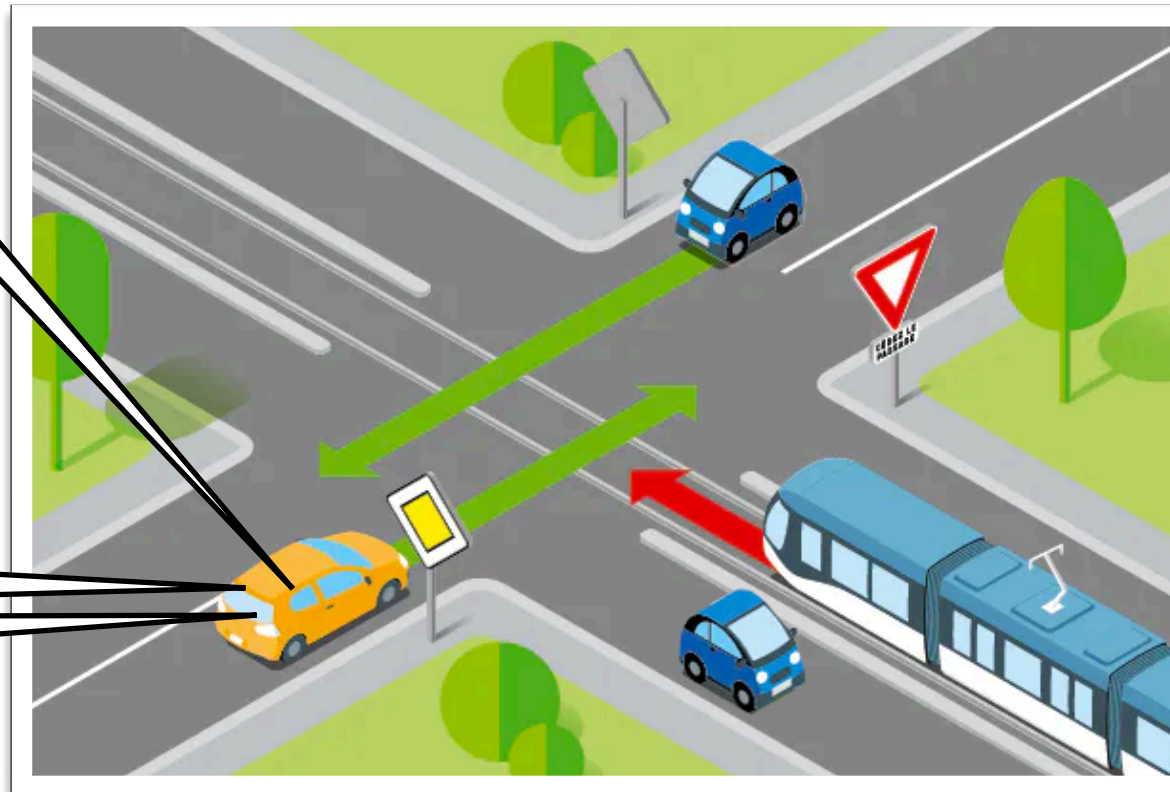
Concept n°2 : L'abstraction (exemple)

Systeme

identité:
iut-100-ms

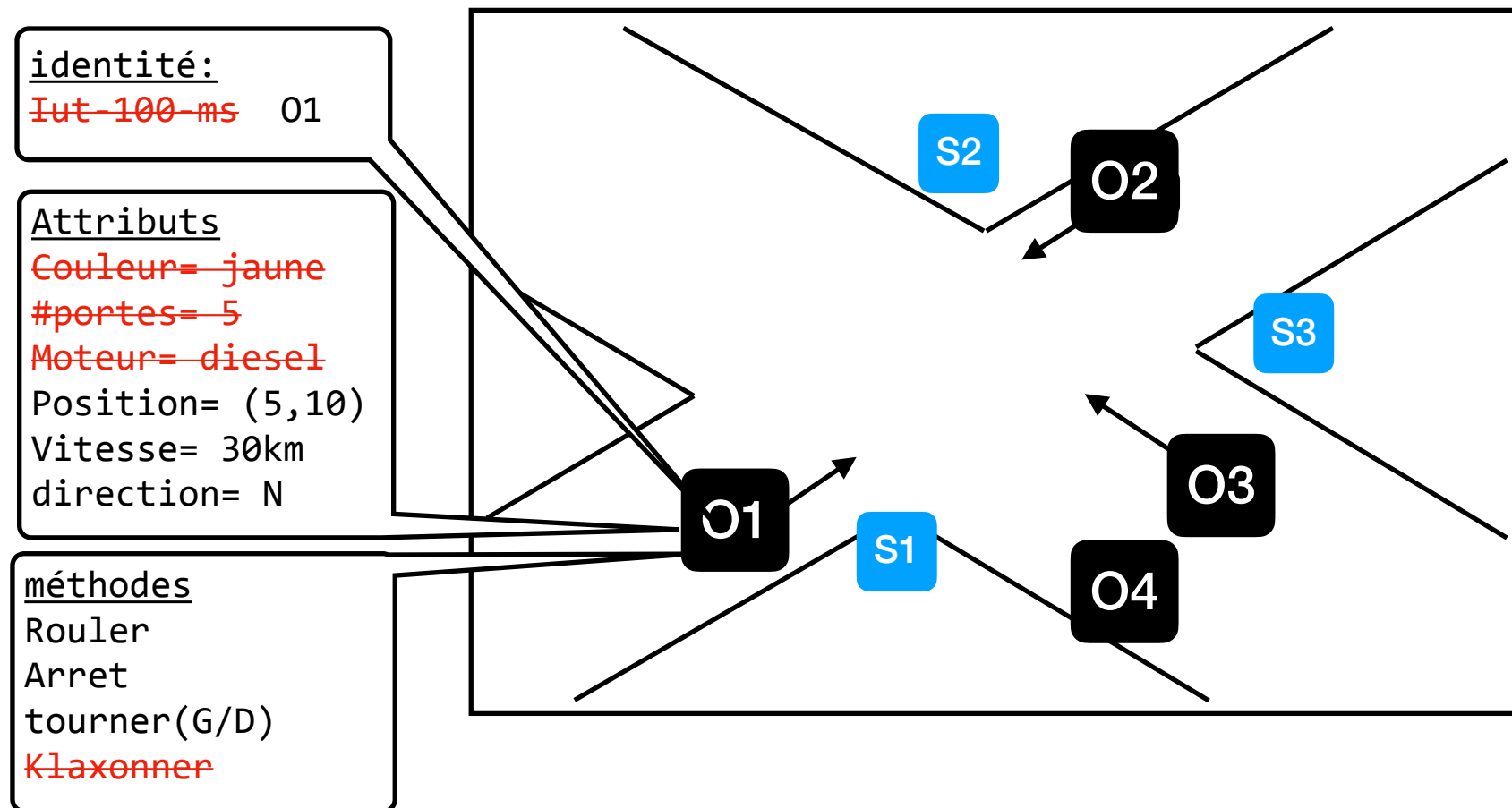
Attributs
Couleur= jaune
#portes= 5
Moteur= diesel
Position= (5,10)
Vitesse= 30km
direction= N

méthodes
Rouler
Arret
tourner(G/D)
Klaxonner



Concepts de Base de l'OO

Concept n°2 : L'abstraction (exemple)



Concepts de Base de l'OO

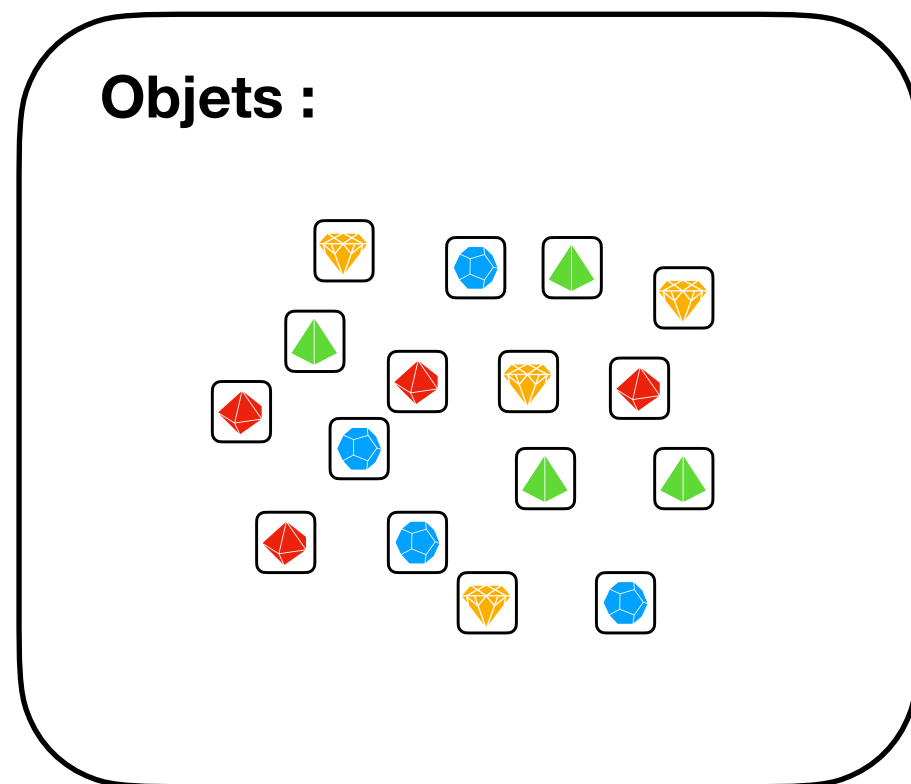
Concept n°3 : Les classes d'objets

- Regrouper l'ensemble les objets qui ont la même structure et le même comportement dans une classe

Concepts de Base de l'OO

Concept n°3 : Les classes d'objets

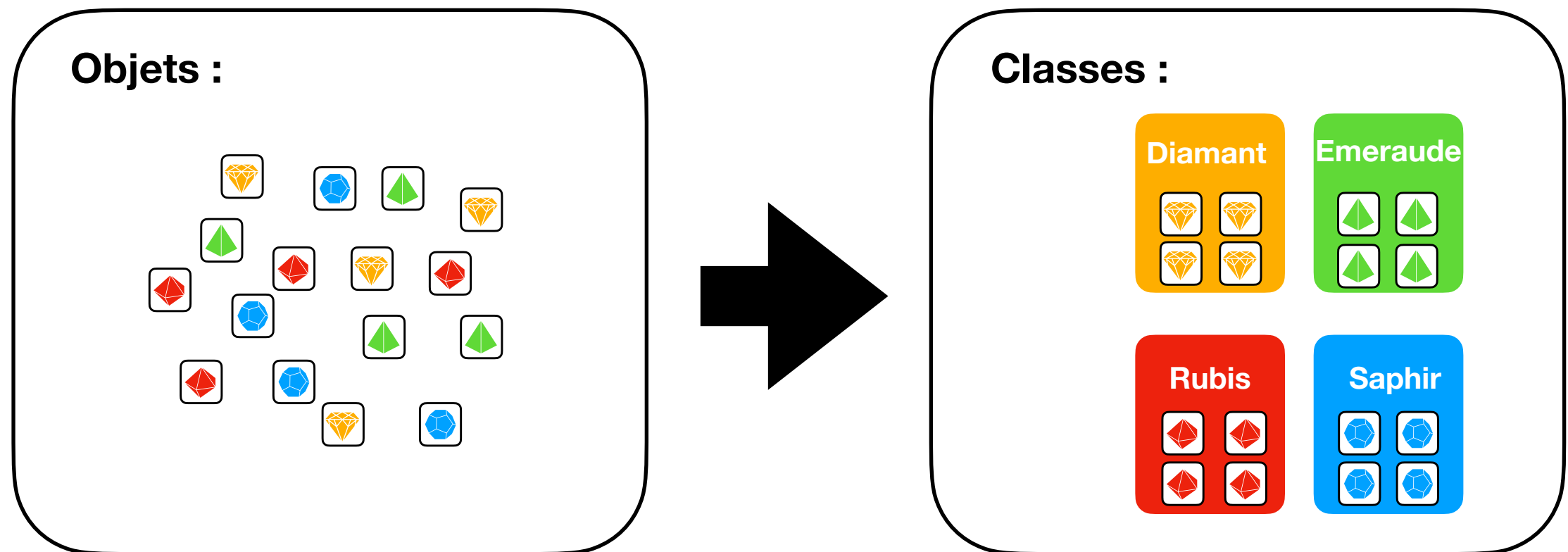
- Regrouper l'ensemble les objets qui ont la même structure et le même comportement dans une classe



Concepts de Base de l'OO

Concept n°3 : Les classes d'objets

- Regrouper l'ensemble les objets qui ont la même structure et le même comportement dans une classe



Concepts de Base de l'OO

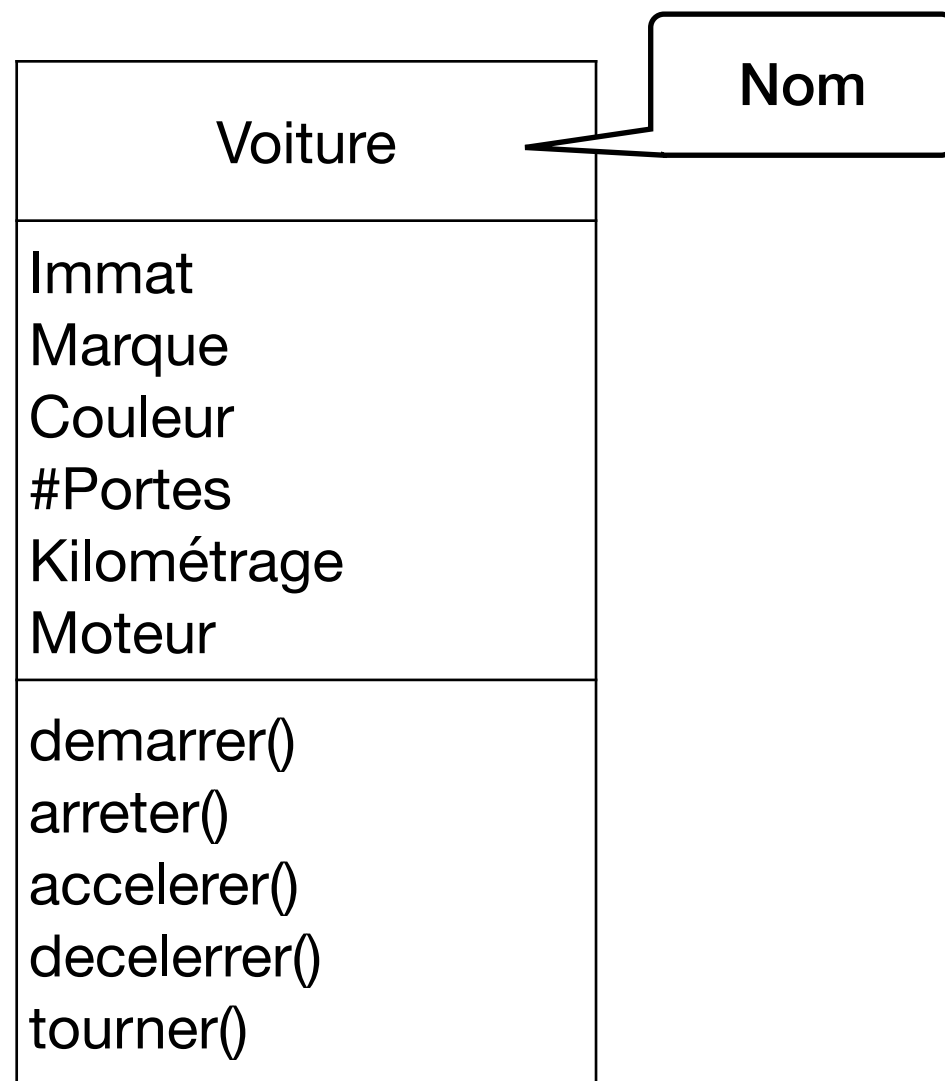
Concept n°3 : Les classes d'objets

Voiture
Immat Marque Couleur #Portes Kilométrage Moteur
demarrer() arreter() accellerer() decelerrer() tourner()

Classe

Concepts de Base de l'OO

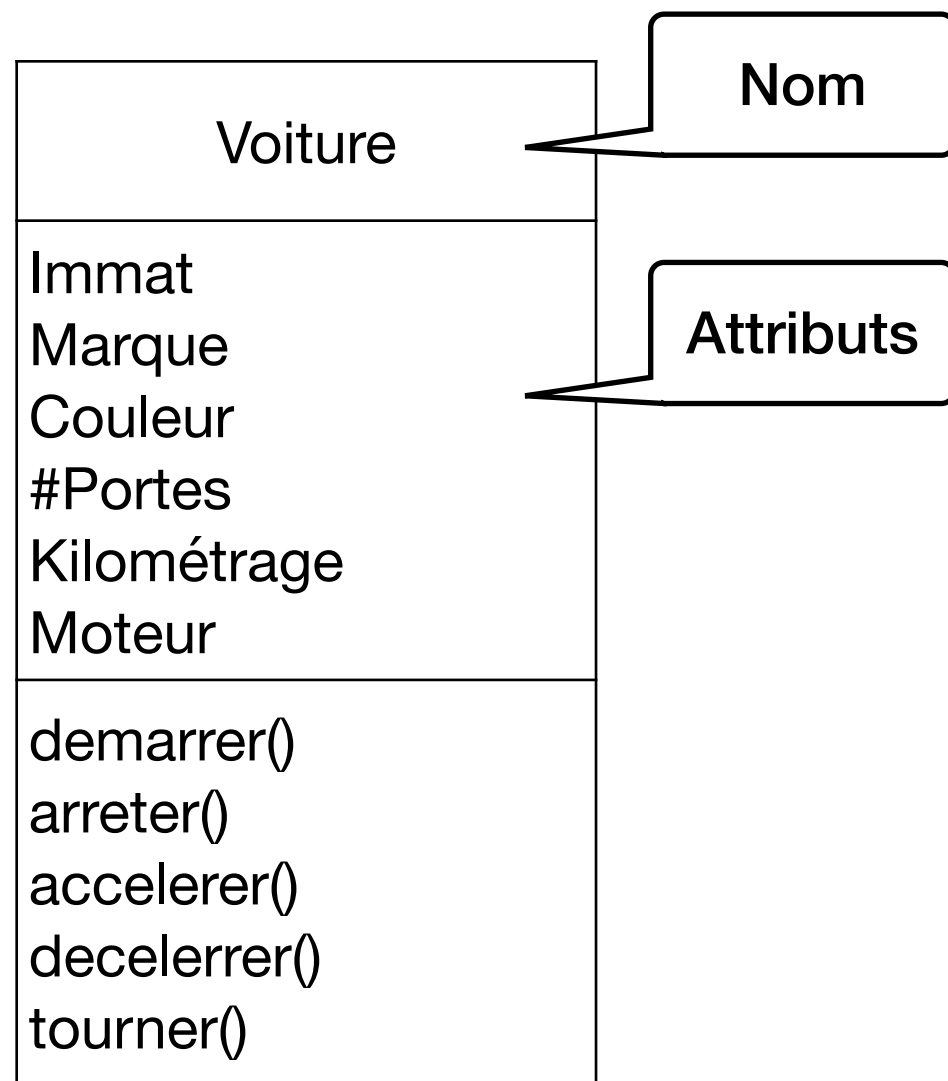
Concept n°3 : Les classes d'objets



Classe

Concepts de Base de l'OO

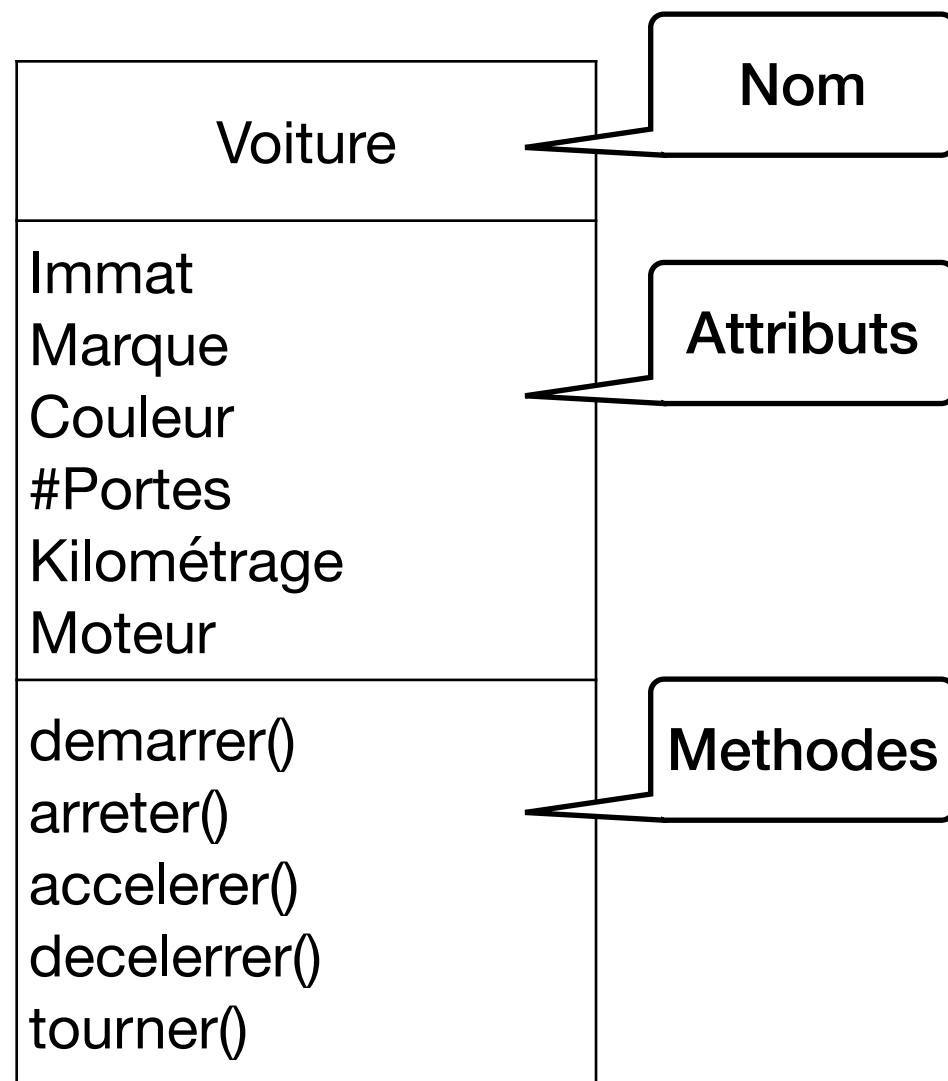
Concept n°3 : Les classes d'objets



Classe

Concepts de Base de l'OO

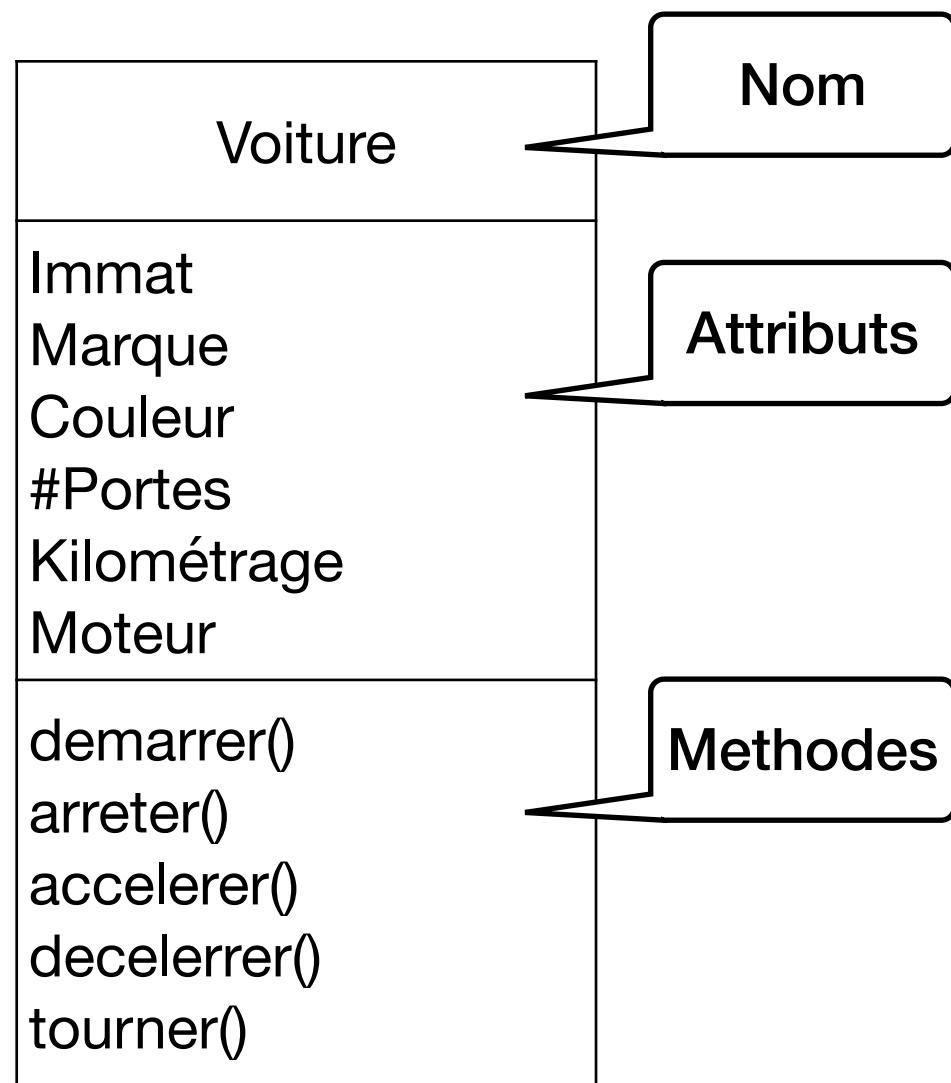
Concept n°3 : Les classes d'objets



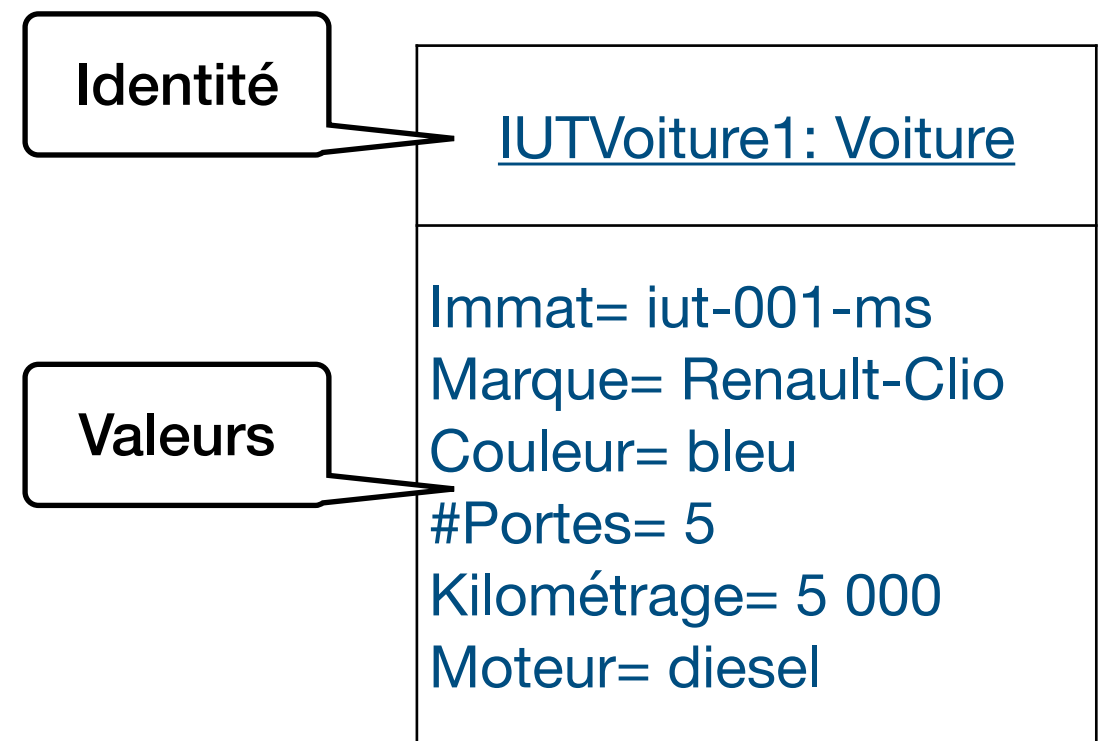
Classe

Concepts de Base de l'OO

Concept n°3 : Les classes d'objets



Classe



Objet

Concepts de Base de l'OO

Concept n°4 : L'encapsulation

Concepts de Base de l'OO

Concept n°4 : L'encapsulation

- Concept qui permet de masquer des attributs/méthodes dédiés à des traitements internes (mais pas que...)

Concepts de Base de l'OO

Concept n°4 : L'encapsulation

- Concept qui permet de masquer des attributs/méthodes dédiés à des traitements internes (mais pas que...)
- Une abstraction et une simplification de la présentation d'un objet vis-à-vis des objets extérieurs

Concepts de Base de l'OO

Concept n°4 : L'encapsulation

- Concept qui permet de masquer des attributs/méthodes dédiés à des traitements internes (mais pas que...)
- Une abstraction et une simplification de la présentation d'un objet vis-à-vis des objets extérieurs

Partie externe



[Pinterest]

Concepts de Base de l'OO

Concept n°4 : L'encapsulation

- Concept qui permet de masquer des attributs/méthodes dédiés à des traitements internes (mais pas que...)
- Une abstraction et une simplification de la présentation d'un objet vis-à-vis des objets extérieurs

Partie externe



[Pinterest]

Partie interne



[Pinterest]

Concepts de Base de l'OO

Concept n°4 : L'encapsulation (exemple)

Fraction
+numérateur +denominateur -pgcd
+somme(Fraction) +multiplication(Fraction) -simplification()

mais pas que...

Concepts de Base de l'OO

Concept n°4 : L'encapsulation (exemple)

Fraction
+numérateur -denominateur -pgcd
+somme(Fraction) +multiplication(Fraction) +getDenom() +setDenom() -simplification()

Concepts de Base de l'OO

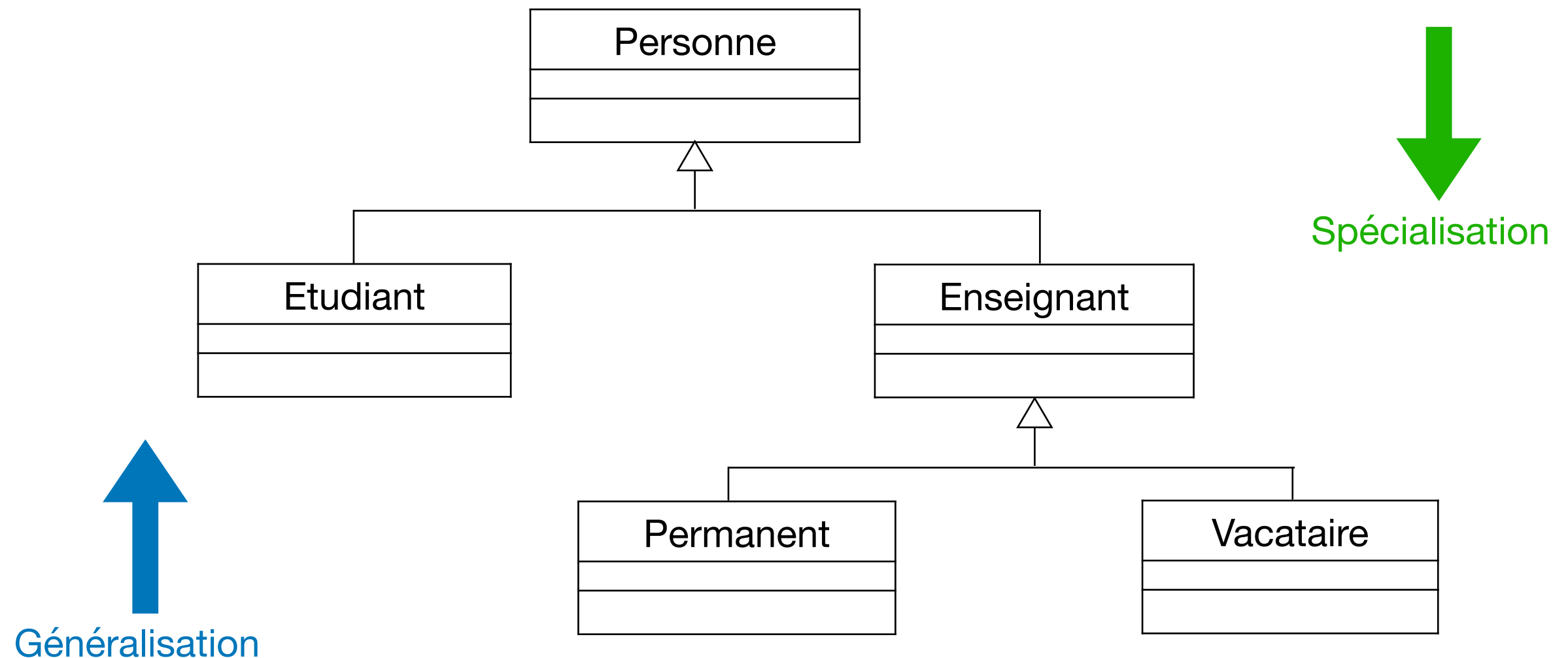
Concept n°5 : La spécialisation et la généralisation

- Un objet d'une classe peut appartenir à une sous-classe (**spécialisation**) et/ou à une super-classe (**généralisation**).

Concepts de Base de l'OO

Concept n°5 : La spécialisation et la généralisation

- Un objet d'une classe peut appartenir à une sous-classe (**spécialisation**) et/ou à une super-classe (**généralisation**).



Concepts de Base de l'OO

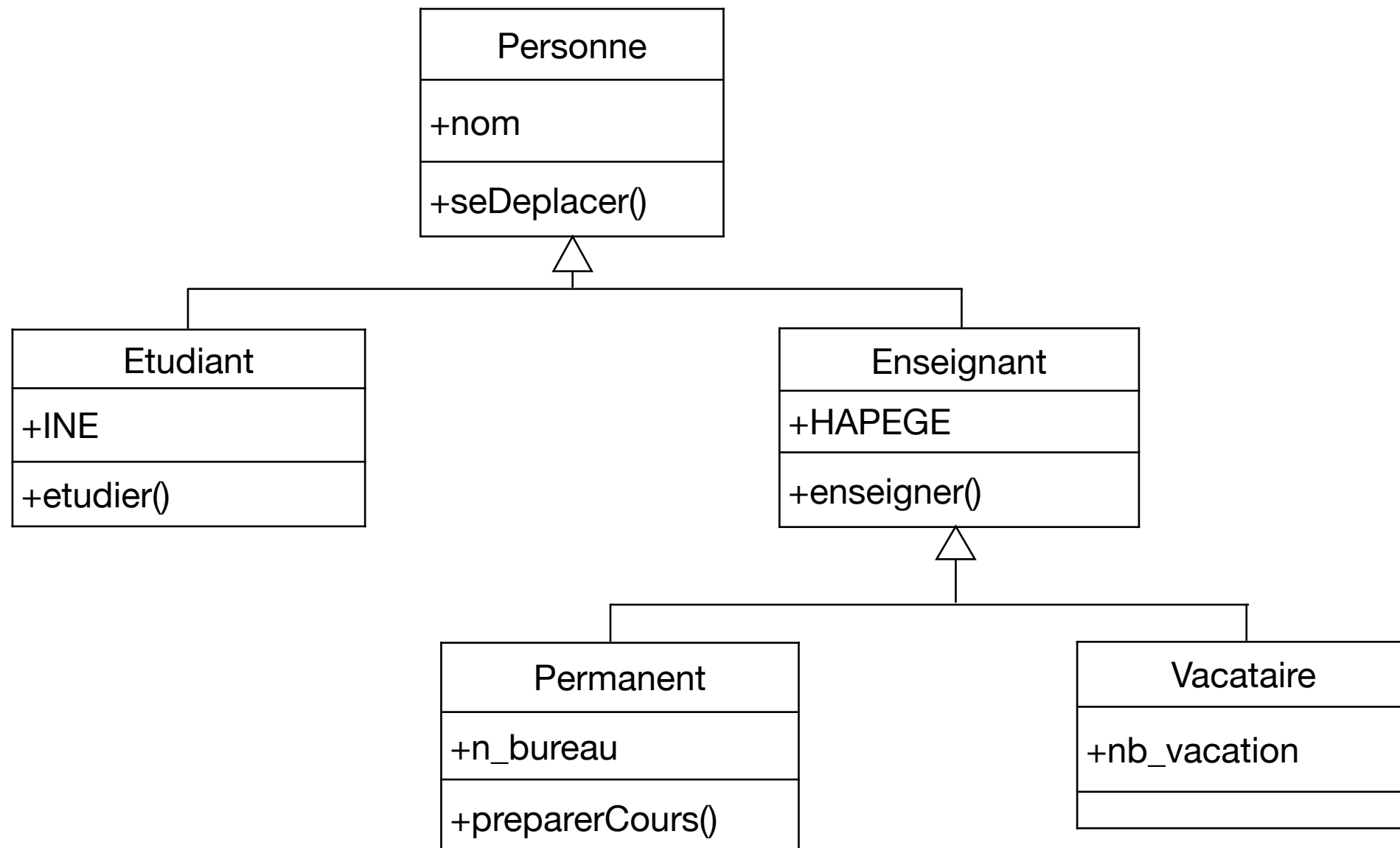
Concept n°6 : L'héritage

- Une classe hérite de sa super-classe la structure et le comportement.

Concepts de Base de l'OO

Concept n°6 : L'héritage

- Une classe hérite de sa super-classe la structure et le comportement.



Concepts de Base de l'OO

Concept n°6 : L'héritage

- Une classe hérite de sa super-classe la structure et le comportement.

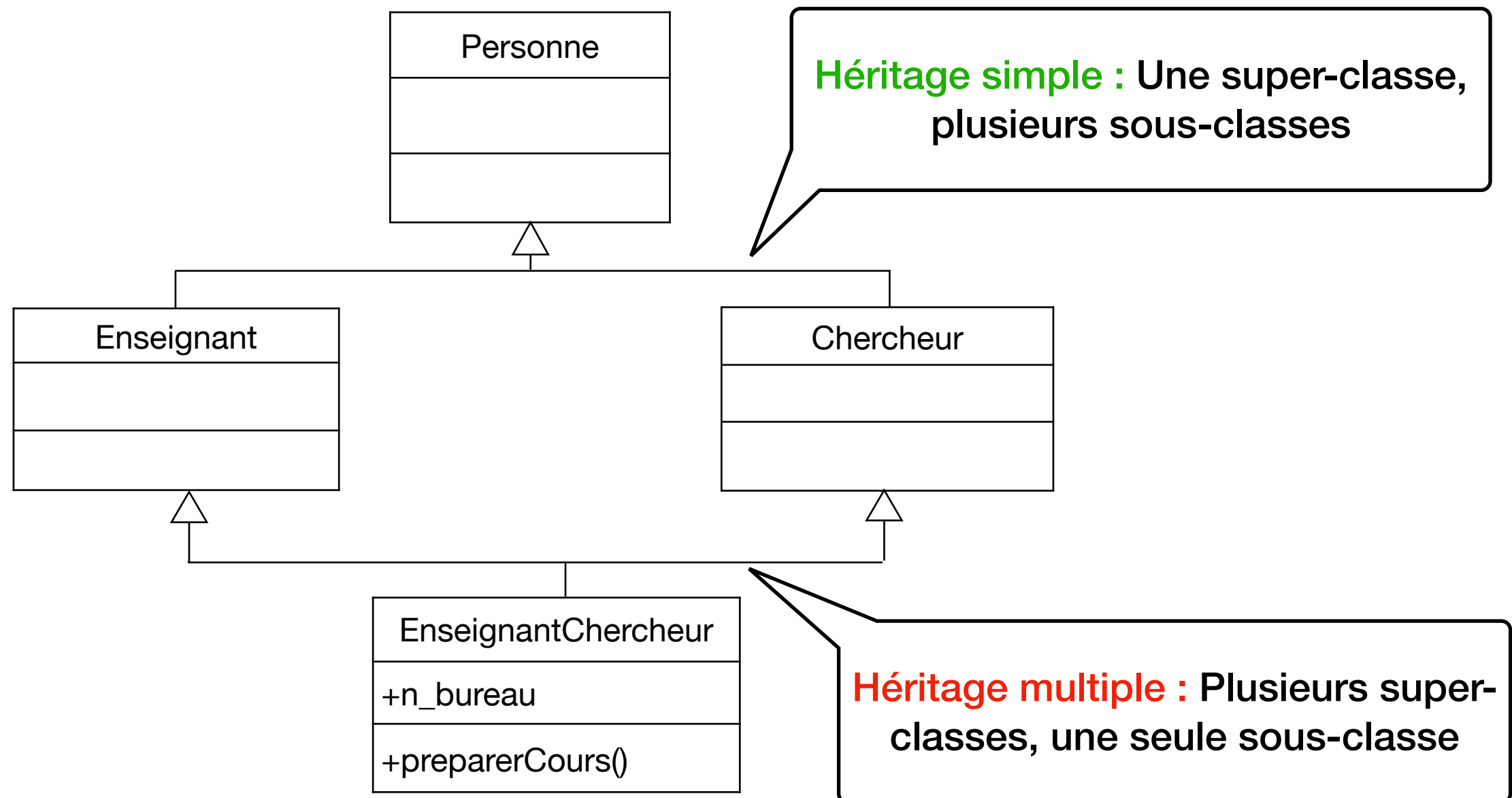
Héritage simple : Une super-classe, plusieurs sous-classes

Héritage multiple : Plusieurs super-classes, une seule sous-classe

Concepts de Base de l'OO

Concept n°6 : L'héritage

- Une classe hérite de sa super-classe la structure et le comportement.



Concepts de Base de l'OO

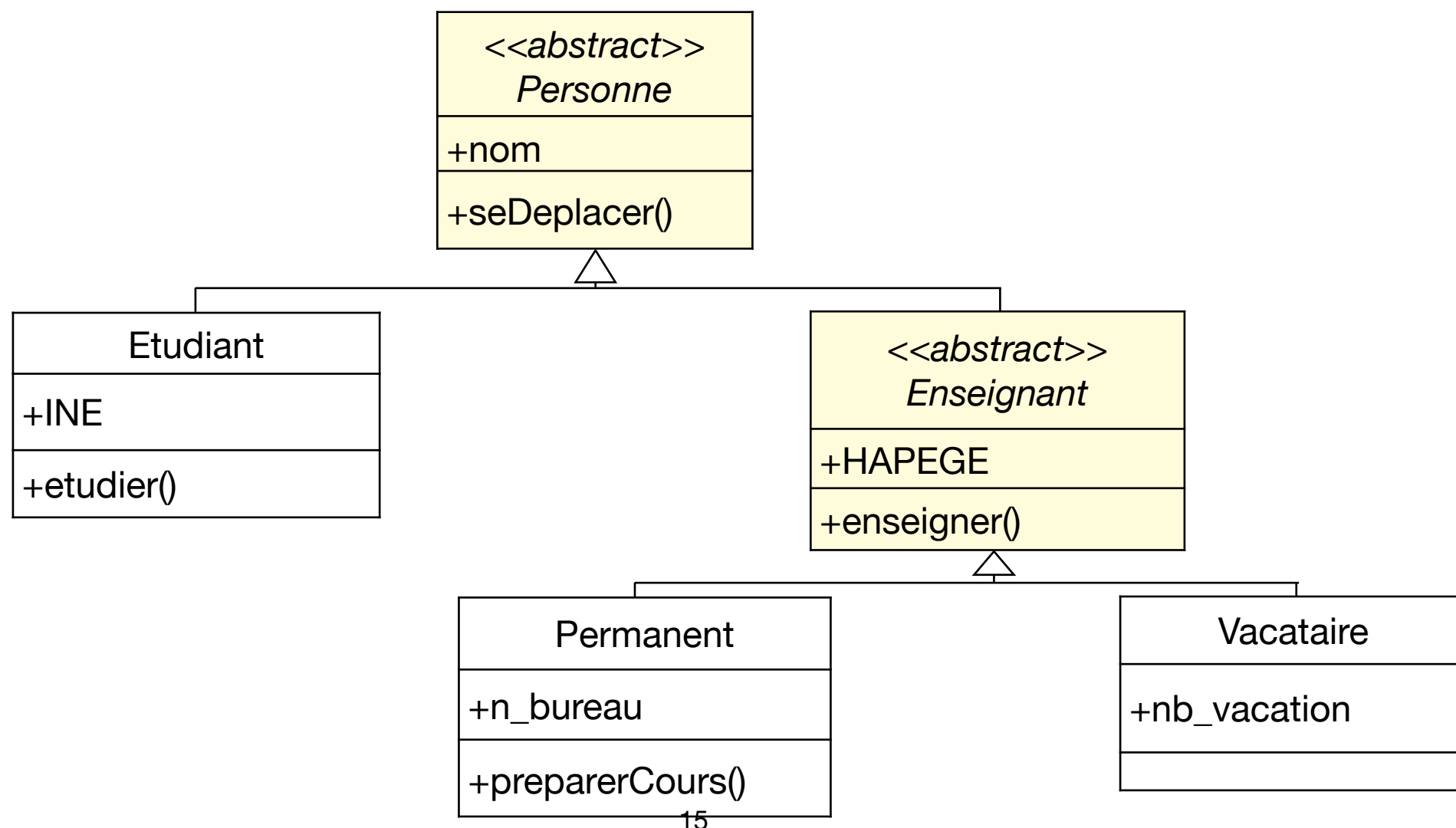
Concept n°7 : Les classes abstraites et concrètes

- Classe abstraite possède des sous-classes concrètes
- Permet de factoriser des attributs et des méthodes des sous-classes

Concepts de Base de l'OO

Concept n°7 : Les classes abstraites et concrètes

- Classe abstraite possède des sous-classes concrètes
- Permet de factoriser des attributs et des méthodes des sous-classes



Concepts de Base de l'OO

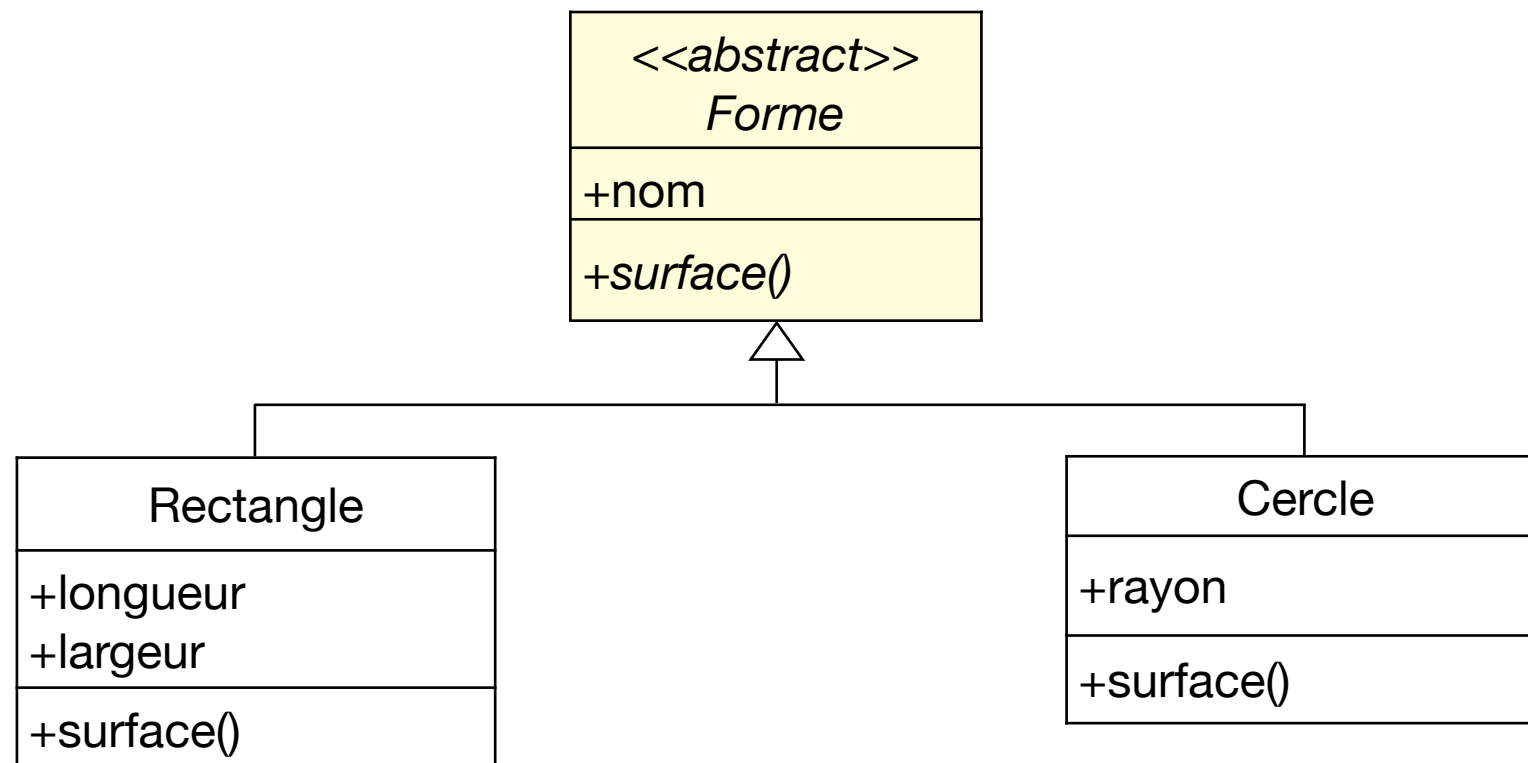
Concept n°8 : Le polymorphisme

- Comportement abstrait au niveau d'une classe qui se concrétise de différentes manières selon les sous-classes

Concepts de Base de l'OO

Concept n°8 : Le polymorphisme

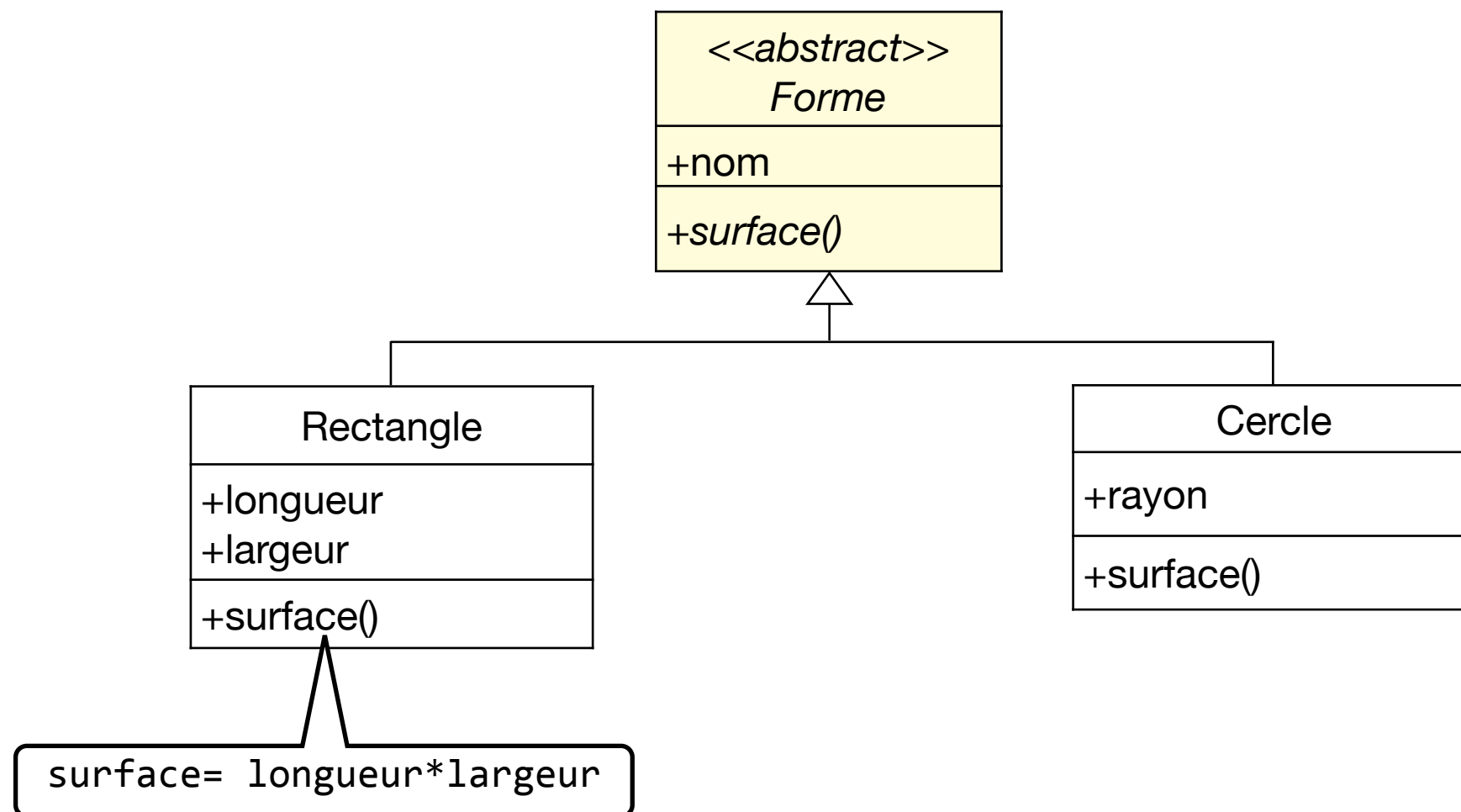
- Comportement abstrait au niveau d'une classe qui se concrétise de différentes manières selon les sous-classes



Concepts de Base de l'OO

Concept n°8 : Le polymorphisme

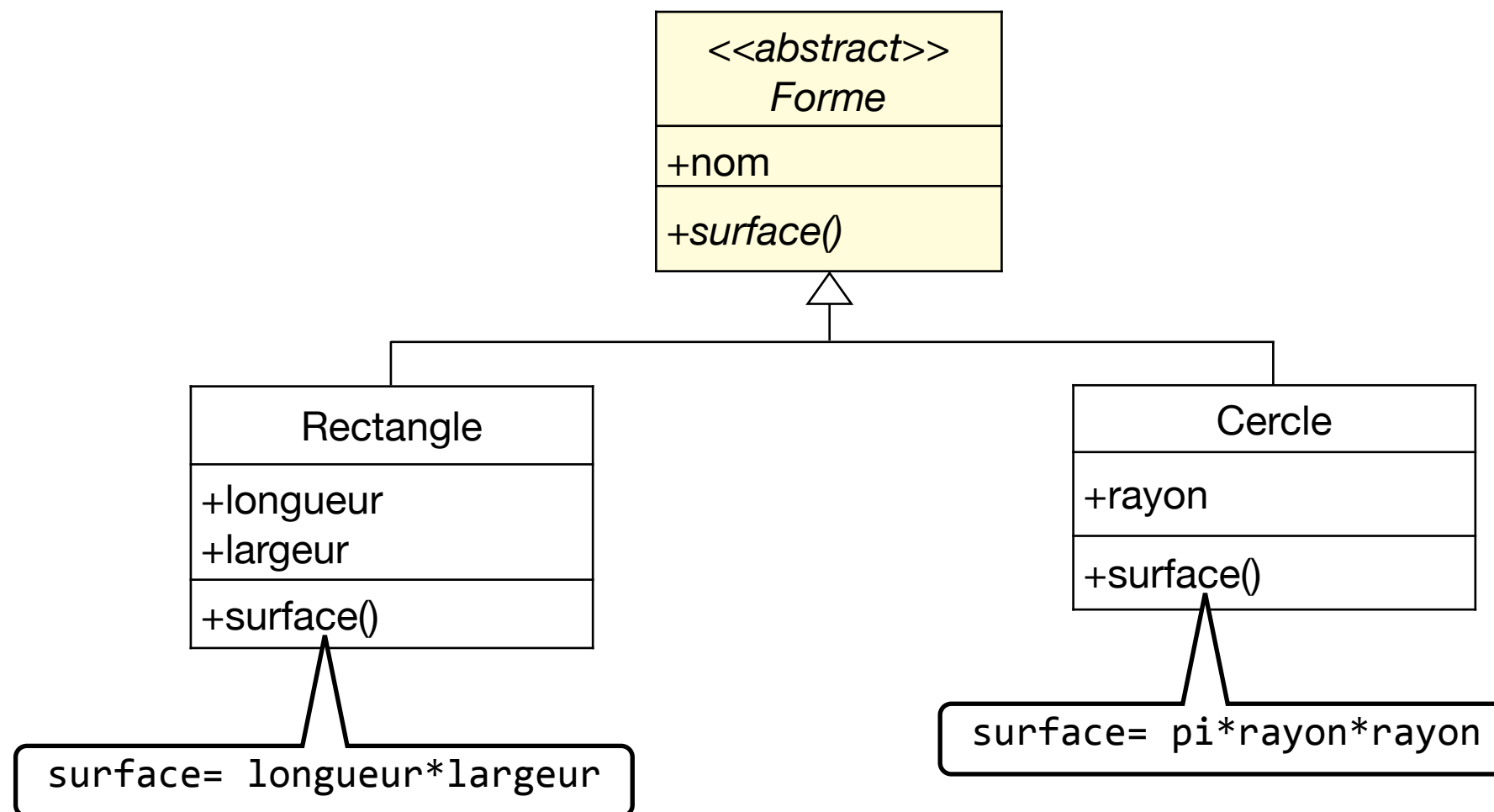
- Comportement abstrait au niveau d'une classe qui se concrétise de différentes manières selon les sous-classes



Concepts de Base de l'OO

Concept n°8 : Le polymorphisme

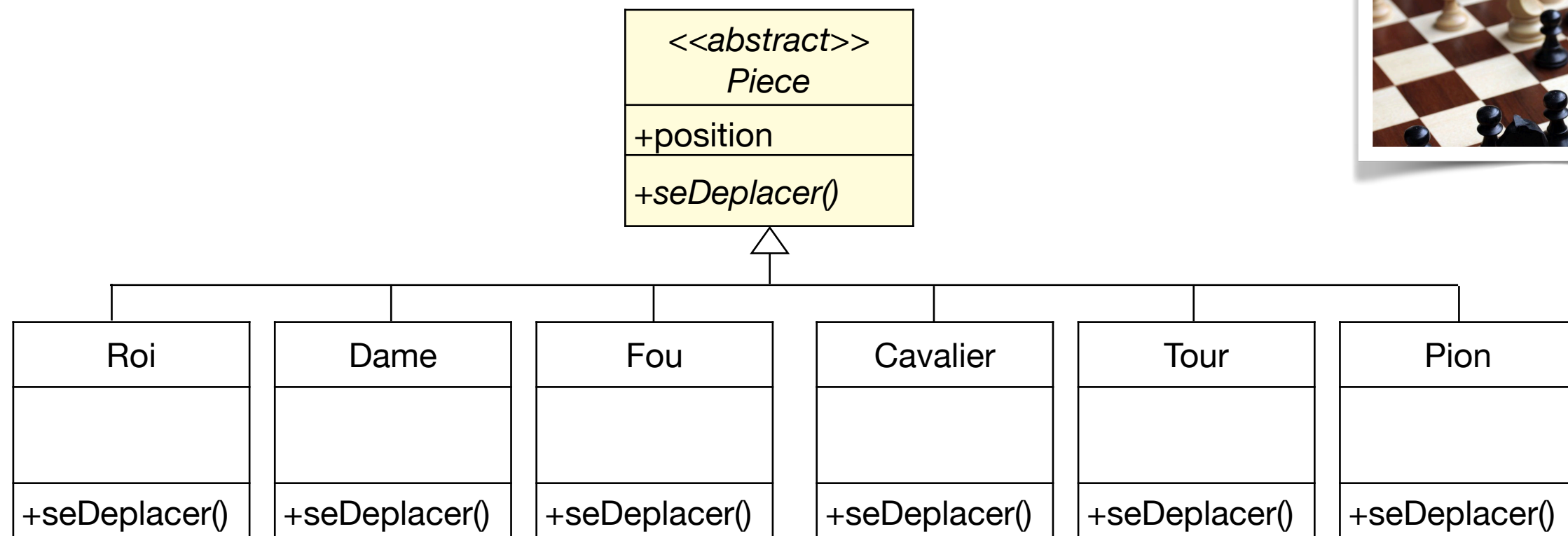
- Comportement abstrait au niveau d'une classe qui se concrétise de différentes manières selon les sous-classes



Concepts de Base de l'OO

Concept n°8 : Le polymorphisme

- Comportement abstrait au niveau d'une classe qui se concrétise de différentes manières selon les sous-classes



Concepts de Base de l'OO

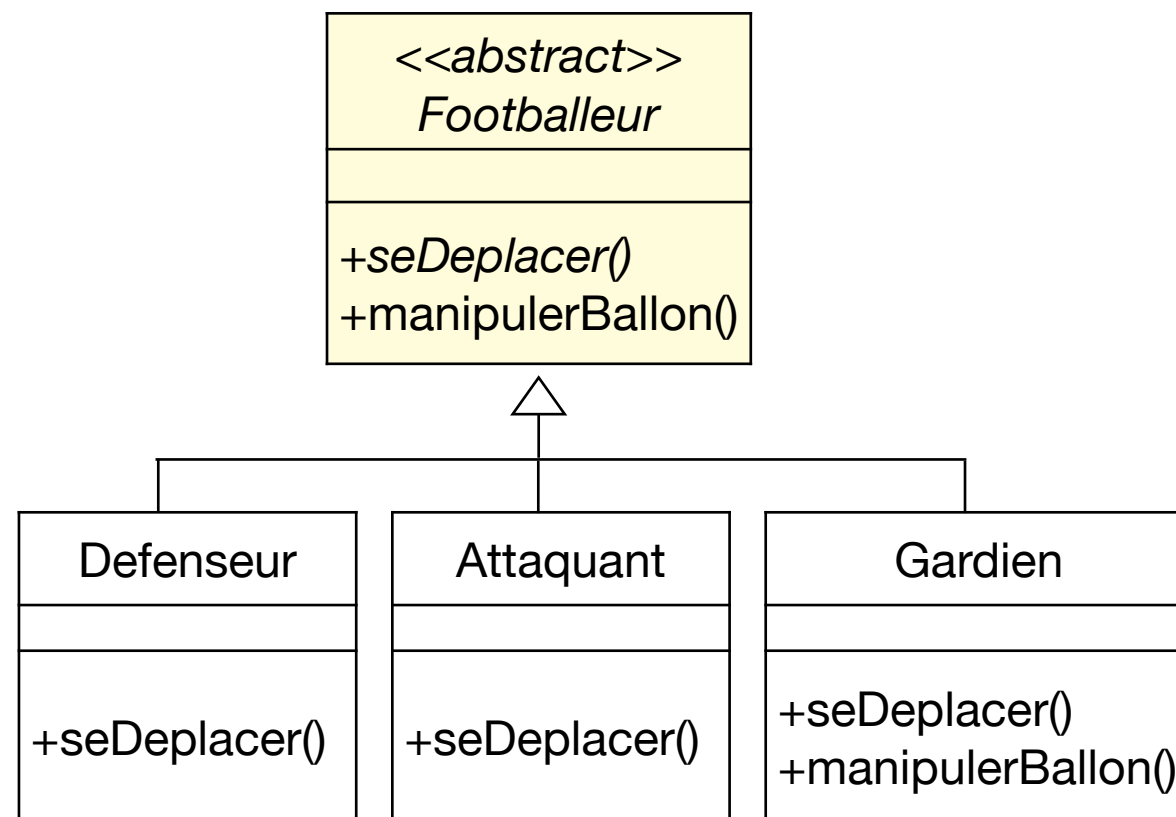
Concept n°8 : Le polymorphisme

- Redéfinition (overriding) des méthodes : définition d'une méthode dans la super-classe et possibilité de la redéfinir localement dans la sous-classe
- Trois raisons valables pour redéfinir une méthode: restriction, extension et optimisation.

Concepts de Base de l'OO

Concept n°8 : Le polymorphisme

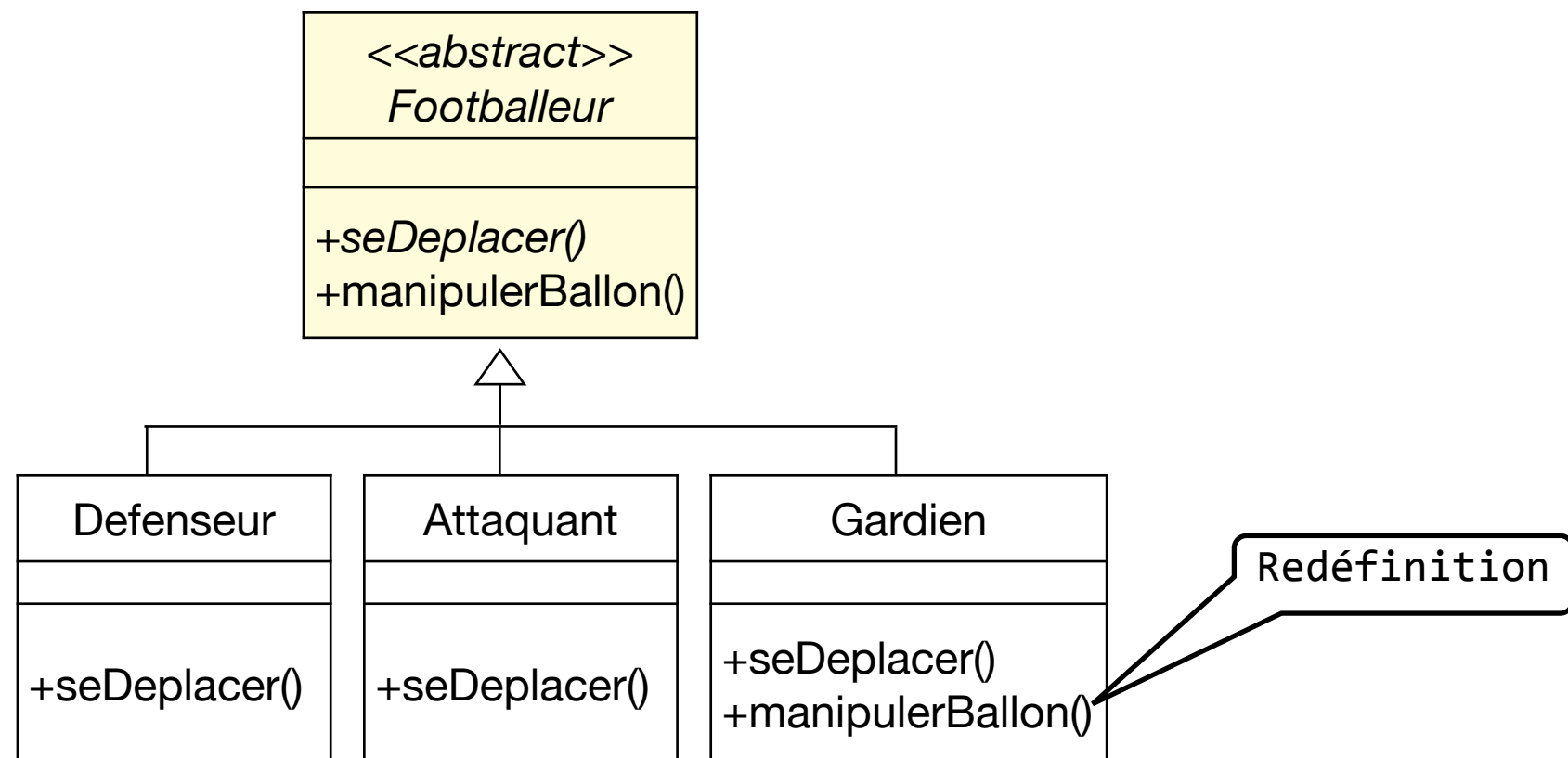
- Redéfinition (overriding) des méthodes : définition d'une méthode dans la super-classe et possibilité de la redéfinir localement dans la sous-classe
- Trois raisons valables pour redéfinir une méthode: restriction, extension et optimisation.



Concepts de Base de l'OO

Concept n°8 : Le polymorphisme

- Redéfinition (overriding) des méthodes : définition d'une méthode dans la super-classe et possibilité de la redéfinir localement dans la sous-classe
- Trois raisons valables pour redéfinir une méthode: restriction, extension et optimisation.



Concepts de Base de l'OO

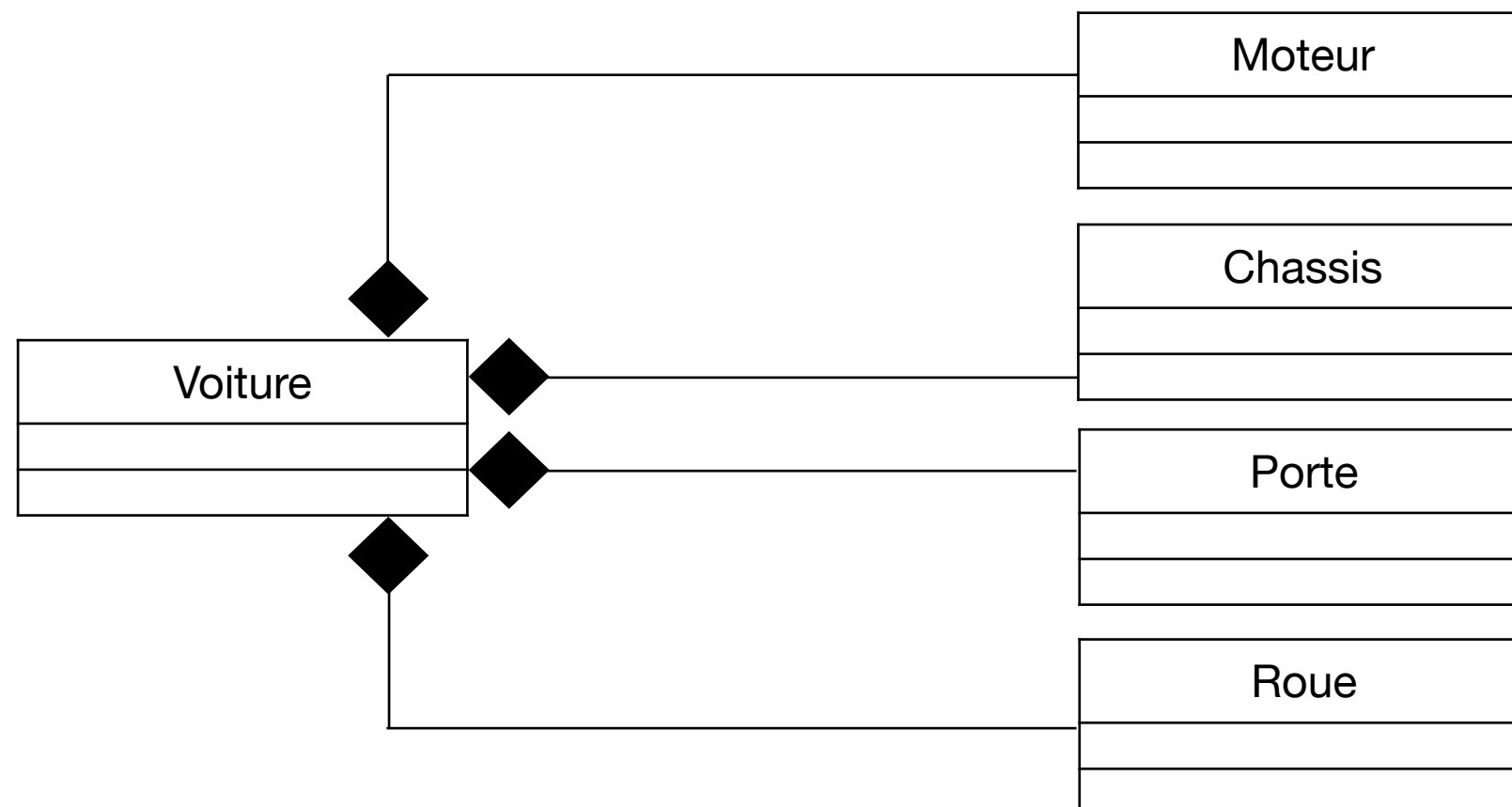
Concept n°9 : La composition

- Un objet peut être complexe et par conséquent une composition d'objets
- Un composé est formé à partir de composants

Concepts de Base de l'OO

Concept n°9 : La composition

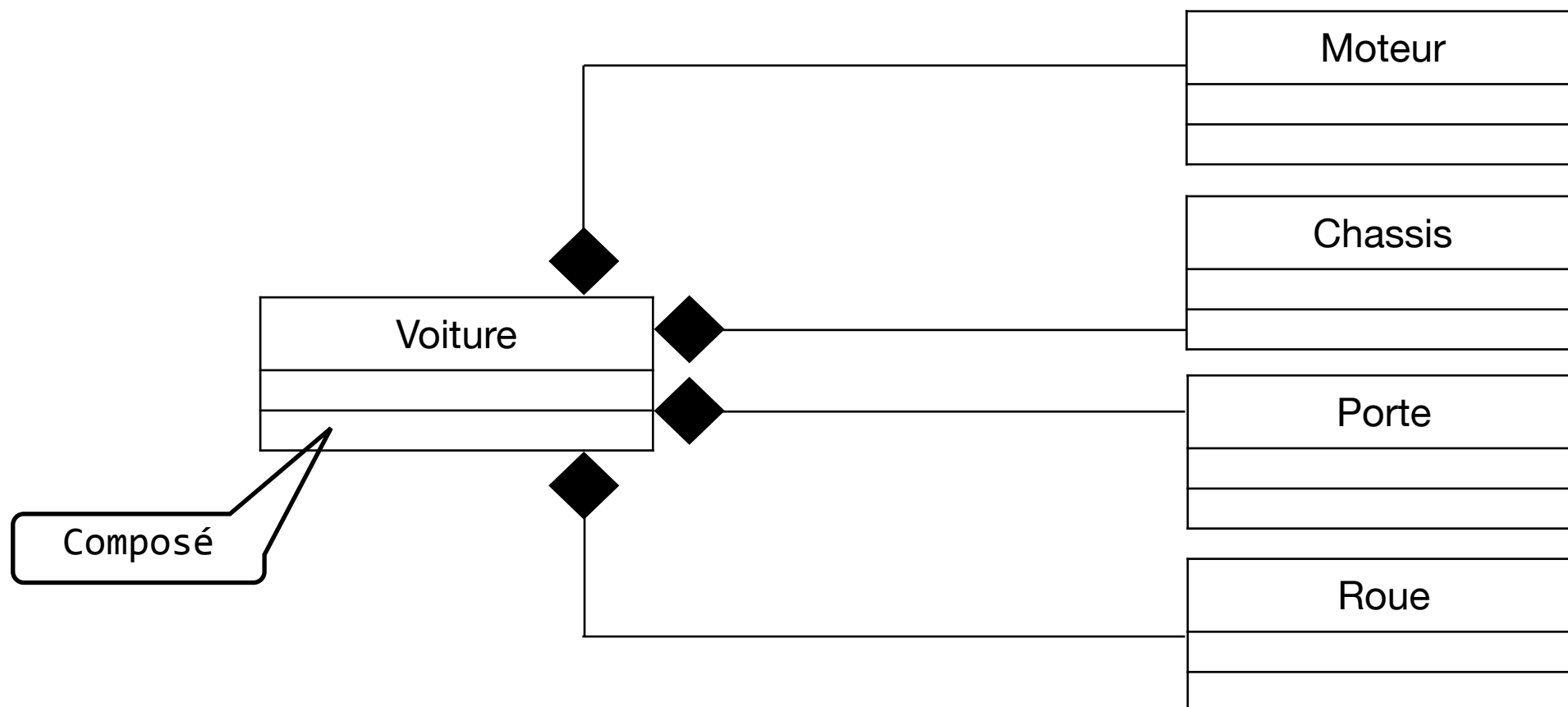
- Un objet peut être complexe et par conséquent une composition d'objets
- Un composé est formé à partir de composants



Concepts de Base de l'OO

Concept n°9 : La composition

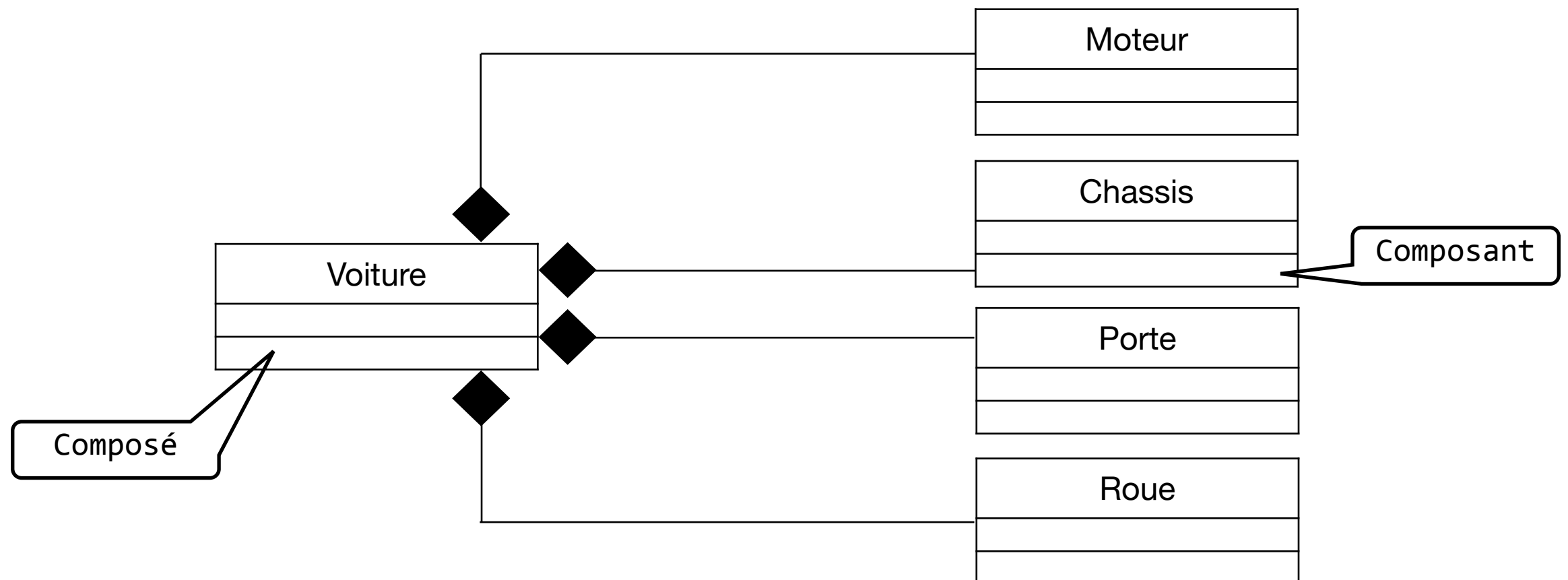
- Un objet peut être complexe et par conséquent une composition d'objets
- Un composé est formé à partir de composants



Concepts de Base de l'OO

Concept n°9 : La composition

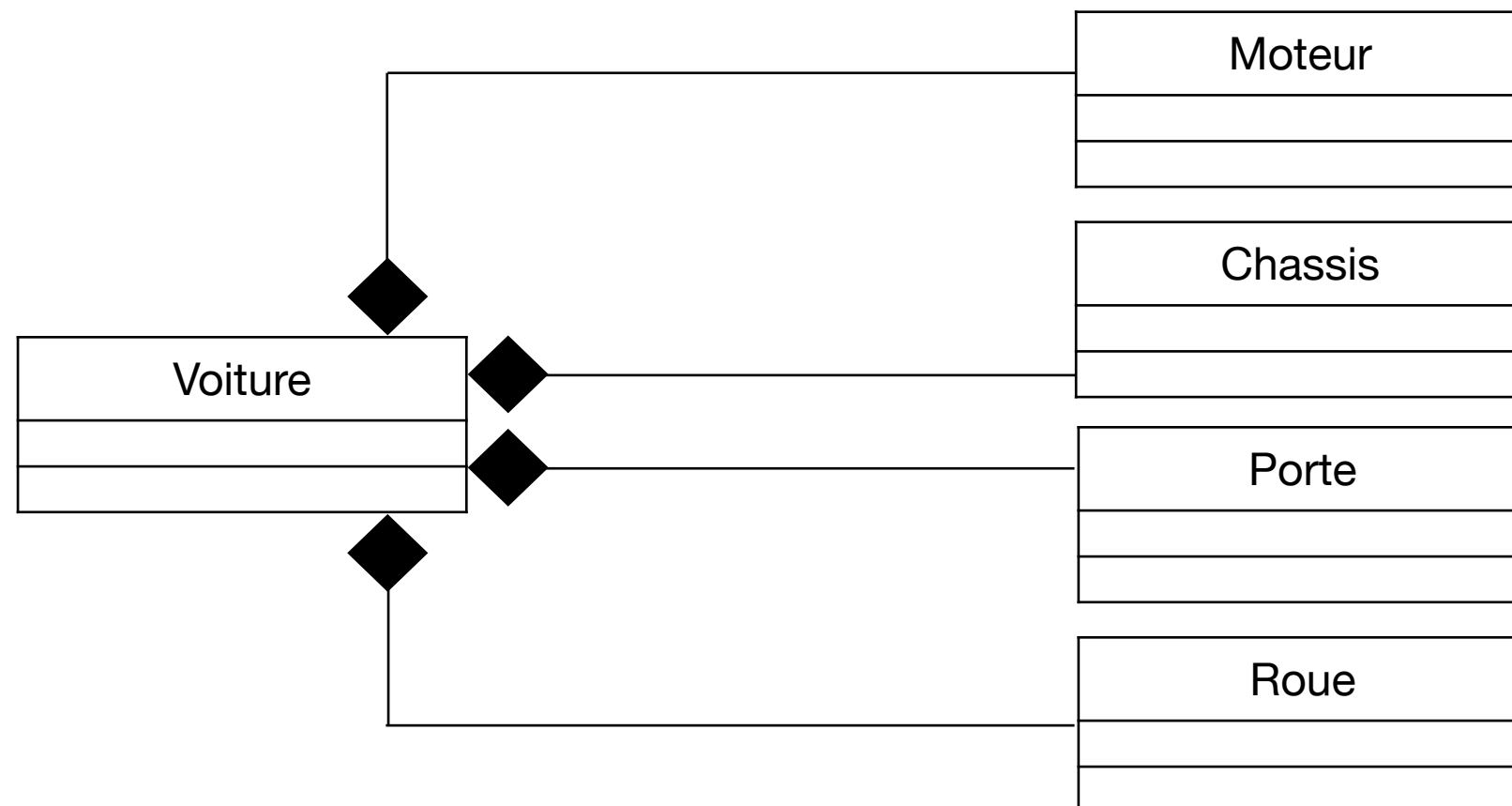
- Un objet peut être complexe et par conséquent une composition d'objets
- Un composé est formé à partir de composants



Concepts de Base de l'OO

Concept n°9 : La composition

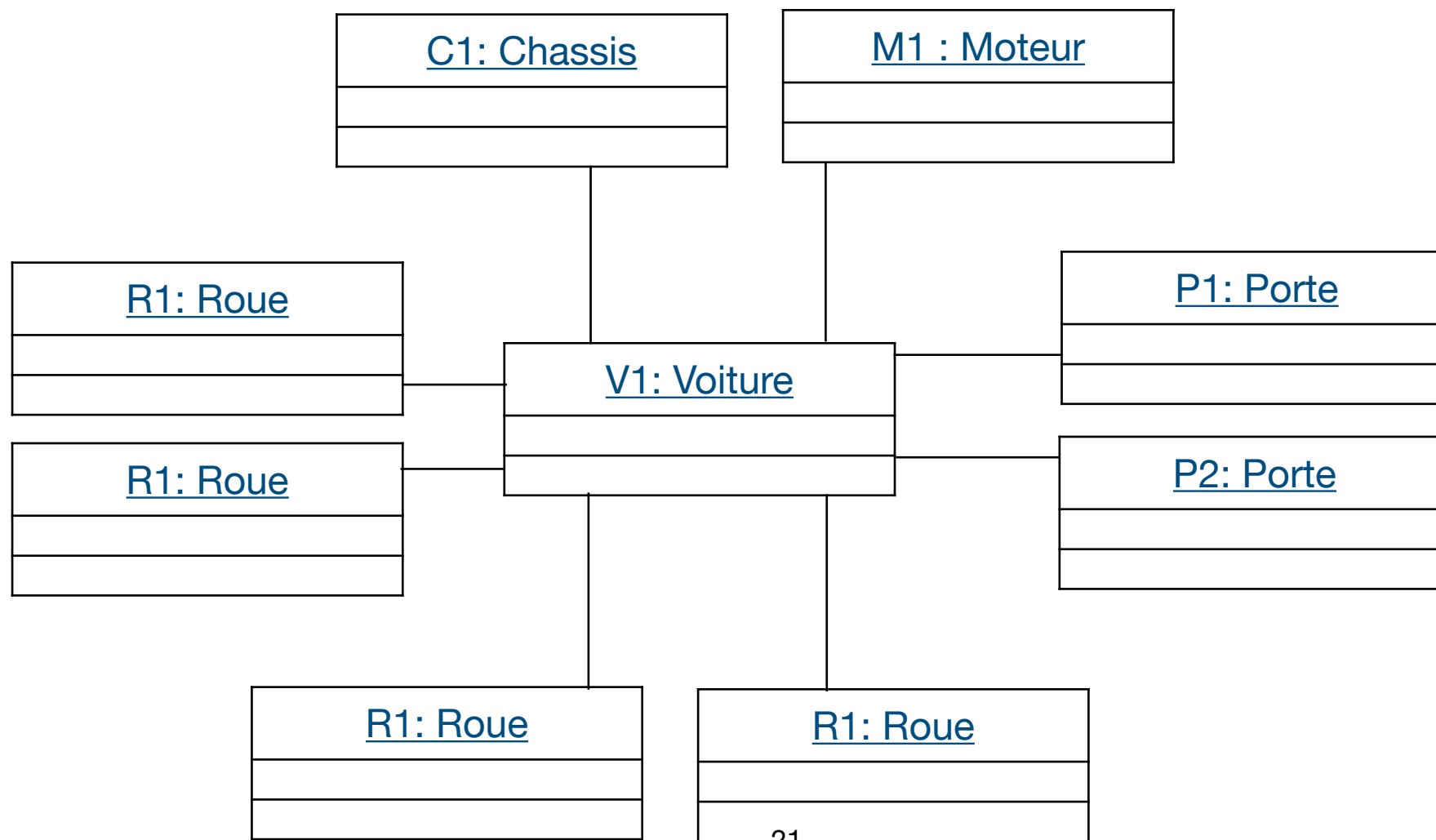
- **Composition forte** : les composants ne peuvent être partagés, la destruction du composé implique la destruction de ses composants



Concepts de Base de l'OO

Concept n°9 : La composition

- **Composition forte** : les composants ne peuvent être partagés, la destruction du composé implique la destruction de ses composants



Concepts de Base de l'OO

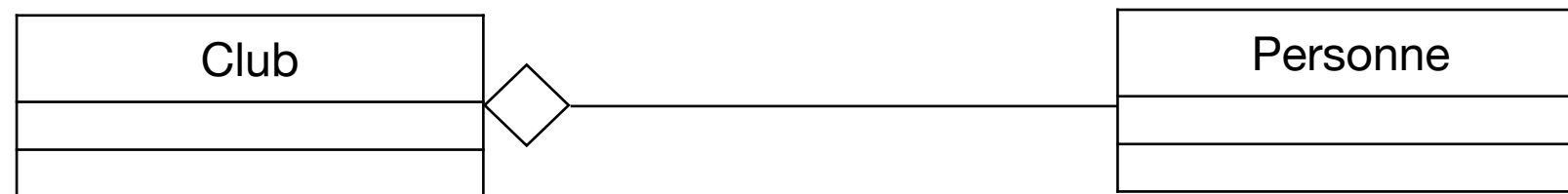
Concept n°9 : La composition

- **Composition faible (agrégation)** : les composants peuvent être partagés entre plusieurs objets complexes

Concepts de Base de l'OO

Concept n°9 : La composition

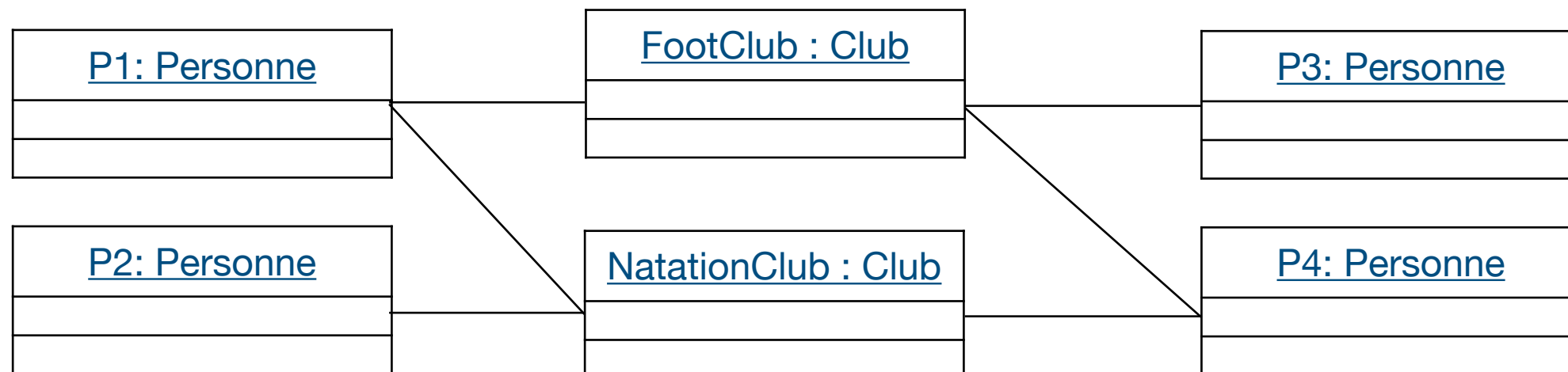
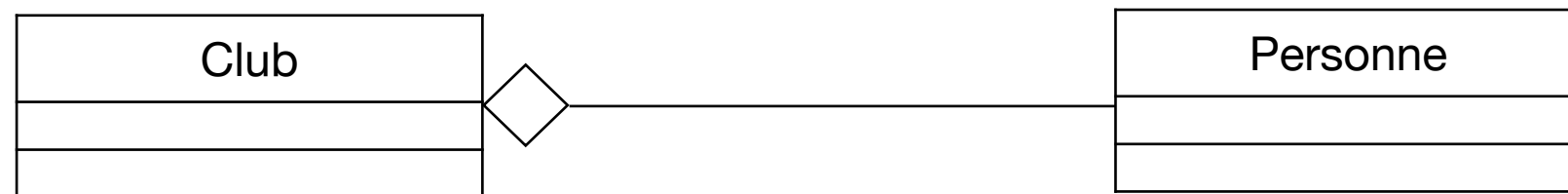
- **Composition faible (agrégation)** : les composants peuvent être partagés entre plusieurs objets complexes



Concepts de Base de l'OO

Concept n°9 : La composition

- **Composition faible (agrégation)** : les composants peuvent être partagés entre plusieurs objets complexes



Many

Thanks to

Noureddine Aribi II Sébastien Bardin II Nassim Belmecheri II
Kevin Chapuis II Pierre Gérard II Arnaud Gotlieb II Marianne Huchard II
Laurent Jean-Pierre II Alain Joubert II Yahia Lebbah II
Delphine Longuet II Samir Loudni II Sophie Nabitz II
Clémentine Nebut II Matthieu Rosenfeld II Alain Sabatier II
Helmut Simonis II Djamel Seriali II Christine Solnon II
Julie Vachon II Petru Valicov II Keunhyuk Yeom II Raphael Yende

Références

Books

- **UML Distilled (Third Edition): A Brief Guide to the Standard Object Modeling Language.** M Fowler 2004.
- **Object-Oriented Software Engineering (Second Edition): Practical Software Development Using UML and Java.** T. Lethbridge and R. Laganière 2005.
- **UML in Practice: The Art of Modeling Software Systems Demonstrated through Worked P.** Rogues 2004.
- **Requirements Engineering: From System Goals to UML Models to Software Specifications.** A. Lamsweerde 2009.
- **Software Engineering with UML.** B. Unhelkar 2018.