



Analyse Orientée Objet

Cours 2 : Diagramme de classes

L'approche orientée objet

- Modélisation du monde réel à l'aide d'un système formé d'objets
- Un objet est une représentation :
 - soit d'une réalité vivante comme un être humain, un animal, ..
 - soit d'une réalité matérielle comme une voiture, une bouteille, ...
 - soit d'une réalité immatérielle ou abstraite comme une idée, une dette, la sécurité sociale, ...

L'approche orientée objet

Cette modélisation est composée de 3 parties :

- l'**identité de l'objet** : ce qui permet de le distinguer d'un autre objet
- l'**état de l'objet** : ensemble des valeurs d'attributs
- le **comportement** : ensemble des méthodes qui décrivent ce qu'on peut faire avec.

Identité de l'objet

Porsche 911 : Voiture

Etat de l'objet

45789 : Numéro de série
911 : Modèle
1500 Kg : Poids
32 litres : Quantité d'essence

Comportement de l'objet

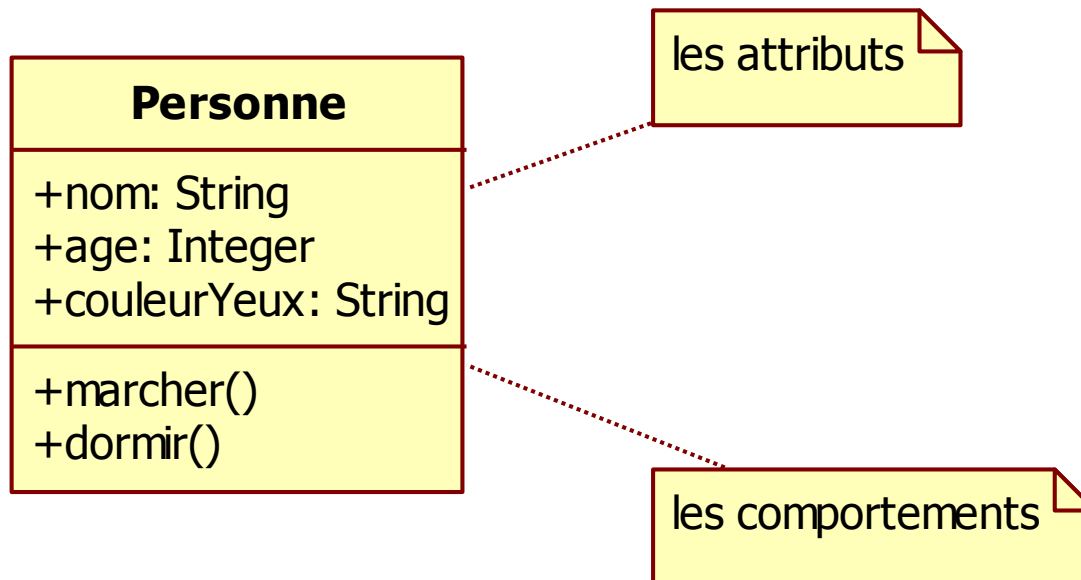
démarrer()
arrêter()
accélérer()
rouler()

Classe et Objet

- Une Ferrari peut se décrire de la même façon qu'une Porsche 911.
- Le nom et les valeurs des attributs permettent de différencier la Porsche 911 de la Ferrari
- On introduit un nouveau concept : **classe**.
- Une classe est une description commune d'objets.
- Un objet est une instance d'une classe. Seuls les objets auront une existence dans les programmes

UML et classes

- En UML, on décrit une classe comme l'indique le schéma qui suit :



Etats d'un objet

- Les valeurs des différents attributs d'un objet déterminent l'état de l'objet
- Un objet doit toujours être dans un état cohérent. Un état est cohérent quand il est prévu dans les spécifications.

Un objet Evaluation a un attribut **note**. Si une note doit être comprise entre 0 et 20 alors un objet Evaluation qui a son attribut **note** égal à -1 n'est pas dans un état cohérent.

Etat et comportement

- Le comportement dépend de l'état. Une opération ne s'exécutera pas nécessairement de la même façon pour deux états de l'objet.
- L'effet de l'opération **accélérer** dépend de la quantité d'essence dans la Ferrari.

Etat et comportement

- L'état d'un objet ne peut être changé que par le comportement.

C'est le concept d'**encapsulation**

L'exécution de l'opération **accélérer** diminue la quantité d'essence dans la Ferrari.

Diagramme de classes

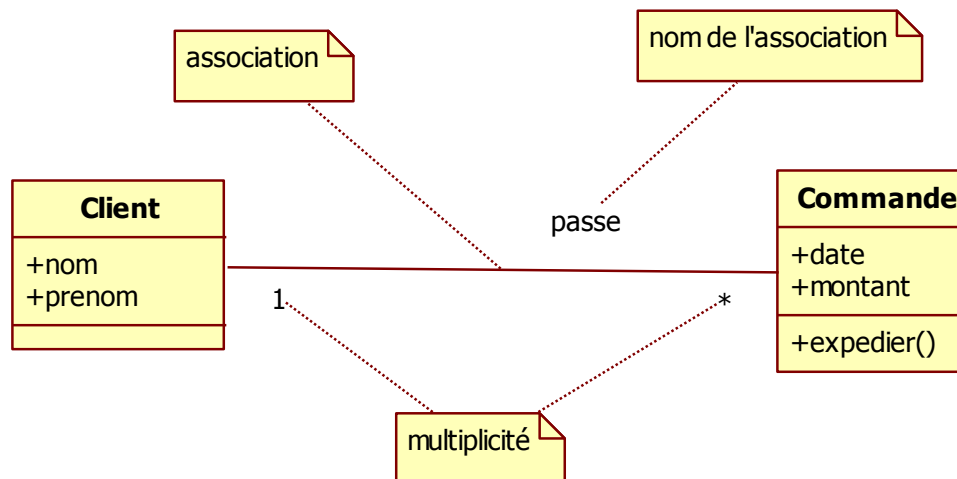
- Dans un système non trivial, on gèrera plusieurs objets d'une même classe et/ou de classes différentes .
- Il faut donc non seulement d'crire les objets avec des classes mais aussi les liens qu'il peut y avoir entre objets de diff'entes classes

Diagramme de classes

- Une collection d'éléments de modélisation statiques qui montre la structure d'un modèle.
- Classes : nom, attributs, opérations
- Relations entre classes (de la plus faible jusqu'à la plus forte) :
 - Association
 - Agrégation
 - Composition
 - Généralisation

Association

- L'association est le lien le plus pauvre entre deux classes.
- Il traduit le fait qu'un objet d'une classe est associé à un ou plusieurs autres objets d'une classe

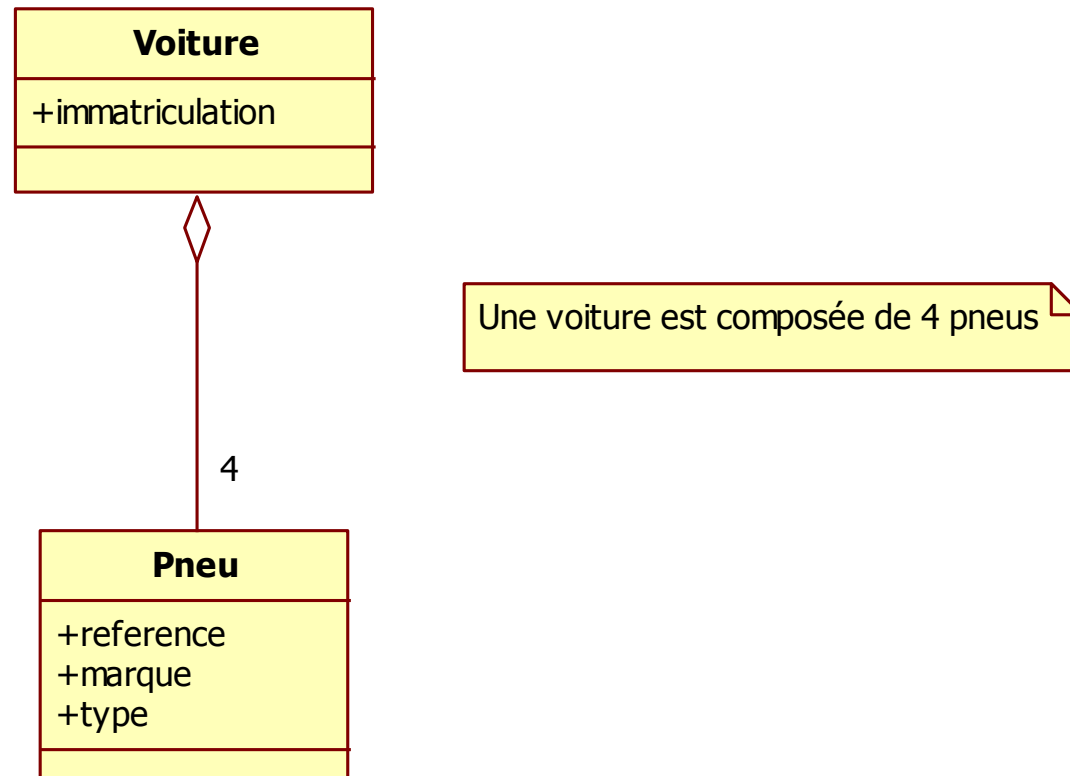


Association et Agrégation

- Une agrégation est une forme contrainte d'une association dans laquelle l'une des classes décrit un tout alors que la classe associée décrit des parties.
- On appelle le tout un composé (ou composite) et une partie un composant.

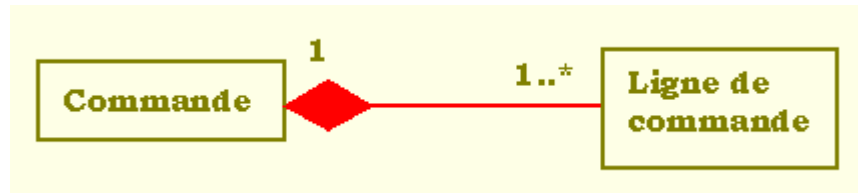
Association et Agrégation

- Une agrégation est une association dissymétrique

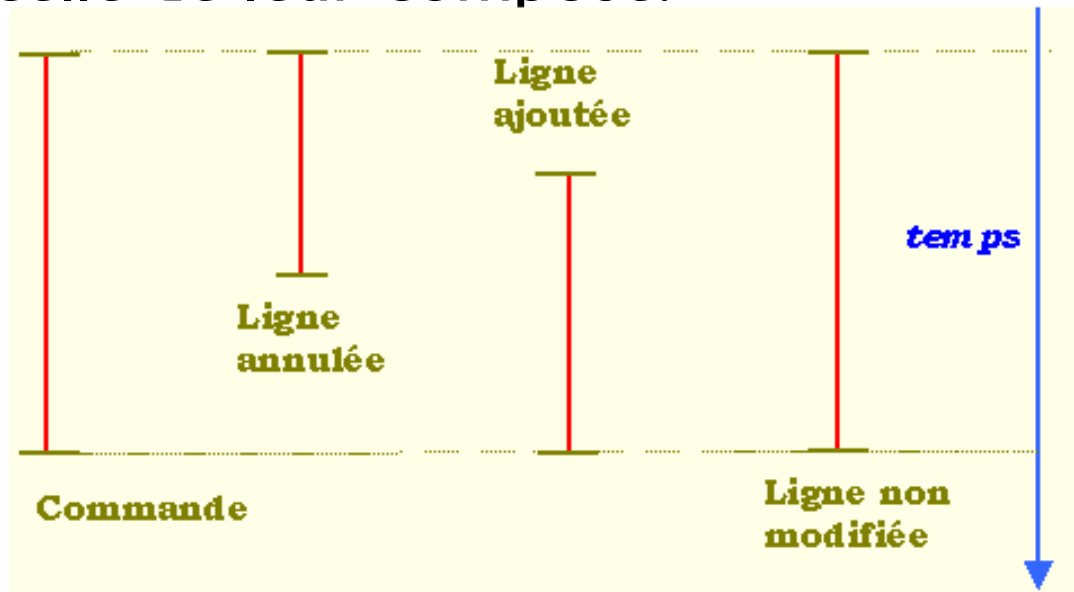


Agrégation et Composition

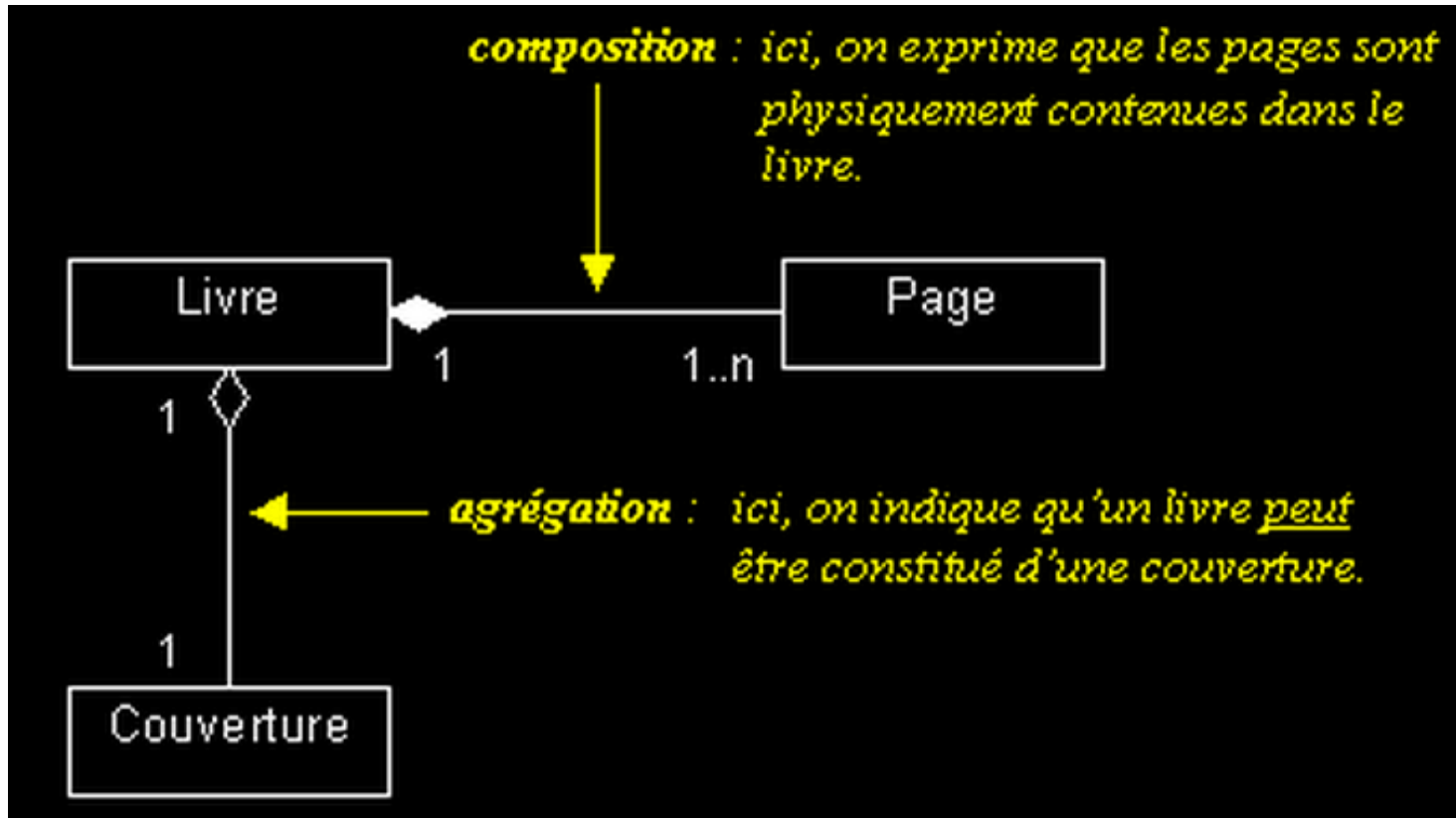
- Une composition est une forme contrainte d'une agrégation.



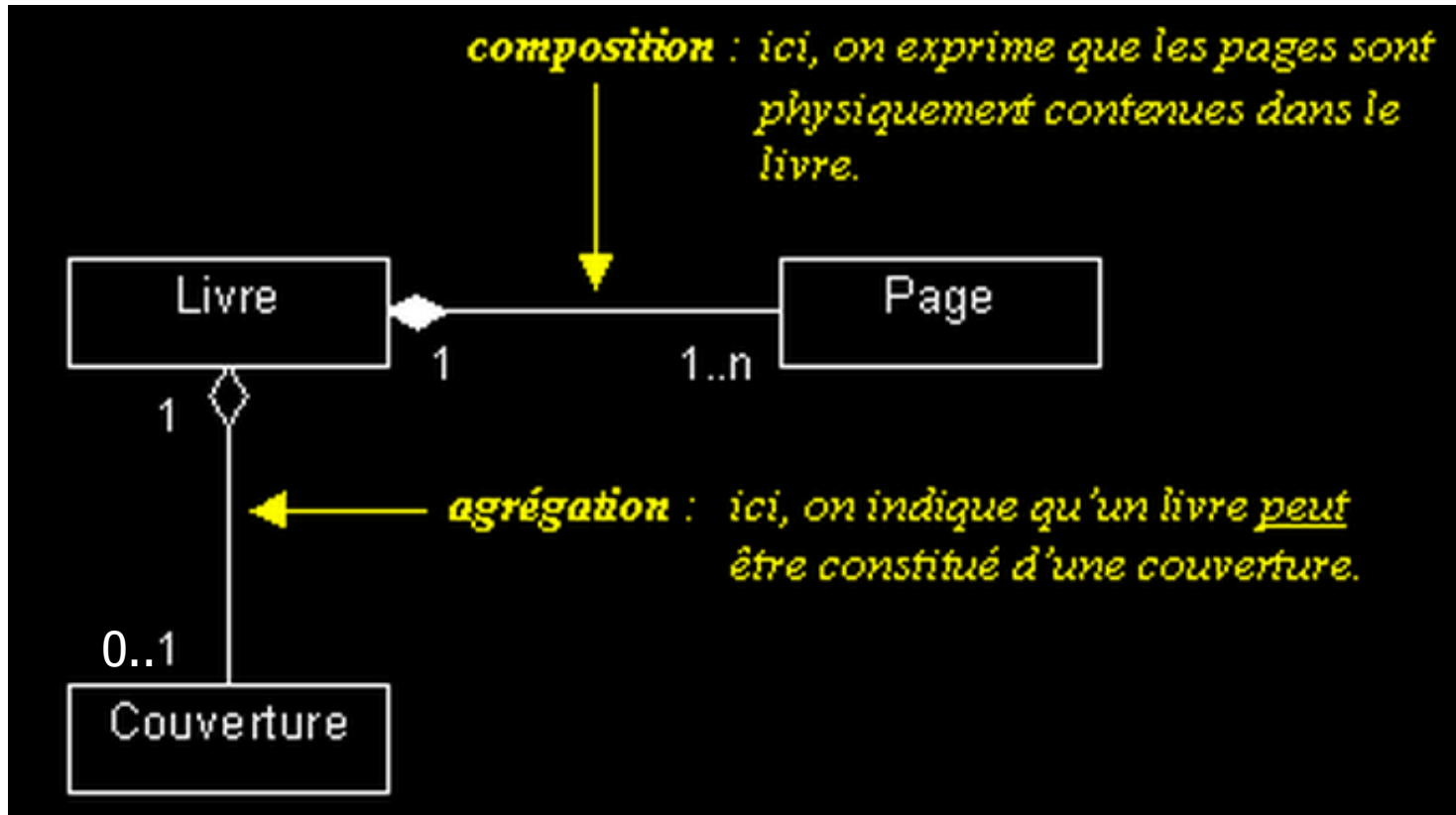
- Les composants ont une durée de vie incluse dans celle de leur composé.



Agrégation et Composition (uml.free.fr)



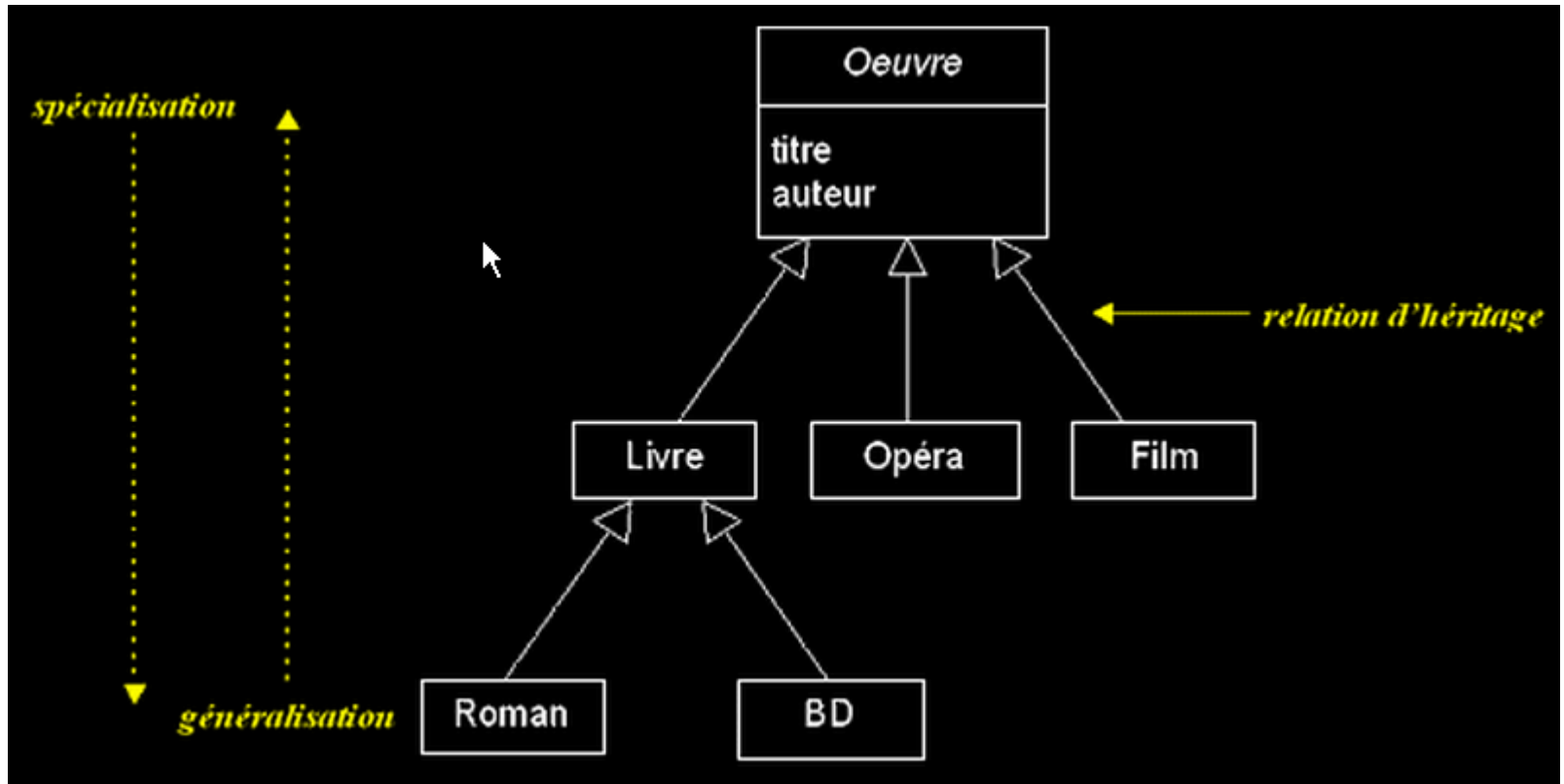
Agrégation et Composition (uml.free.fr)



Héritage : le concept

- Une généralisation est une relation conceptuelle entre classes, respectivement parent, enfant(s), telle que la classe enfant :
 - 1) possède les mêmes descriptions (attributs, opérations, associations) que sa classe parent
 - 2) ajoute des descriptions (attributs, opérations, associations) qui lui sont spécifiques.

Héritage : un exemple (uml.free.fr)



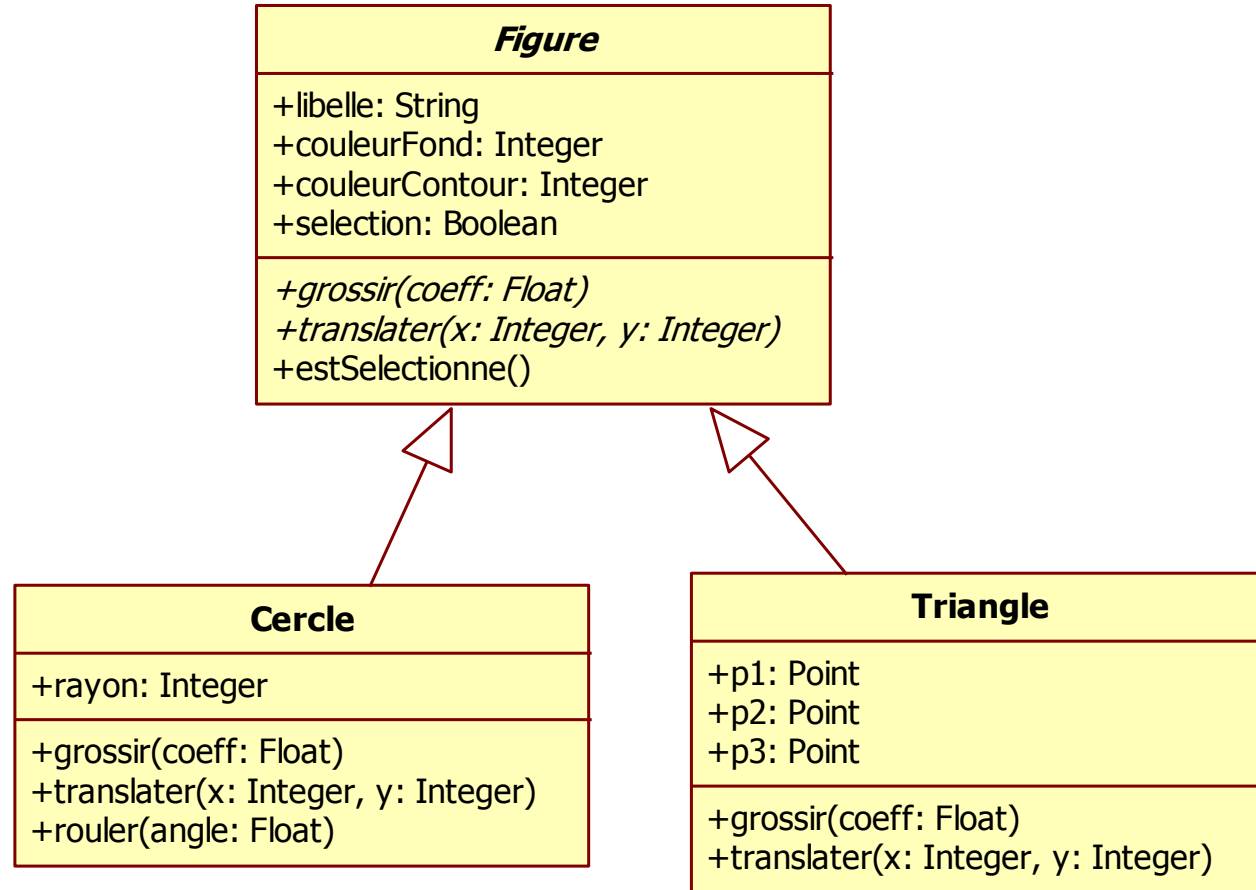
Héritage : remarques

- Dans une classe Enfant, certaines opérations de la classe Mère peuvent être redéfinies
- Un piège sémantique de l'expression :
 - Titi **est un** canari
 - Un canari **est un** oiseau
 - La première phrase est un classement. Titi est une instance de la classe **Canari**.
 - La deuxième phrase est une catégorisation. **Canari** est une sous classe de la classe **Oiseau**.

Héritage et Classes abstraites

- Certaines classes peuvent ne pas être entièrement définies pour donner lieu à des instances. On dit qu'elles sont **abstraites**.
- Elles ne seront utilisées qu'en tant que classes mères dans un héritage.
- Exemple : dans un logiciel de dessins, on fabrique des dessins. Un dessin est un ensemble de figures de formes différentes.

Héritage et Classes abstraites



Héritage et Classes abstraites

- La classe **Figure** est abstraite.
- Les instances seront des triangles ou des cercles mais pas de figures.
- Par contre, on constate que les deux classes Triangle et Cercle ont en commun une partie de leur comportement. C'est le concept de **polymorphisme**.

Les quatre concepts de l'objet

- Etat + comportement
- Encapsulation
- Héritage
- Polymorphisme

Etude de cas : Projet EISTI ☺

- Chaque projet a un nom et a lieu dans un semestre d'une année scolaire donnée.
- Un projet est divisé en plusieurs livrables différents, avec un numéro, une description et une. Il y a au moins 3 livrables par projet
- Les étudiants forment des groupes de projet de taille 4 ou 5. Il faut déposer le travail effectué pour chaque livrable à une date qui ne dépasse pas la deadline !
- Les travaux sont notés par groupe et par livrable.
- Questions :
 - Classes ?
 - Associations ?



Analyse Orientée Objet

Cours 2 : Diagramme de classes (suite)

Classe : la visibilité des attributs

- Un attribut d'une classe peut être
 - **private** (-) : accessible que dans les opérations de la classe
 - **public** (+) : accessible dans toutes les opérations
 - **protected** (#) : accessible que dans les opérations de la classe et les classes dérivées
 - **package** (~) : accessible que dans les opérations des classes dans le même paquetage.

Example

Employee
-id : int -name : string -telephone : string -email : string = "no email" #password : string

Classe : l'exécution des opérations

- Une opération d'une classe peut être
 - **private** (-) : *appelable* que par les opérations de la classe
 - **public** (+) : *appelable* par toutes les opérations
 - **protected** (#) : *appelable* que par les opérations de la classe et les classes dérivées
 - **package** (~) : *appelable* que par les opérations des classes dans le même paquetage.

Example

Employee
-ID[1] : int -Name[1] : string -Telephone[1] : string -Email[1..5] : string = "none" #Password[1] : string
+getID() : int +getName() : string +getTelephone() : string +getEmail(in index : int) : string +setTelephone(in phone : string) #setPassword(in pwd : string)

Classe : les attributs dérivés

- Un attribut peut-être constitué à partir d'autres attributs. On utilisera le symbole /
- Exemples :
 - longueur
 - largeur
 - /surface

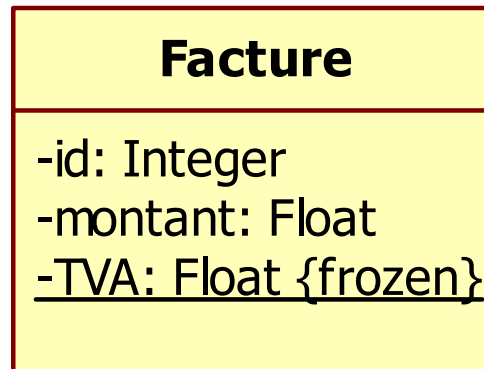
Account
-currentBalance : float
-/isNegative : bool

isNegative == true
iff
currentBalance < 0

Les attributs de classe

- Un **attribut de classe** est un attribut qui est commun à toutes les instances de la classe
- Graphiquement, un attribut de classe est souligné (équivalent à **static** en Java ou C++)
- Exemple : le taux de TVA dans la classe Facture

—



Les opérations de classe

- Une opération de classe est une opération qui est appellable sans passer par une instance
- Graphiquement, la méthode de classe est soulignée
- Exemples :
 - Les opérations qui fabriquent de nouveaux objets
 - Les opérations qui modifient les attributs de classes

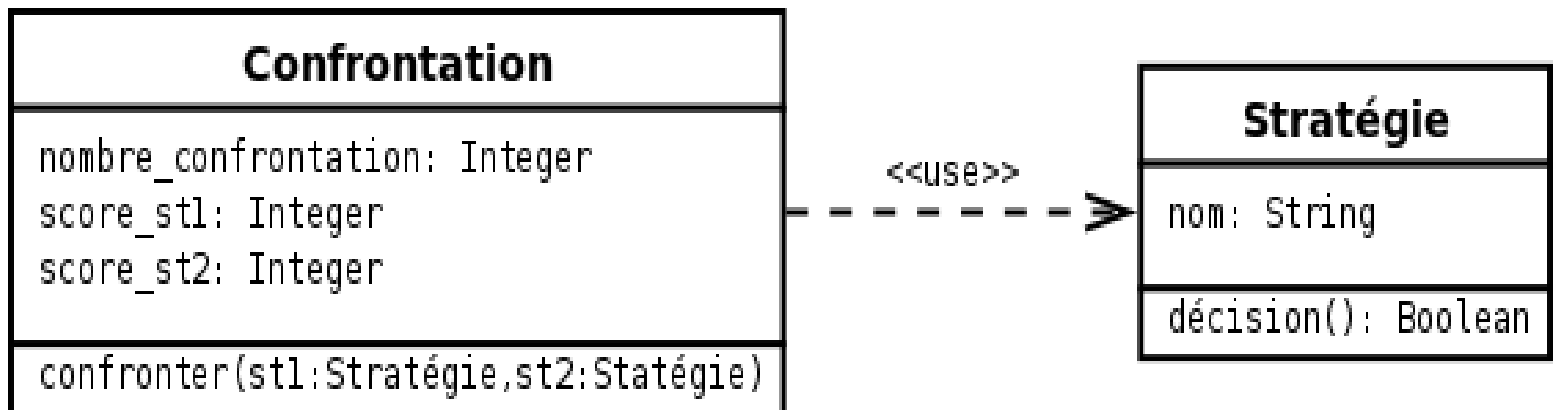
Facture
-id: Integer -montant: Float <u>-TVA: Float {frozen}</u>
<u>+creerFacture(id: Integer, montant: Float): Facture</u> <u>+getTVA(): Float</u> +getId(): Integer +getMontant(): Float

Relations entre classes

- De la plus faible jusqu'à la plus forte :
 - Dépendance
 - **Association**
 - **Agrégation**
 - **Composition**
 - **Généralisation**

Les relations de dépendance

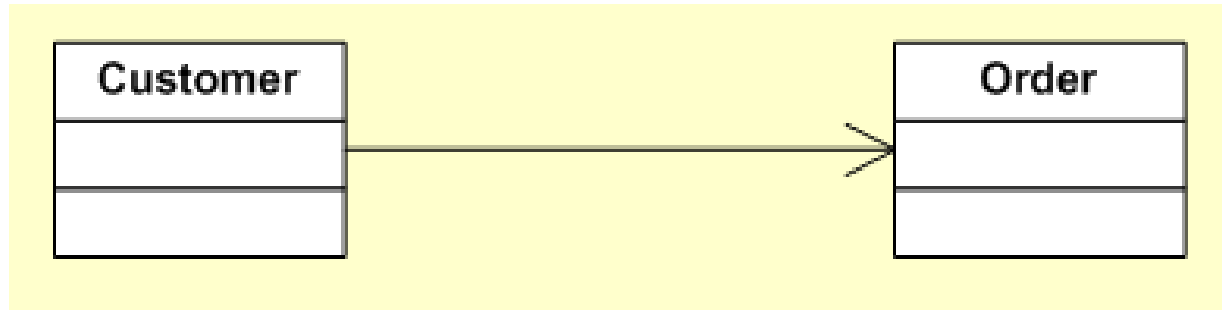
- Une dépendance (**dependency**) est une relation faible entre classes
- Classe A **dépend** de la classe B si A **utilise** B dans une de ces opérations



Associations entre classes

- Une connexion sémantique entre deux classes
- Caractéristiques :
 - **navigabilité** : uni ou bidirectionnelle
 - **cardinalités** : précise le nombre d'instances qui participent à une relation.
 - **rôles** : spécifie la fonction d'une classe pour une association donnée.
 - **qualification** : restreint la portée de l'association à quelques éléments ciblés de la classe.
 - **classe d'association**

Association : navigation



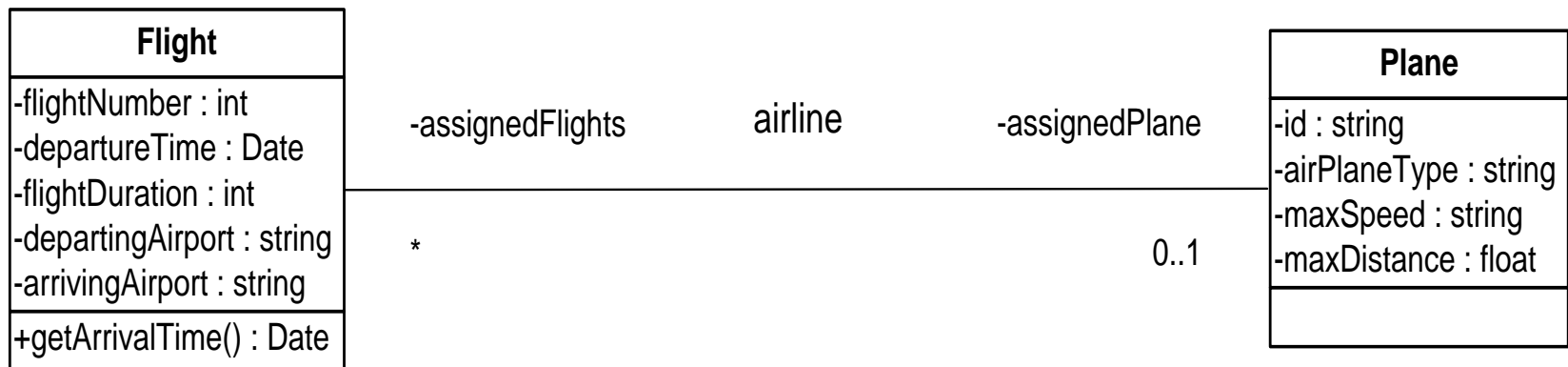
- La **flèche** indique la possibilité de **naviguer** de A à B
- La **navigabilité** de la classe A à la class B indique qu'une instance de la classe A peut accéder aux instances associées de la classe B.
- Quand l'association peut naviguer dans les deux directions, alors aucune flèche n'est dessinée.

Les rôles d'association



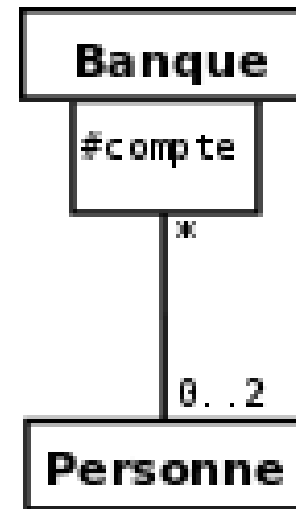
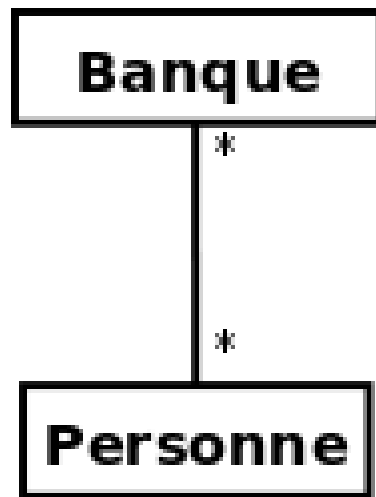
- employé et employeur sont les rôles (terminaisons) de l'association travailler pour

D'autres exemples

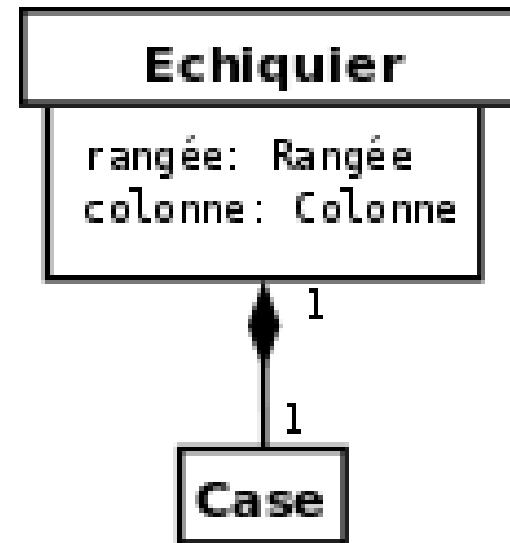
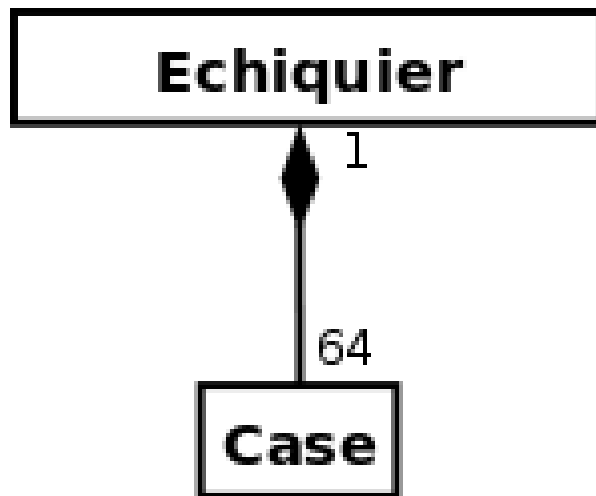


Qualification (association qualifiée)

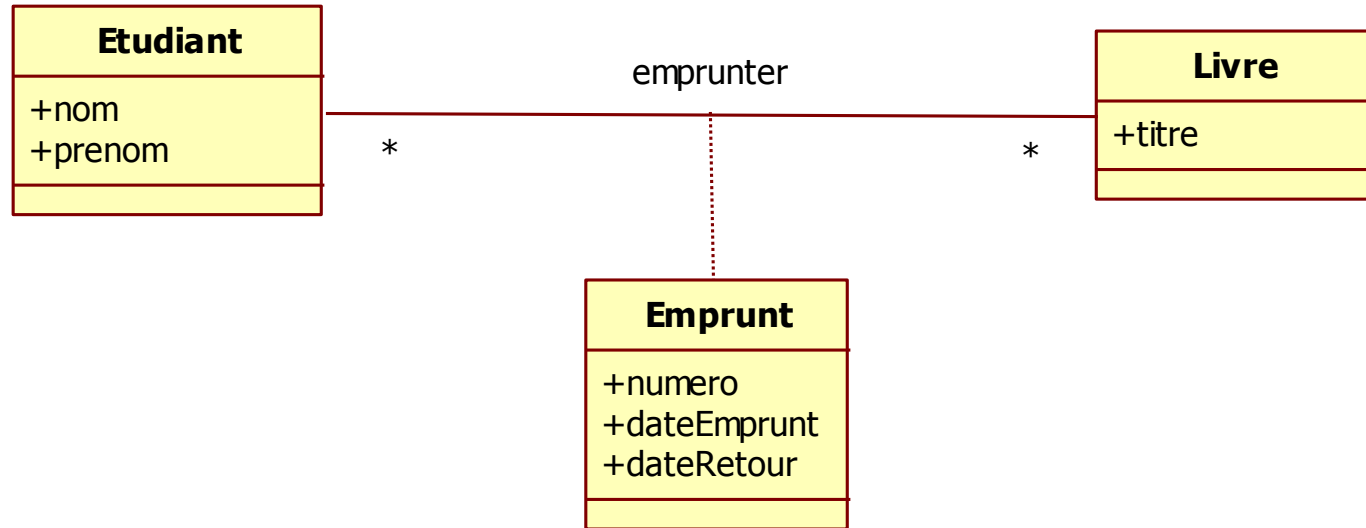
- Restreindre la portée de l'association à quelques attributs ciblés de la classe.



Un autre exemple de qualification

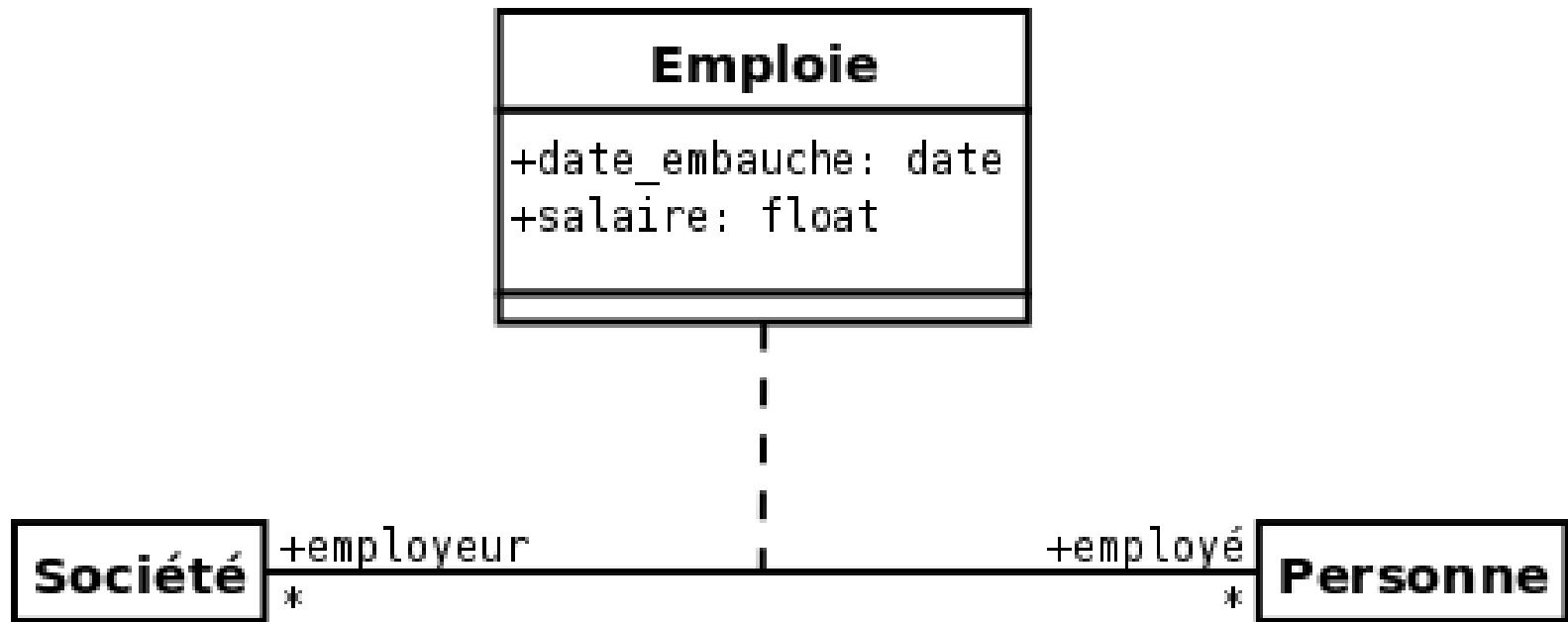


Les classes d'association

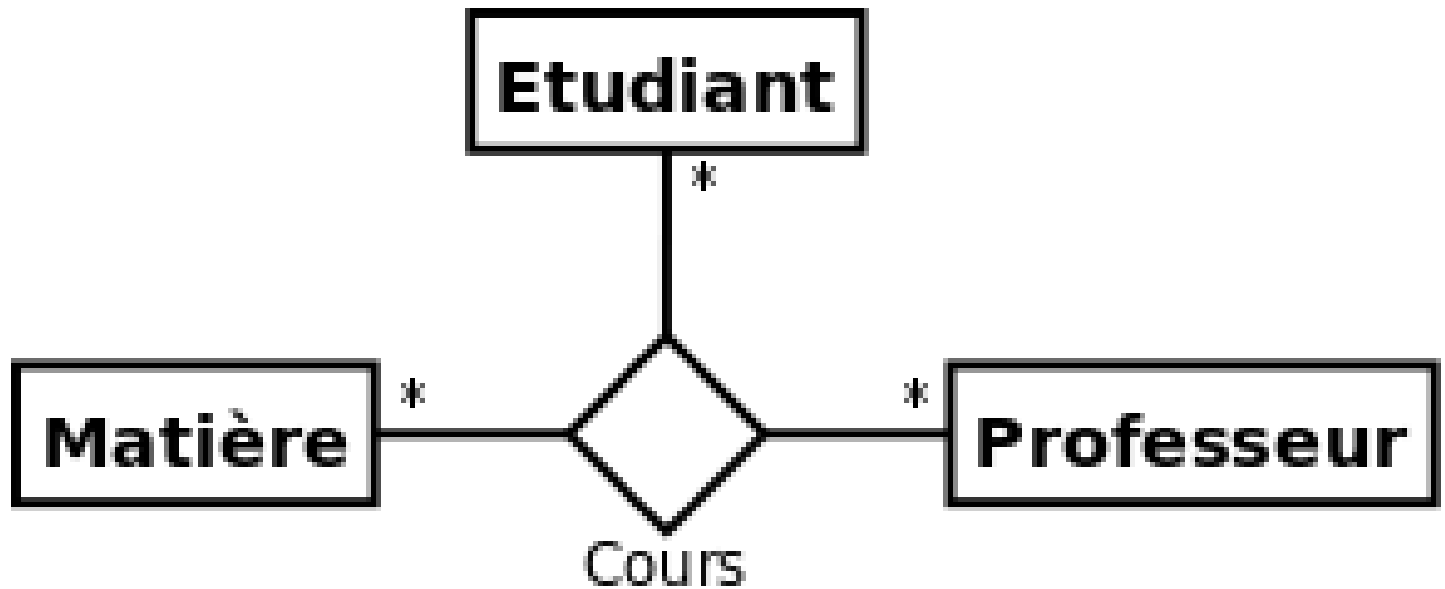


- Dans certaines associations, à chaque instance du couple, on associe un objet d'une autre classe. On dit que l'association est porteuse d'informations
- Exemple : Dans l'association qui relie la classe **Etudiant** à la classe **Livre**, on définit la classe d'association **Emprunt**

Un autre exemple de classe d'association



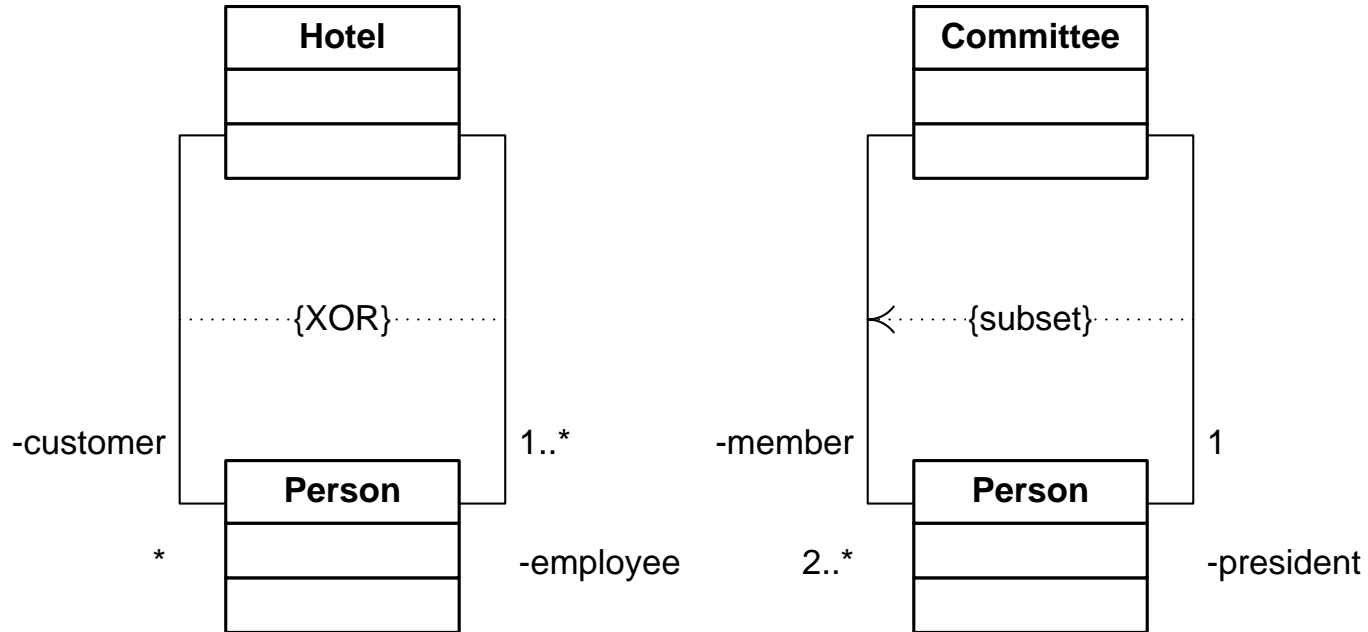
Association n-aire



Contraintes d'association

- {XOR}
- {subsets <property_name>}
- {redefines <property_name>}
- {union}
- {ordered}
- {bag}
- {sequence} ou {seq}

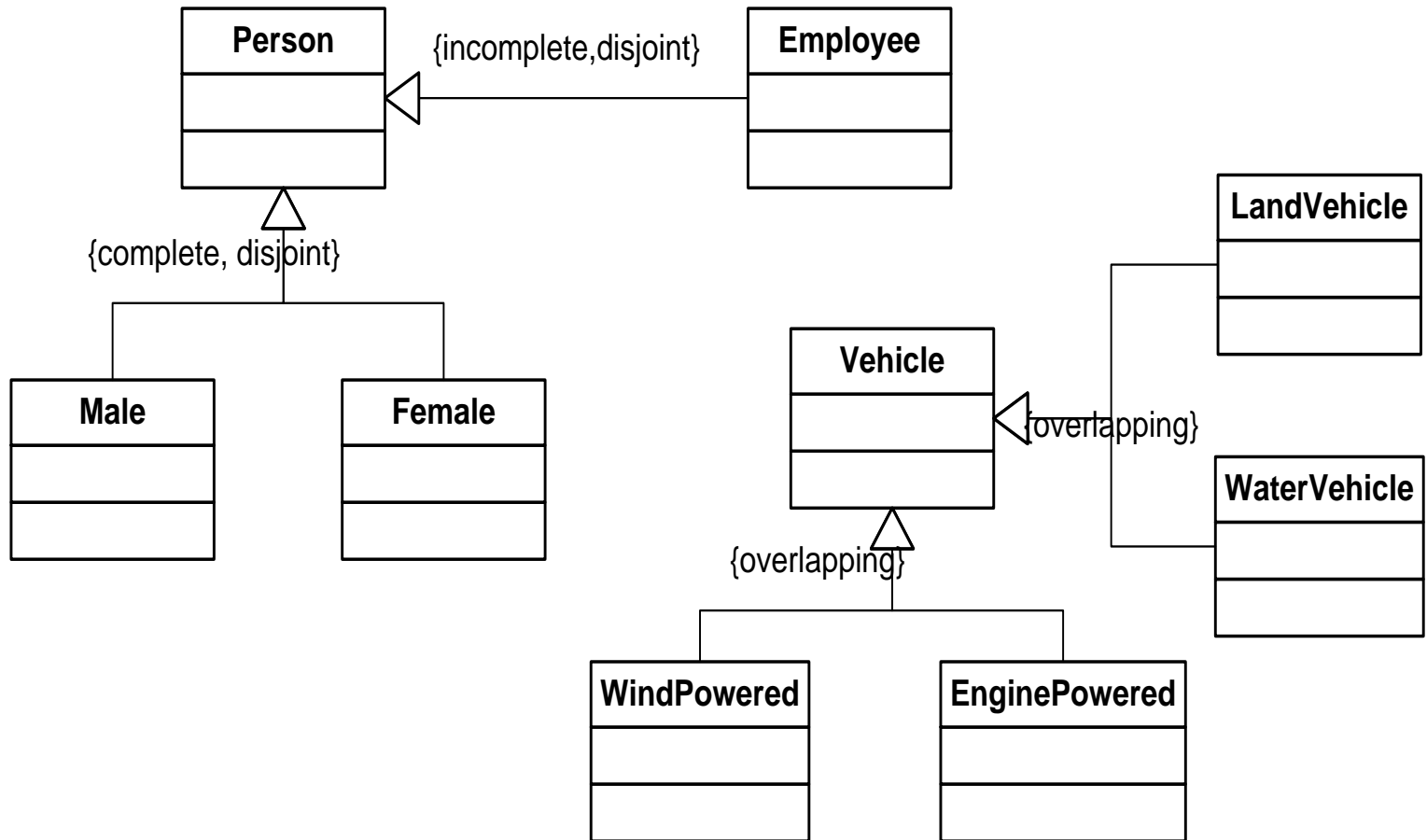
Contraintes d'association : exemples



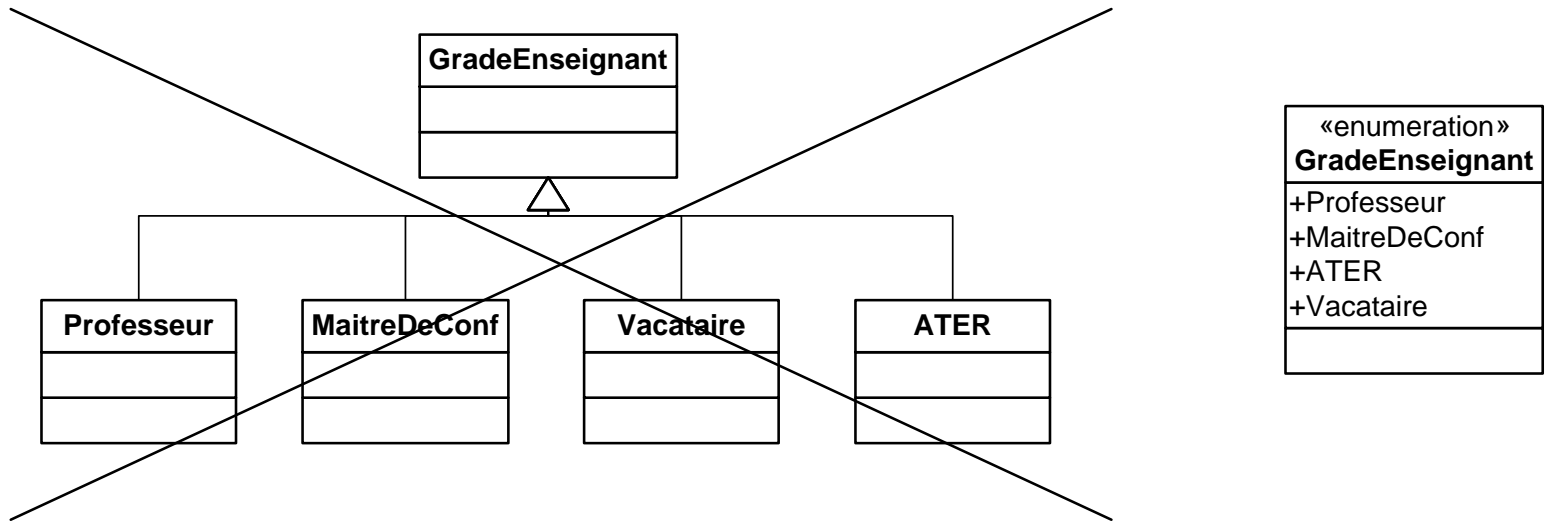
Contraintes de généralisation

- {complete, disjoint}
 - pas extensible, pas d'instance commune
- {incomplete, disjoint}
 - extensible, pas d'instance commune
- {complete, overlapping}
 - pas extensible, avec des instances communes
- {incomplete, overlapping}
 - extensible, avec des instances communes
- Par défaut : {incomplete, disjoint}

Contraintes de généralisation : exemples



Enumérations



- Enumérations sont des classes représentant des objets qui peuvent avoir un nombre fini des valeurs

Interface

- **Encapsulation** : un objet n'est accessible de l'extérieur qu'à travers ses opérations.
- La déclaration d'une opération d'un objet est constituée :
 - du nom de l'opération
 - ses paramètres
 - du retour

Cet ensemble est la **signature de l'opération**.

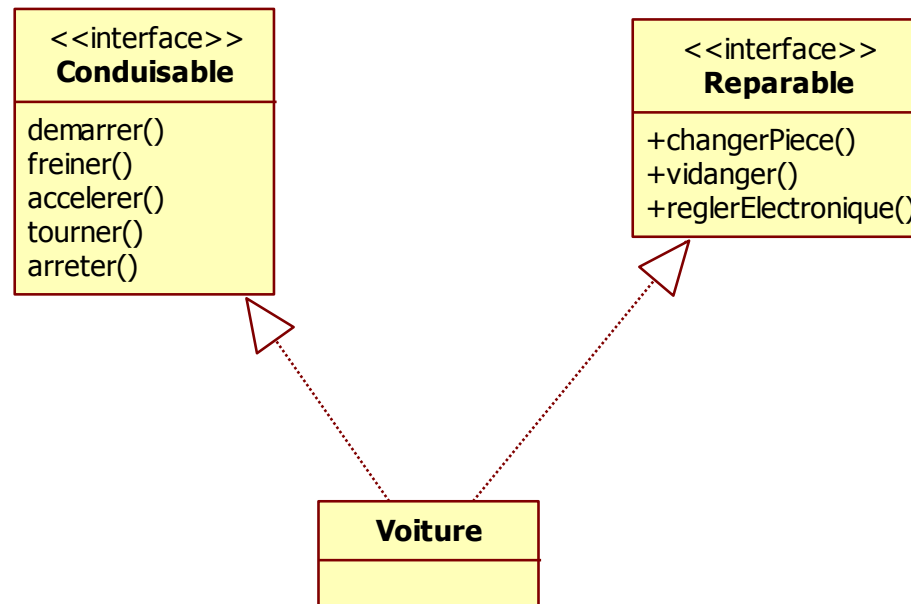
- L'ensemble des signatures des **opérations publiques** d'un objet est appelée **l'interface de l'objet**.

Interface

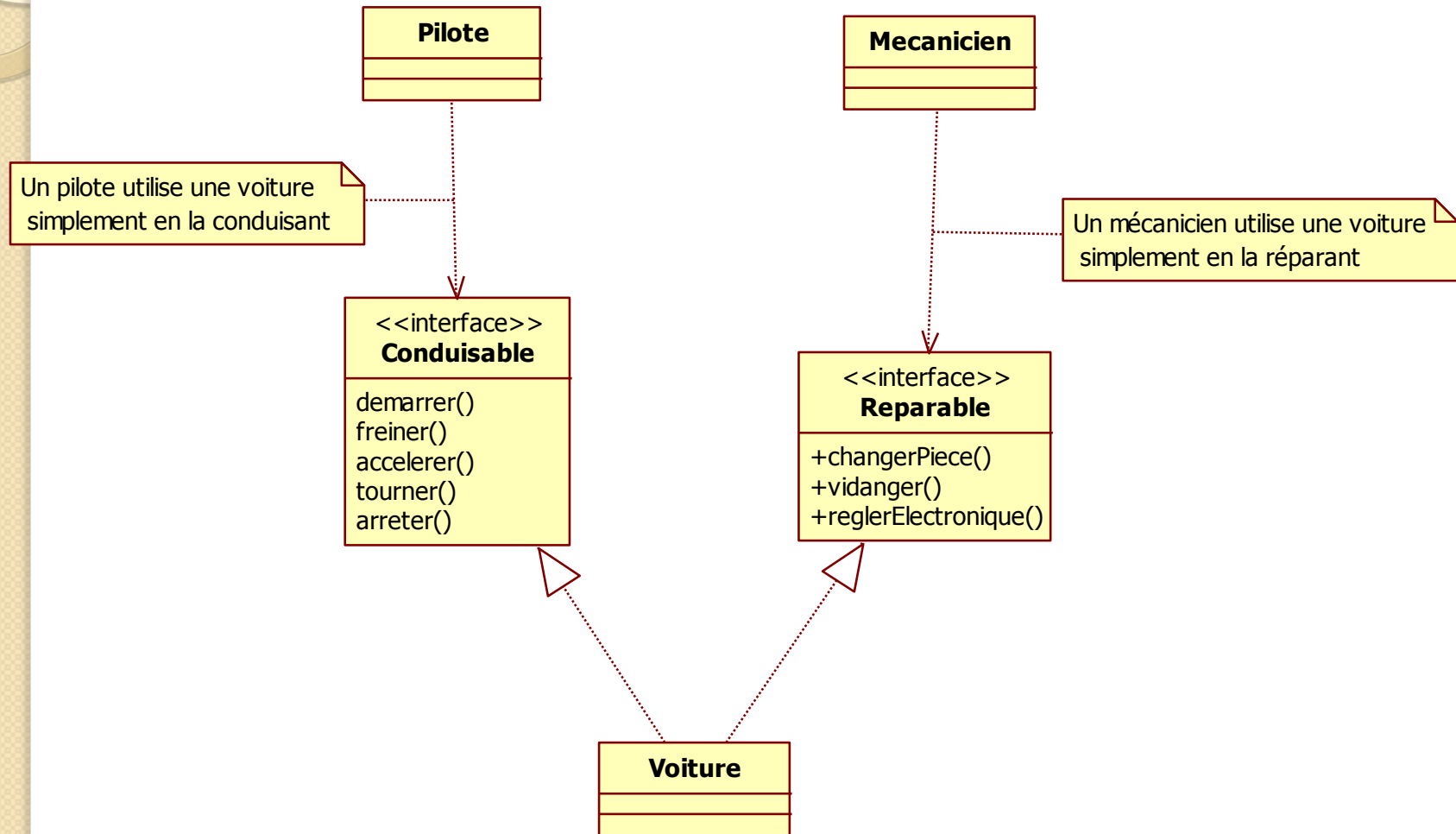
- L'interface est la vue externe d'un objet, elle définit les services accessibles (offerts) aux utilisateurs de l'objet.
 - Masquer les détails d'implémentation d'un objet (**encapsulation**)
 - Garantir l'intégrité des données contenues dans l'objet.

Interface : exemple

- Voiture à :
 - **conduire** : démarrer, accélérer, freiner, tourner, arrêter
 - **entretenir** : changer de pièce, vidanger, régler l'électronique



Interface : exemple



Etude de cas : Projet EISTI ☺

- Chaque projet a un nom et a lieu dans un semestre d'une année scolaire donnée.
- Un projet est divisé en plusieurs livrables différents, avec un numéro, une description et une. Il y a au moins 3 livrables par projet.
- Les étudiants forment des groupes de projet de taille 4 ou 5. Il faut déposer le travail effectué pour chaque livrable à une date qui ne dépasse pas la deadline !
- Les travaux sont notés par groupe et par livrable.
- Proposez le diagramme de classes pour ce problème, avec :
 - Les classes : attributs, méthodes
 - Les associations : cardinalité, rôles, ...
 - Classes d'association ?
 - Agrégations, compositions, généralisations ?

