

Plan


- 
- Diagrammes fonctionnels
 - *Diagrammes statiques*
 - *De classes*
 - *D'objets*
 - *De composants*
 - *De structure composite*
 - Diagrammes dynamiques
 - Diagrammes d'implémentation

Diagramme de classes


- 
- Définition des éléments formant une application et de leurs relations
 - Structuration statique de l'application
 - Définition des classes existantes
 - Définition de la structure interne des classes (attributs, opérations)
 - Définition des relations entre les classes
 - 2 principaux types de relations entre classes
 - **Association**
 - Un client peut louer un certain nombre de vidéos
 - **Sous-typage/généralisation**
 - Un étudiant est une personne
 - Important : documenter les diagrammes de classes

Diagramme de classes

- Exemple de 3 usages possibles d'un diagramme de classe
 - **Diagramme conceptuel**
 - Concepts métier du domaine étudié à un niveau abstrait
 - Sans lien avec l'implémentation
 - **Diagramme de spécification**
 - Première approche du logiciel par la définition de ses interfaces
 - Interface = type de l'objet, définit les interactions
 - Classe = implémentation de l'objet
 - Un type (ou interface) peut avoir plusieurs réalisations (liées à l'environnement, choix de conception/implémentation...)
 - **Diagramme d'implémentation**
 - Vision « bas-niveau » de l'implémentation du logiciel

Diagramme de classes

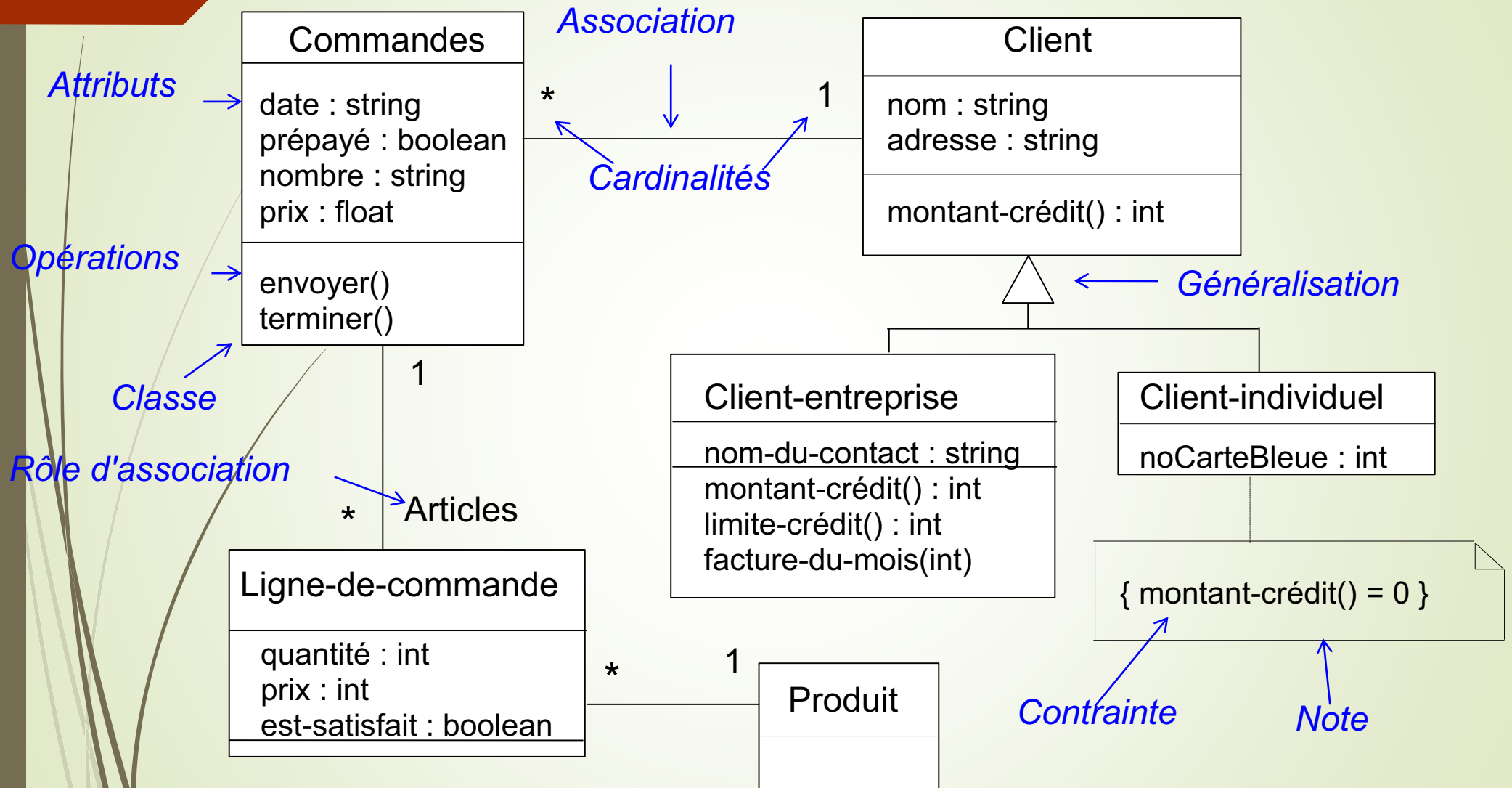


Diagramme de classes

➤ Attributs

➤ Élément caractérisant une partie de l'état d'un objet

➤ Syntaxe UML pour la définition d'un attribut :

visibilité nom [multiplicité] : type = init {propriétés}

➤ visibilité : + (public), # (protégé) ou – (privé (par défaut))

➤ nom : nom de l'attribut

➤ multiplicité : nombre d'attributs de ce type (tableau : [1..5])

➤ type : type de l'attribut

➤ init : valeur initiale de l'attribut

➤ propriétés : propriétés, contraintes associées à l'attribut

Diagramme de classes

➤ Opérations

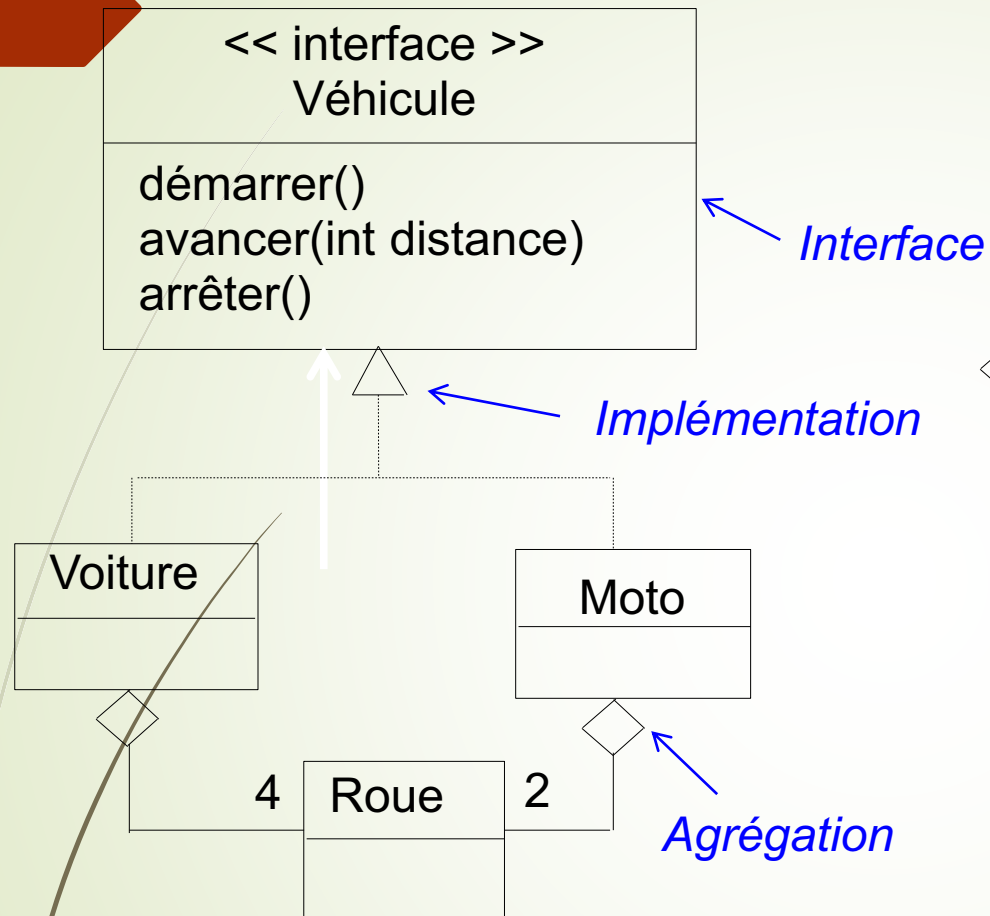
- Processus/fonction qu'une classe sait exécuter
- Appelées également méthodes dans les langages objets

➤ Syntaxe UML pour la définition d'une opération :

visibilité nom(paramètres) : typeRetourné {propriétés}

- visibilité : + (public), # (protégé) ou – (privé)
- nom : nom de l'opération
- paramètres : liste des paramètres de l'opération
- typeRetourné : type de la valeur retournée par l'opération (si elle retourne une valeur)
- propriétés : propriétés, contraintes associées à l'opération

Diagramme de classes



Agrégation : les éléments existent toujours quand l'association est détruite

Composition : les éléments disparaissent quand l'association est détruite

➡ Variante avec classe abstraite

➡ Nom en italique

➡ Ne peut être instanciée

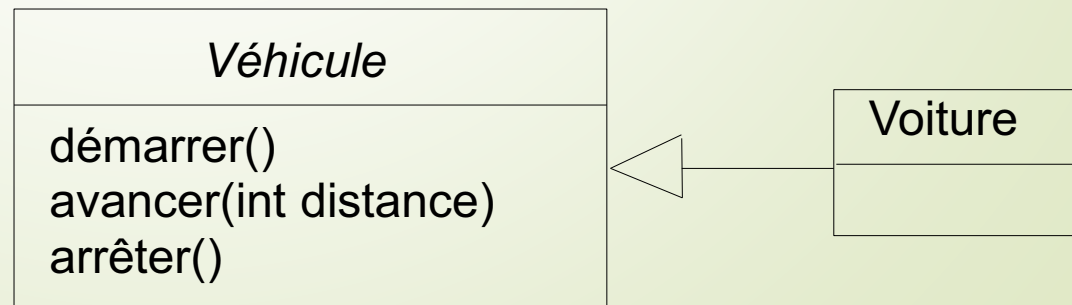


Diagramme de classes

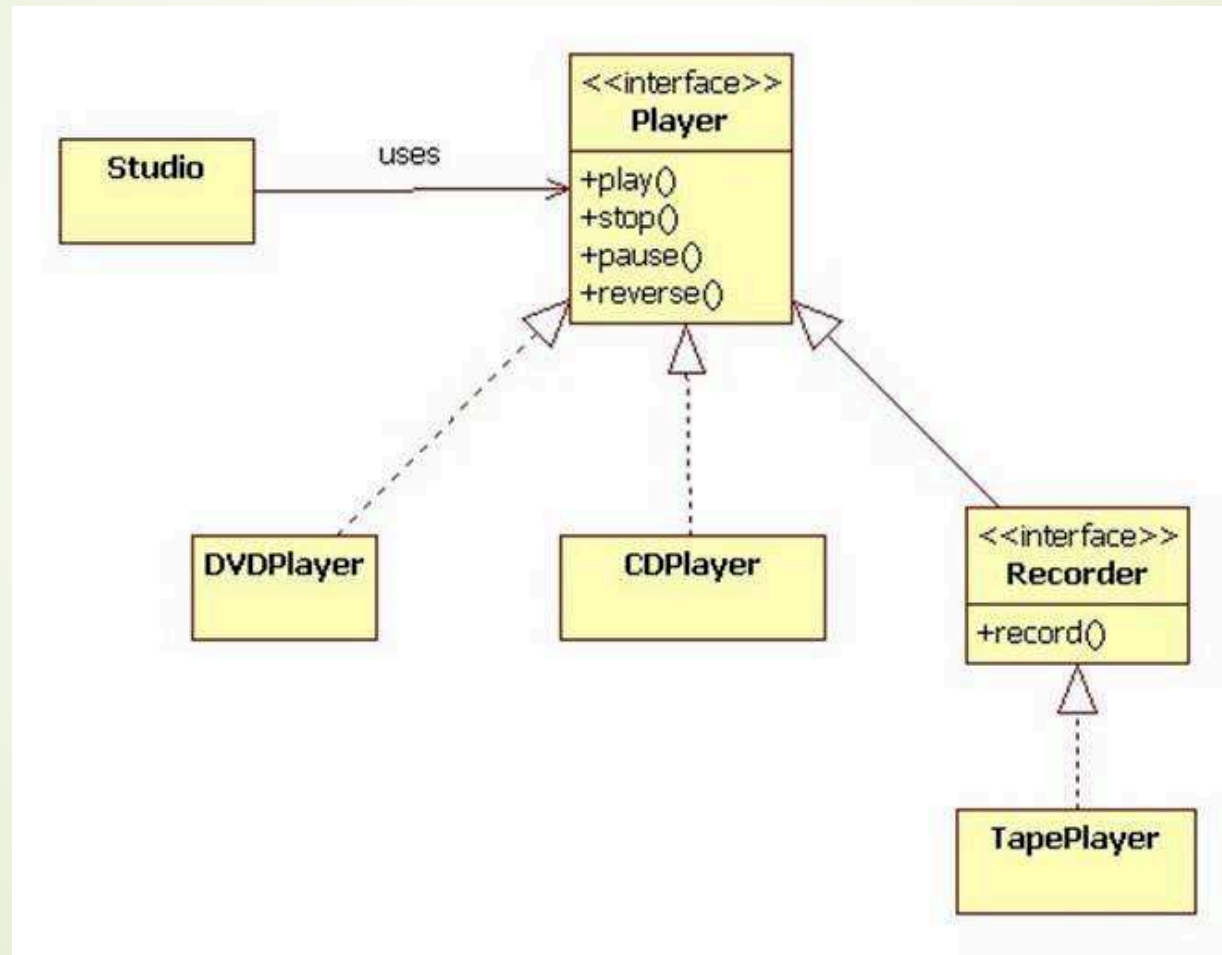
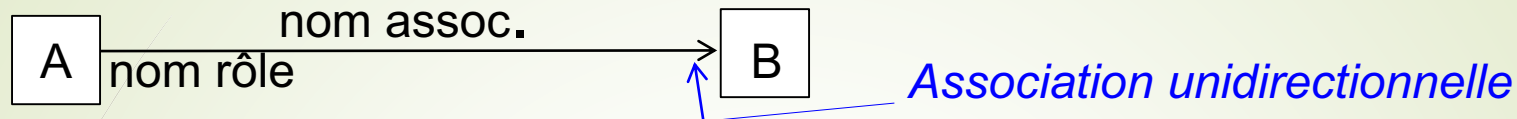
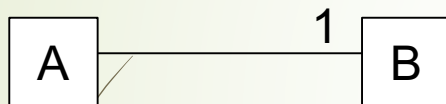


Diagramme de classes

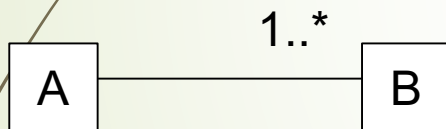
Détails sur associations



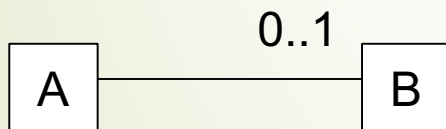
- ➔ Exemples de cardinalités d'associations
- ➔ La cardinalité indique la ou les quantités de la classe la plus proche pour lesquelles la relation peut s'appliquer



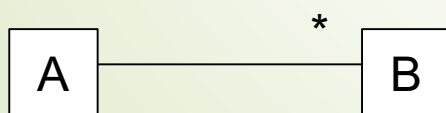
Une instance de la classe A est toujours associée avec une instance de la classe B



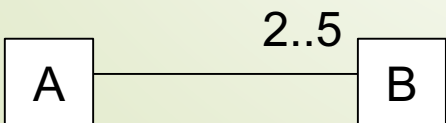
Une instance de la classe A est toujours associée avec une ou plusieurs instances de la classe B



Une instance de la classe A est associée avec zéro ou une instance de la classe B



Une instance de la classe A est associée avec zéro, une ou plusieurs instances de la classe B



Une instance de la classe A est associée avec entre deux et cinq instances de la classe B

Diagramme de classes

- Enumeration
- Liste de valeurs manipulées comme un type
- Classe d'association
- A chaque couple des éléments de l'association, une instance d'une autre classe est associée
- Ici, à chaque employé d'une entreprise sont associées les informations sur son poste

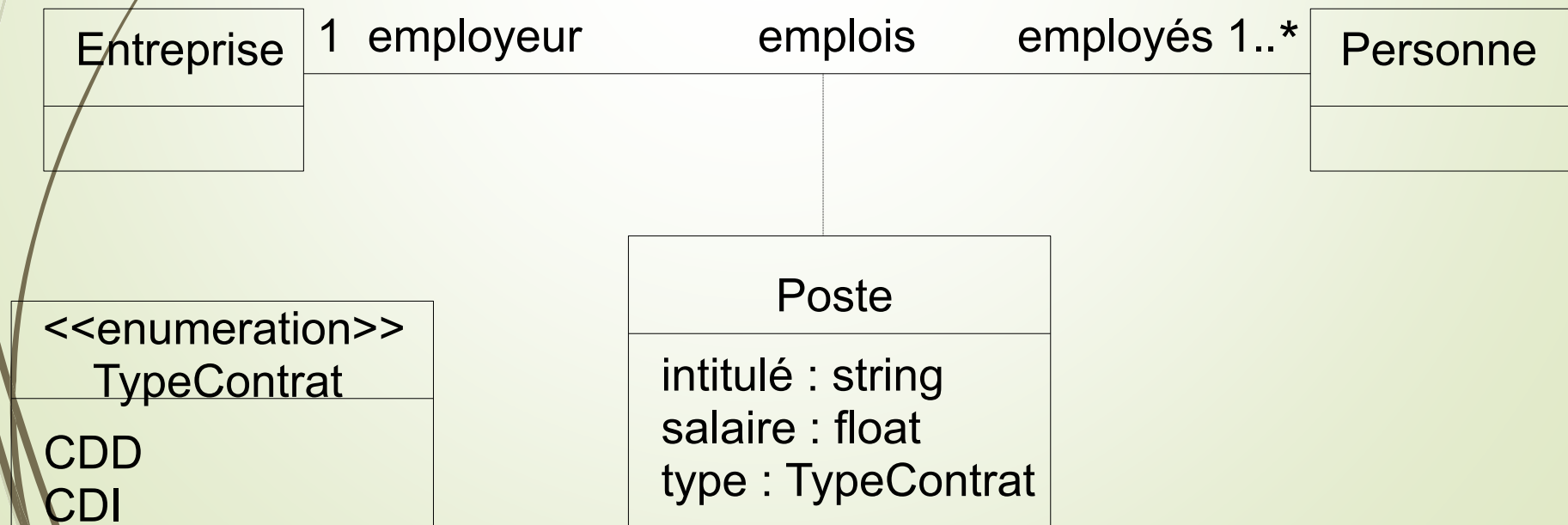
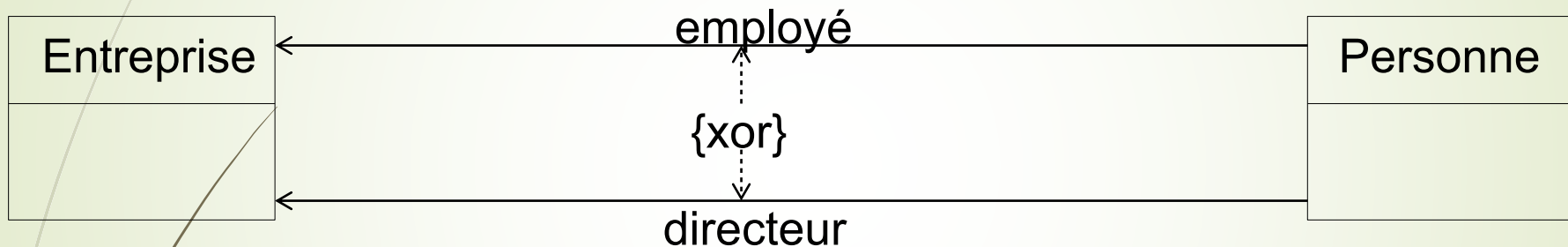


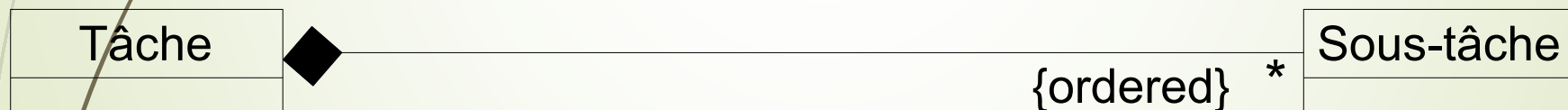
Diagramme de classes

Contraintes sur les associations (en plus des cardinalités)

- Relation d'exclusion entre deux associations : soit l'une soit l'autre mais pas les deux à la fois



- Les éléments d'une association peuvent être ordonnés



- Une association peut être le sous-ensemble d'une autre

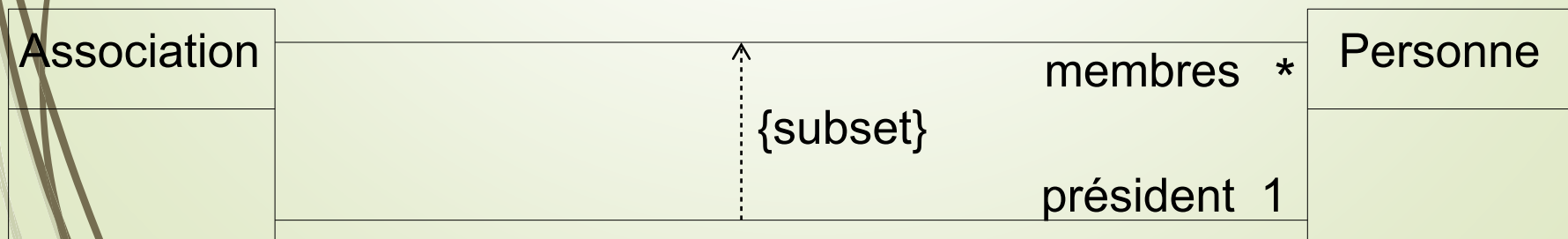
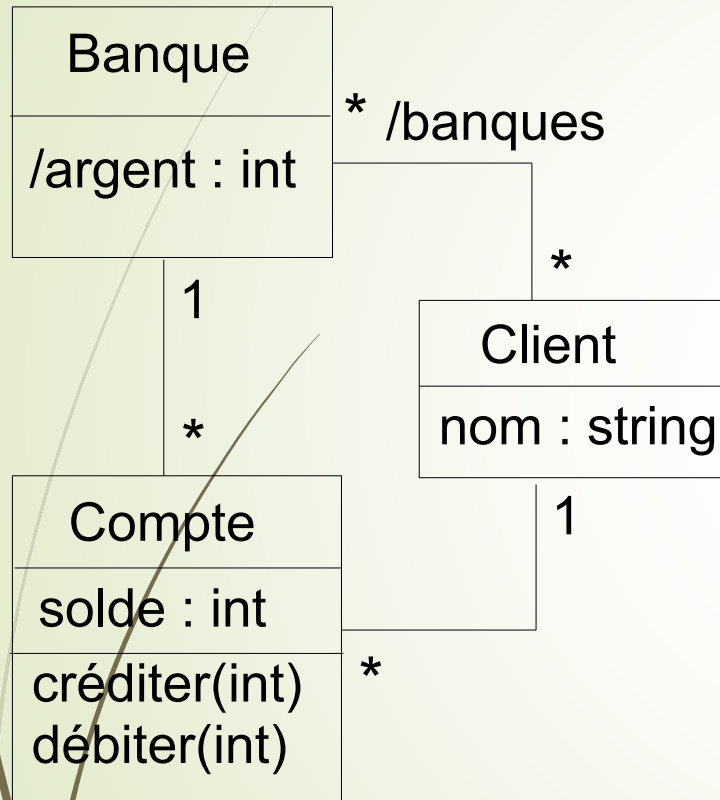


Diagramme de classes



➤ Éléments dérivés

➤ Principalement pour attributs et associations

➤ Se déduisent d'autres parties du diagramme

➤ Nom de l'élément commence par /

➤ Exemples

➤ L'ensemble des banques dont on est client se déduit de ses comptes bancaires

➤ L'argent géré par une banque est la somme des soldes de ses comptes

➤ Ces éléments dérivés peuvent formellement être définis en OCL

An Example of Class Diagram

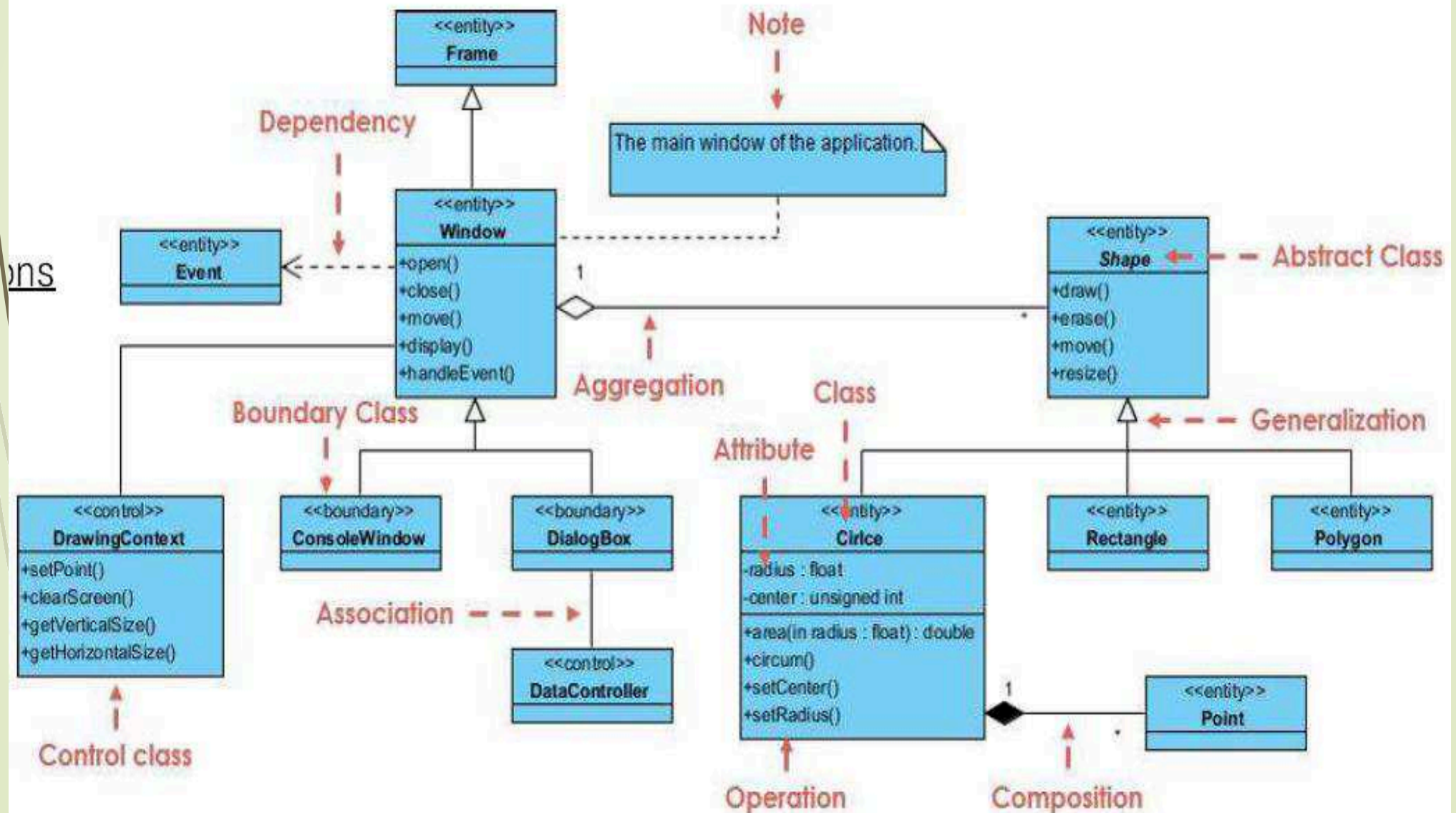



Diagramme de classes



➤ Contraintes

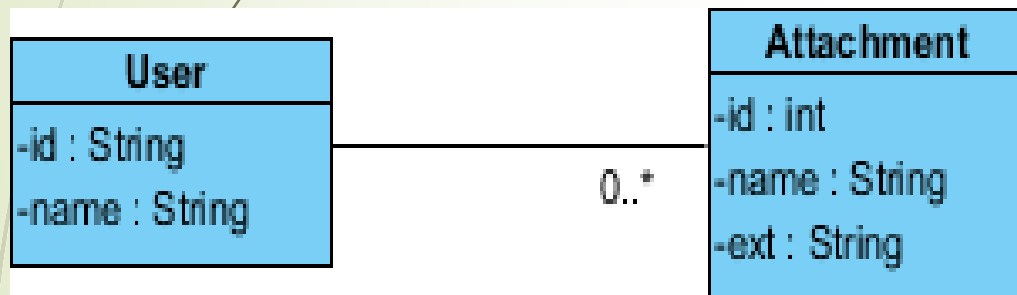
- Associations, attributs et généralisations spécifient des contraintes importantes (relations, cardinalités), mais ils ne permettent pas de définir toutes les contraintes
- UML permet d'ajouter des contraintes sur des éléments (classe, attribut, association, ...)
 - Soit des prédéfinies
 - Exemple : {ordered} et {xor} pour les associations
 - Soit des spécifiques définies par le concepteur
 - Pas de syntaxe précise préconisée, uniquement l'utilisation de { ... }
- En pratique, pour être précis, on exprime ces contraintes en OCL

Diagramme d'objets

- 
- **Objet** = instance d'une classe
 - **Diagramme d'objets** : ensemble d'objets respectant les contraintes du diagramme de classe
 - Respect des cardinalités
 - Chaque attribut d'une classe a une valeur affectée dans chaque instance de cette classe
 - **Diagramme de classes** = définition d'un cas général
 - **Diagramme d'objets** = définition d'un cas particulier de ce cas général

Class Diagram v/s Object Diagram

Class Diagram



Object Diagram

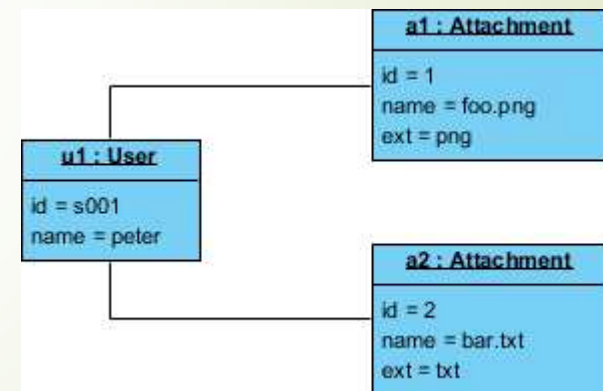


Diagramme de composants

Composant

- Élément spécifiant ses interactions avec l'extérieur via la définition de ses interfaces fournies et requises
- On connecte une interface requise d'un composant à l'interface fournie compatible d'un autre composant : assemblage
- Composant composite
 - Composant peut être formé de composants internes assemblés par leurs interfaces
 - Composition hiérarchique de composants
- Port
 - Point d'interaction du composant
 - Associé à une interface d'opérations (en mode requis ou fourni)
- Connecteur
 - De délégation : lie un port du composite à un port d'un de ses éléments
 - D'assemblage : lie une interface d'un élément interne avec celle d'un autre élément interne

Diagramme de composants

Ensemble de composants connectés entre eux par assemblage ou composition

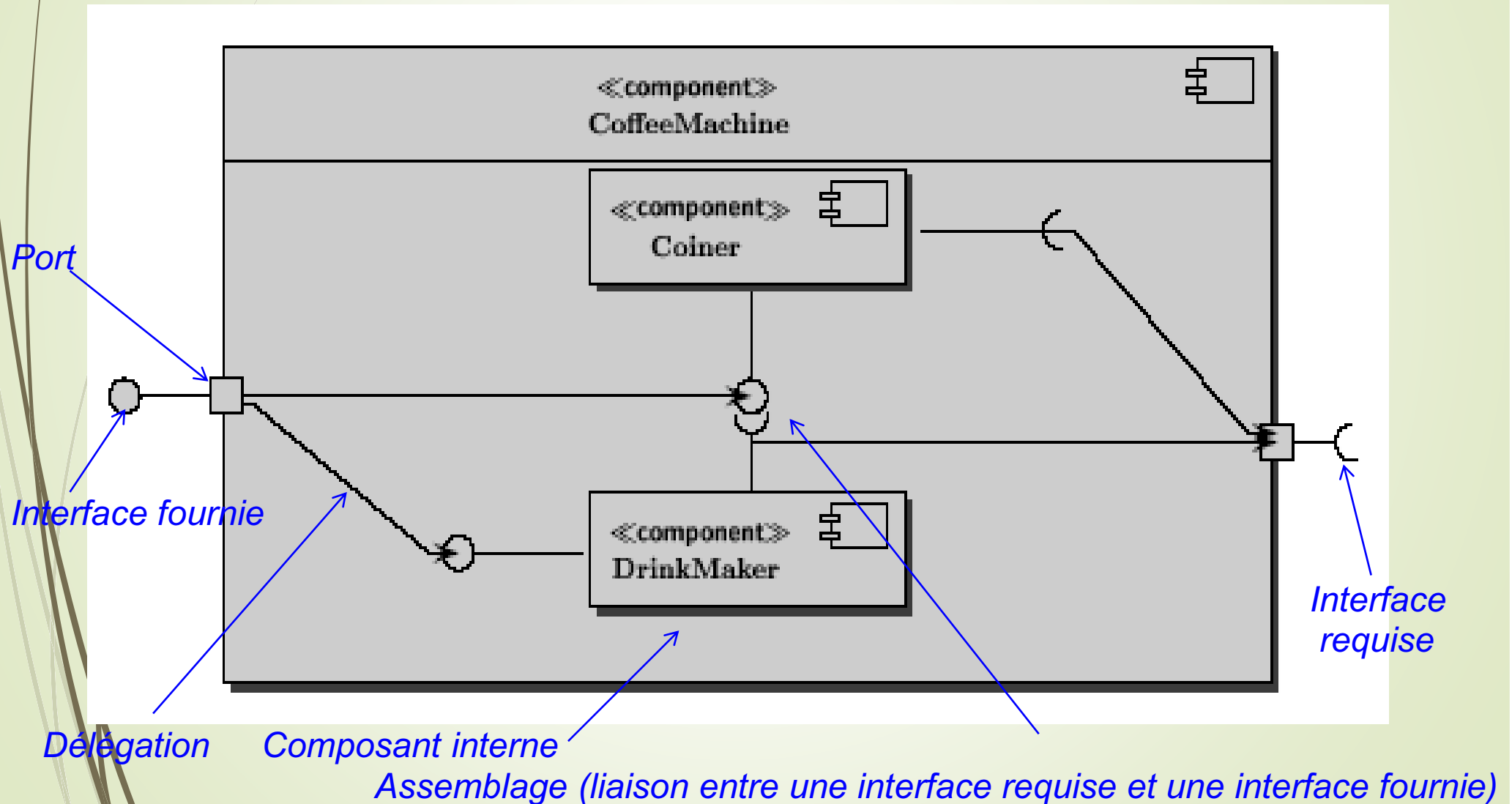


Diagramme de structure composite

- ▀ Diagramme conceptuellement assez proche d'un diagramme de composants
- ▀ Définit l'architecture interne d'une classe
 - ▀ Les éléments qui la forment (les *parts*)
 - ▀ Les interactions entre ces éléments
- ▀ Ici *Engine* est composite (pointillés)

