I.U.T. LUMIÈRE

UNIVERSITÉ
LUMIÈRE LYON 2

Bases de Données Avancées
-
UML et Bases de Données

Pierre Parrend
IUT Lumière Lyon II, 2005-2006
pierre.parrend@univ-lyon2.fr

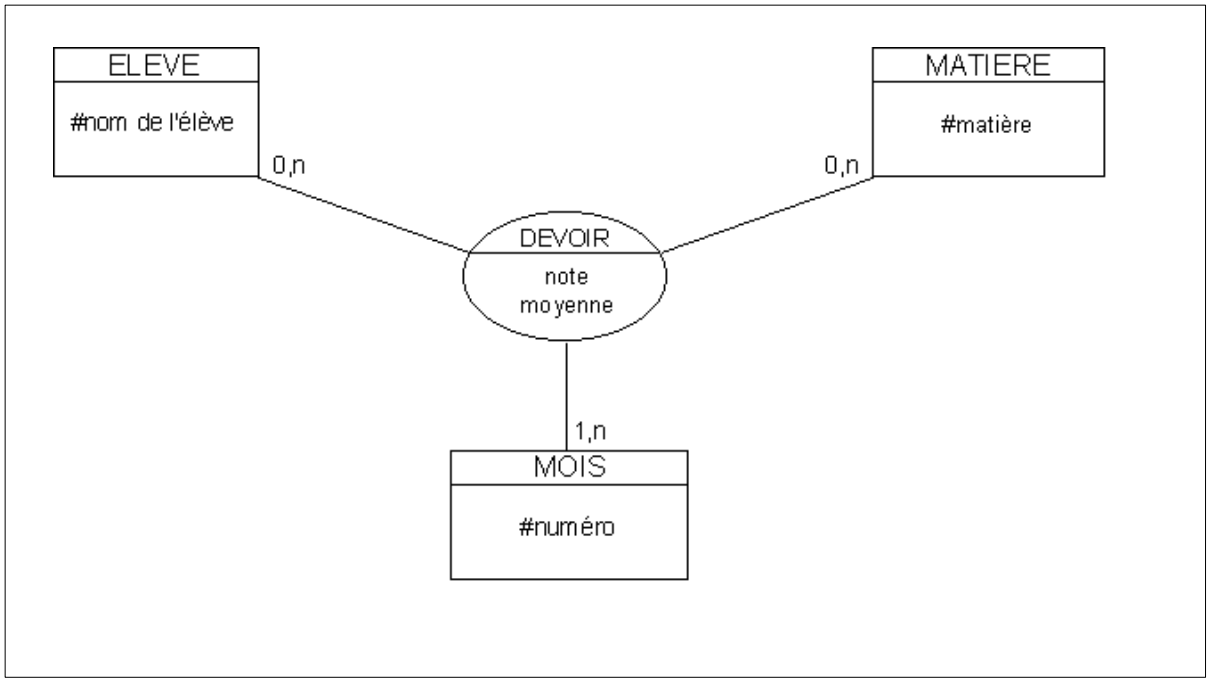
Sommaire

I.A UML

- i. Ce qu'est UML : Principes
  - Analyse Orienté Objet
  - Outil de Conception
    - Diagramme de Cas d'Utilisation
    - Vue Système
    - Analyse des problèmes
  - Outil de Réalisation
    - Diagramme de Classes
    - Vue Programme
    - Résolution des problèmes

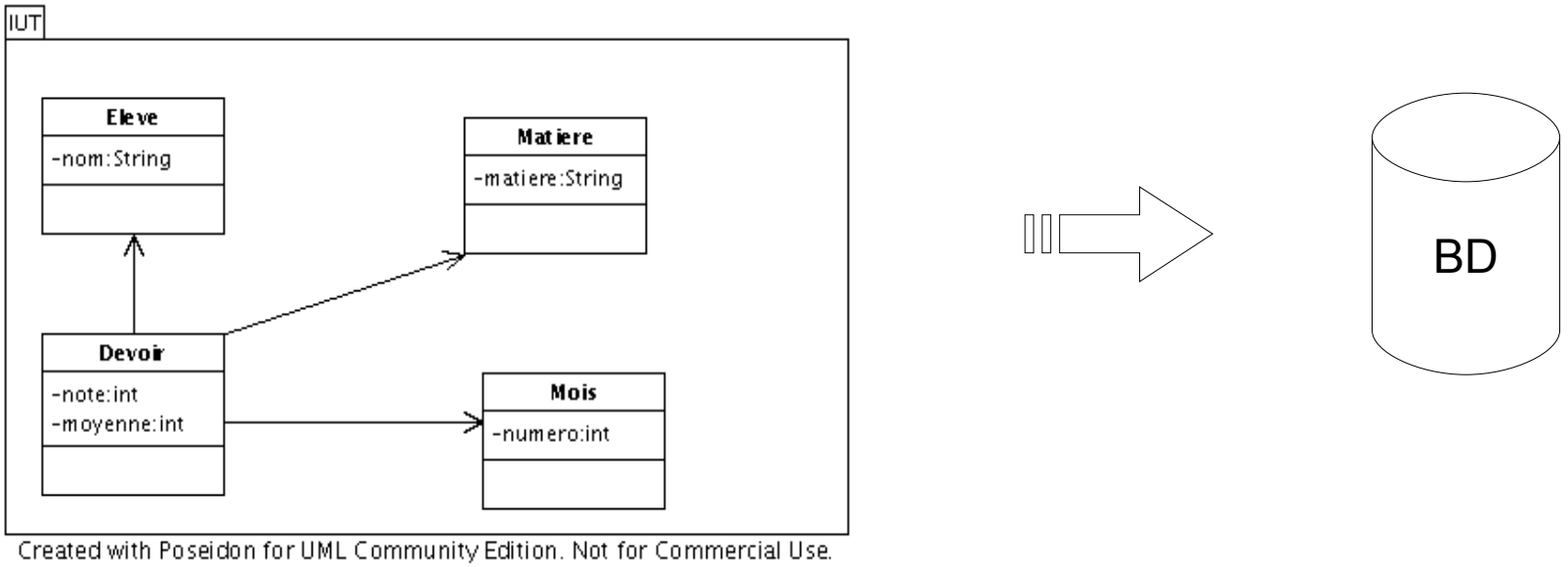
I.A UML

- ii. UML et Bases de Données
  - 1 Objet UML = 1 entité MCD
  - 1 Assocation UML = 1 Relation MCD



I.A UML

- UML et Bases de Données
  - Programme
  - Données
    - Projection directe en Base de Données (Persistence)



I.A UML

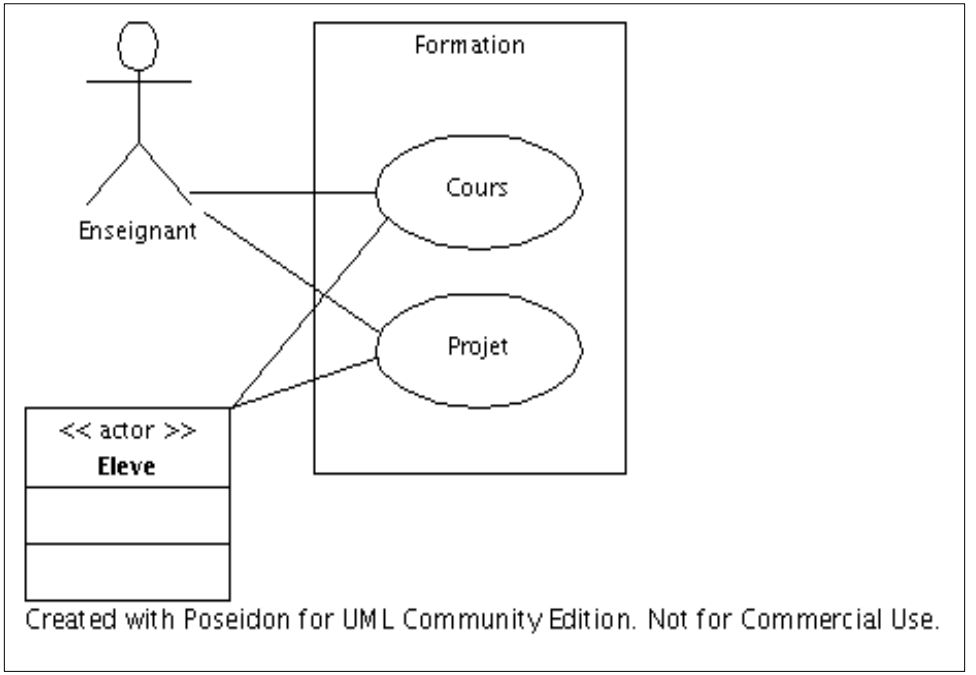
- iii. Programmation Procédurale
  - Séparation Fonctions / Données
- Programmation Objet
  - Analyse des systèmes en Objets
    - = Attributs (Données) + Méthodes (Fonctions)

Sommaire

- **UML**
  - **Ce qu'est UML**
  - **Diagrammes de Cas d'utilisation**
  - **Diagrammes de Classes**
- **UML et les Bases de Données**

I.B UML – Use Case

- i. Diagrammes de Cas d'utilisation
  - Objectif : détermination des besoins
  - Fonctionnalités du système
  - Acteurs
  - Cas d'utilisation
  - Relations entre les 2
    - association
  - Système



I.B UML – Use Case

- Les Diagrammes de cas d'utilisation
  - Formalisation du cahier des charges
  - Evolutions du système
  - Centrés sur l'utilisateur
  - Expression simple
    - Permet le dialogue entre le client et le développeur
  - Point de départ du développement

I.B UML – Use Case

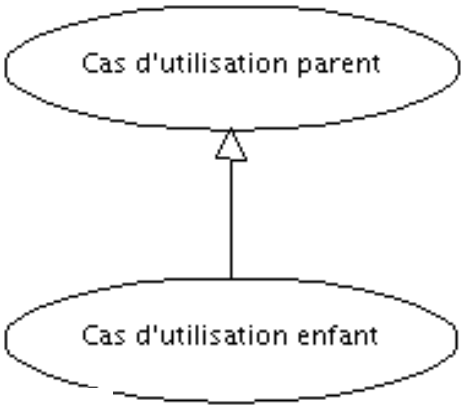
- ii. Les acteurs
  - 1 acteur = plusieurs personnages
    - Ex : acheteurs
  - 1 personnage = plusieurs acteurs
    - Ex : boulanger, vendeur
  - Toute personne qui interagit avec le système

I.B UML – Use Case

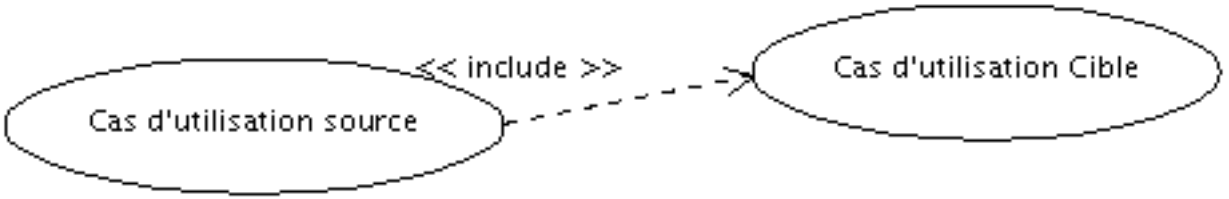
- Les acteurs
  - Acteurs principaux
    - Ex : client, dans le cas d'un distributeur de billets
  - Acteurs secondaires
    - Maintenance, tâches administratives
  - Matériel externe
    - Indispensable au domaine. Ex : imprimante pour le reçu
  - Autres systèmes
    - Ex : système bancaire

I.B UML – Use Case

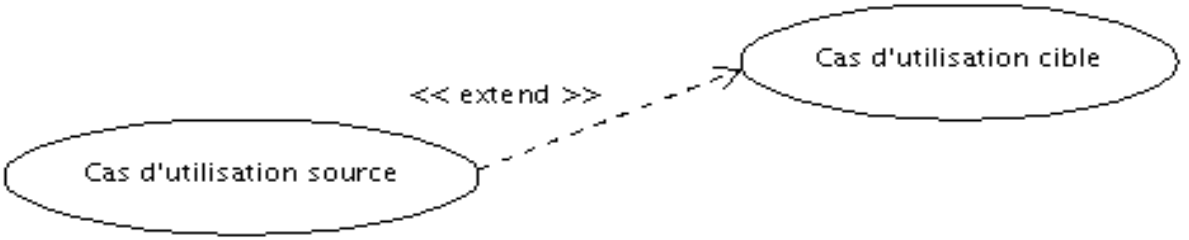
- iii. Les cas d'utilisation
  - Relation de généralisation



  - Relation d'inclusion



  - Relation d'extension



I.B UML – Use Case

- iv. Les cas d'utilisation
  - Représentent
    - Les tâches de l'acteur
    - Informations créées ou utilisées par l'acteur
    - Changements externes nécessaires au système
  - Doivent être simples
    - Possibilité de les fractionner
  - A partir de scénarii

I.B UML – Use Case

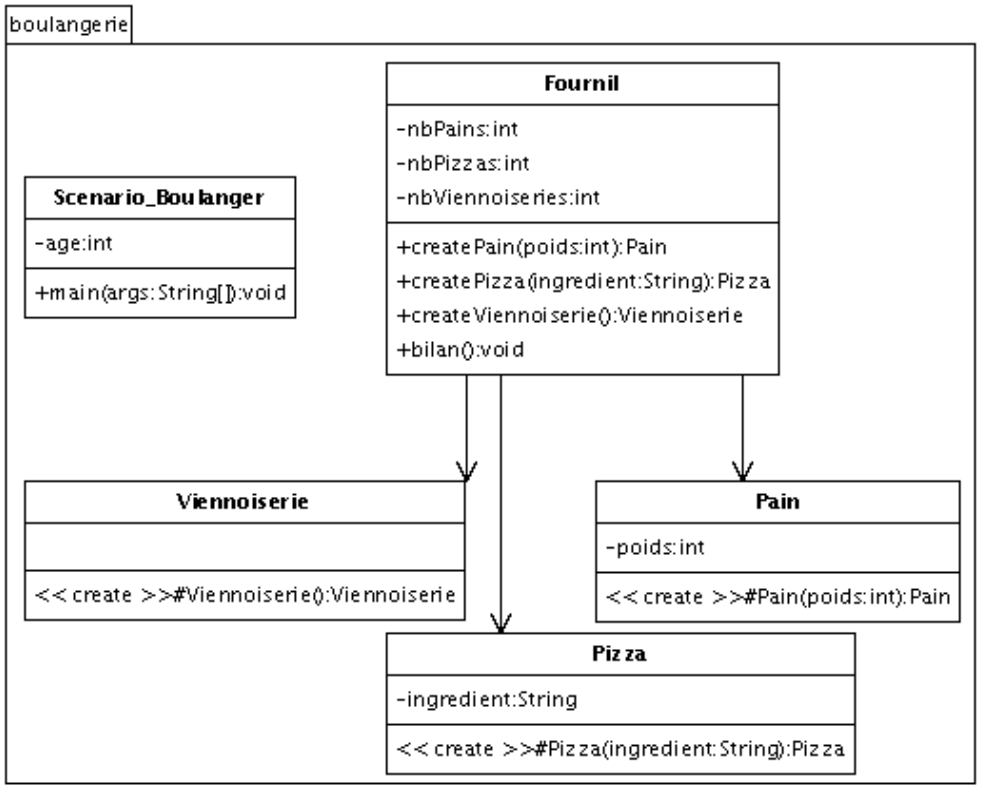
- Les cas d'utilisation
  - Analyse
    - Pas conception
  - 'Build the right system'
    - Pas 'Build the system right'
  - Passage à une vue Objet indispensable
    - Pour la réalisation du système

Sommaire

- **UML**
  - **Ce qu'est UML**
  - **Diagrammes de Cas d'utilisation**
  - **Diagrammes de Classes**
- **UML et les Bases de Données**

I.C UML - Classes

- i. Diagrammes de Classes : Exemple

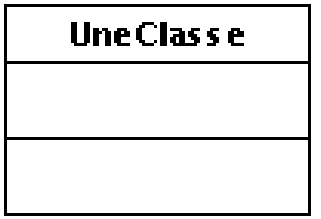


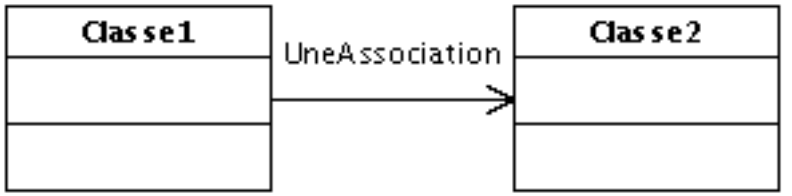
I.C UML - Classes

- ii. Diagrammes de Classes : Principes
  - Structure statique des systèmes
  - Contient
    - Classes
    - Relations entre ces classes
    - Interfaces
    - Packages

I.C UML - Classes

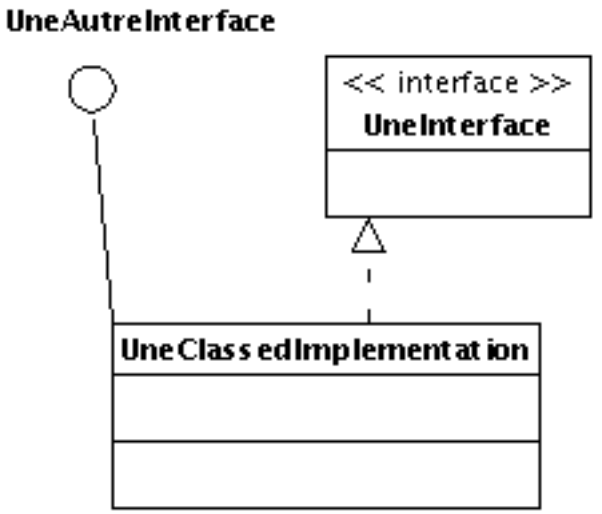
- iii. Eléments
  - Classe
    - Instanciée par des objets
    - Unique dans un package
    - Syntaxe : 'nomPackage :: nomClasse'
    - Stéréotype, propriétés
  - Associations
    - Instanciées par des liens





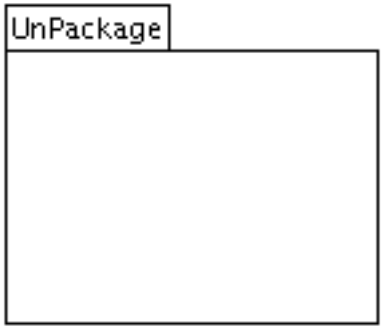
I.C UML - Classes

- Eléments
  - Interface
    - 'Vue totale ou partielle sur un ensemble de services'
    - Descripteur des opérations
    - Sans code
    - Pas d'attribut
    - Pas d'association



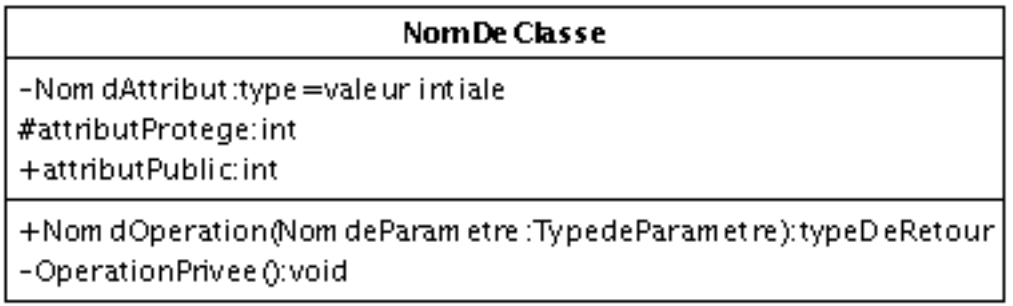
I.C UML - Classes

- Eléments
  - Package



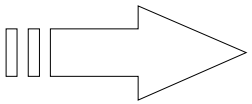
I.C UML - Classes

- iv. Attributs et Opérations
  - Les compartiments
    - Nom de la classe
    - Attributs
    - Opérations



I.C UML - Classes

- Attributs et Opérations
  - Attributs
    - ***Ce qu'est la classe***
    - Nom : type = valeur initiale
    - Test : boolean = false
    - Peut être constant
      - 'const'
    - Représentation par la composition
      - Voir Associations
      - Représentation implicite



Données d'une classe

I.C UML - Classes

- Attributs et Opérations
  - Opérations
    - ***Ce que fait la classe***
    - Nom (nom de paramètre : type de paramètre) : type de Retour
    - multiplier(a : int, b : int) : int
    - Propriétés
      - Direction des arguments des opérations
      - Types d'opération

I.C UML - Classes

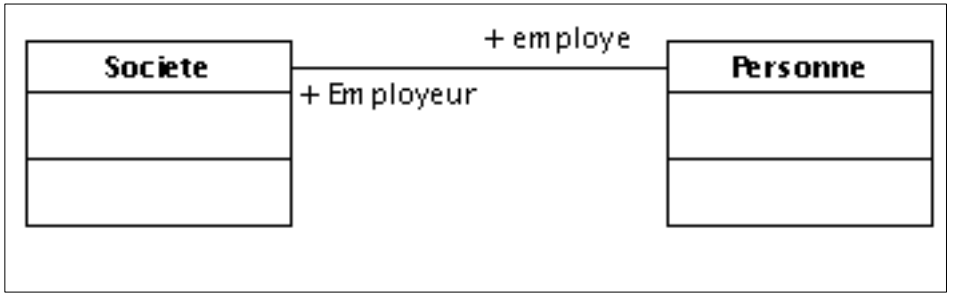
- Attributs et Opérations
  - Visibilité
    - Pour attributs et opérations
    - + public
      - Accessible par tous les objets (dans et hors de la classe)
    - # protected
      - Accessible seulement par la classe et les sous-classes
    - - private
      - Accessible seulement par les objets de la classe

I.C UML - Classes

- v. Associations
  - Relation structurelle entre deux classes d'objets
  - Durée de vie non négligeable
    - Par rapport aux objets qui instancient les classes concernées
  - Relie deux classificateurs
    - Classes, interfaces
    - Parfois plus : association représentée par une classe

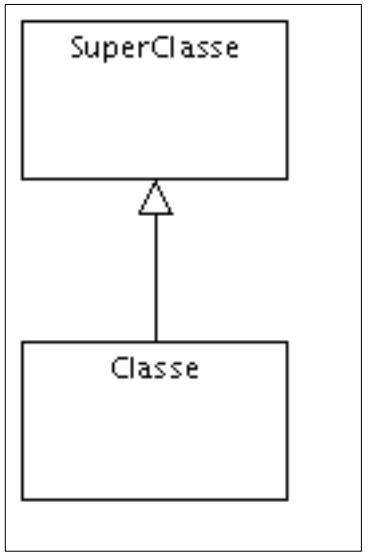
I.C UML - Classes

- Associations
  - Rôles
    - Extrémité d'une association
    - Indication des rôles relatifs des deux classes reliées par association
    - Pseudo-attribut de la classe source
      - Ex : Employeur est un pseudo attribut de la classe Personne
    - Indication de visibilité
      - Public par défaut
      - Privé (-) ou protégé (#)



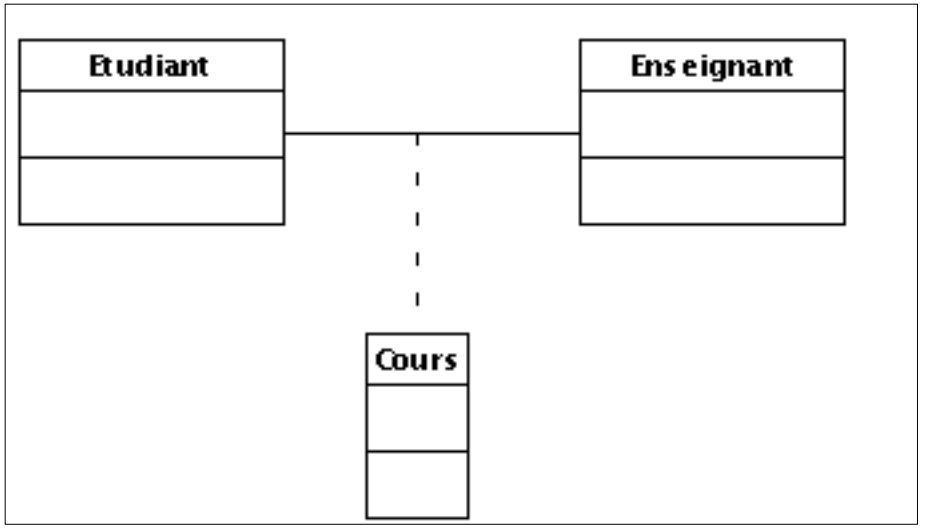
I.C UML - Classes

- Associations
  - Généralisation
    - Une classe générique (super classe)
    - Une classe spécialisée (classe fille)



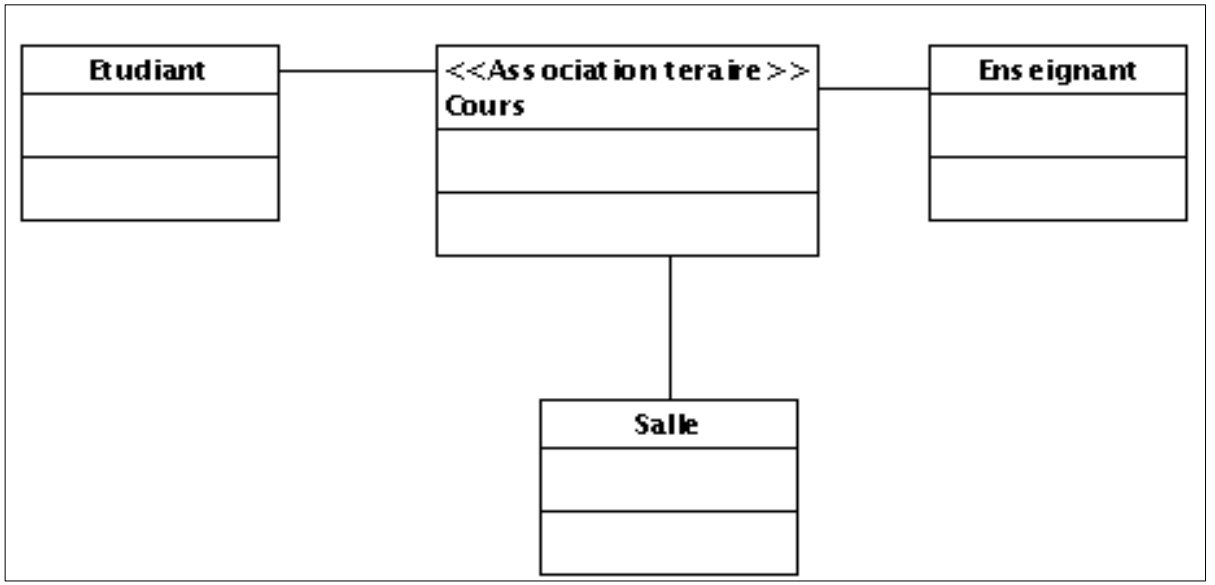
I.C UML - Classes

- Associations
  - Classe d'association
    - L'association peut être manipulée



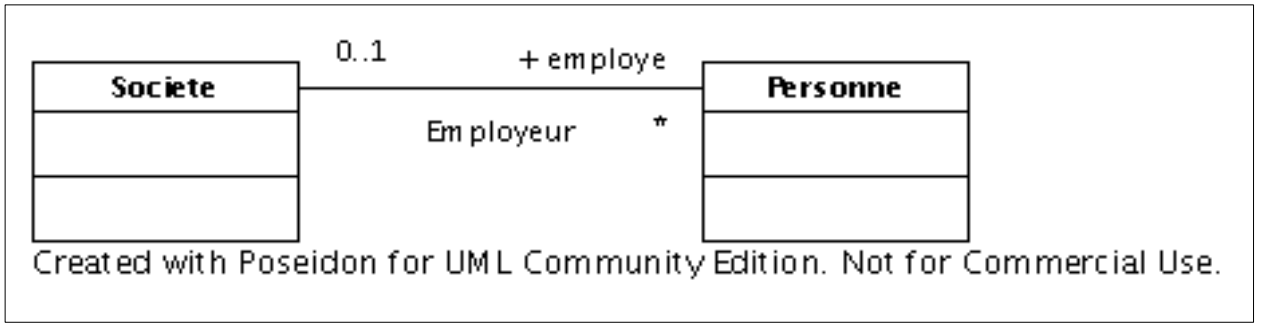
I.C UML - Classes

- Associations
  - Arité
    - Associations binaires - classiques
    - Associations n-aires
      - Parfois représentées par un losange



I.C UML - Classes

- Associations
  - Multiplicité
    - Contraintes liées au domaine d'application
    - Valable pendant toute la vie de l'objet
    - Pas d'influence sur l'ordre de création des objets (associations simples)



I.C UML - Classes

- Associations
  - Multiplicité
    - possibilités

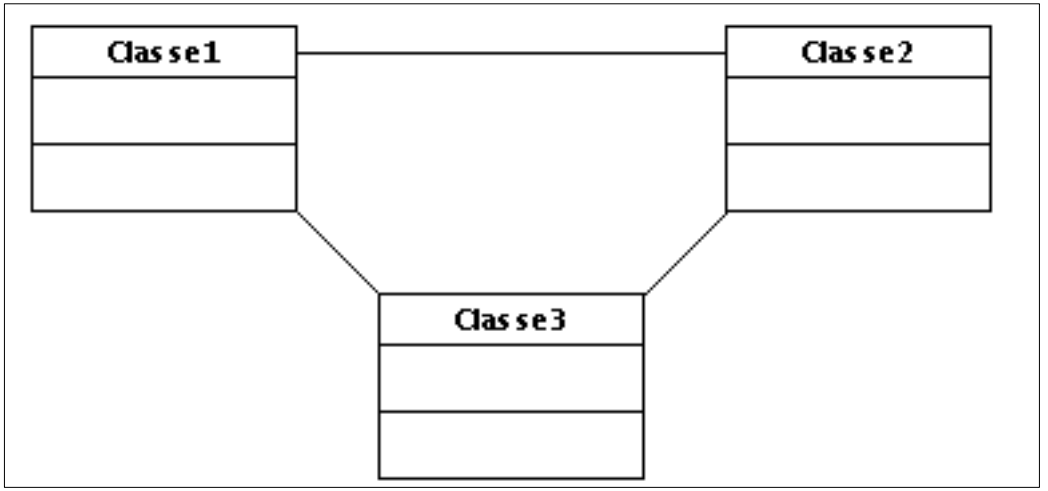
1	Un seul
0..1	Zéro ou un
N	N (entier naturel)
M..N	De M à N (entiers naturels)
*	De zéro à plusieurs
0..*	De zéro à plusieurs
1..*	D'un à plusieurs

I.C UML - Classes

- Associations
  - Navigabilité
    - Possibilité d'accès d'une classe à l'autre
    - Association orientée
      - Compte est un attribut de Personne

I.C UML - Classes

- Associations
  - Navigabilité
    - Association bidirectionnelle
      - Une modification dans une classe est répercutée dans la classe associée
      - Complexe si association en chaîne
      - Erreur si boucle d'associations

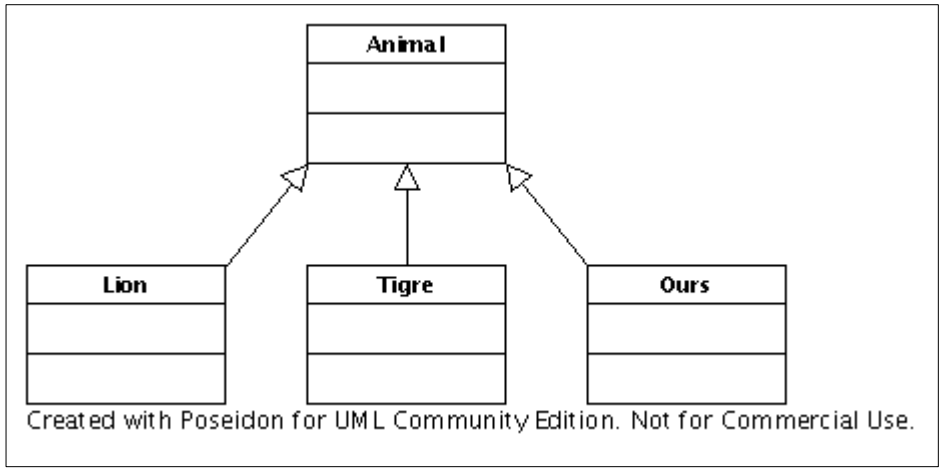


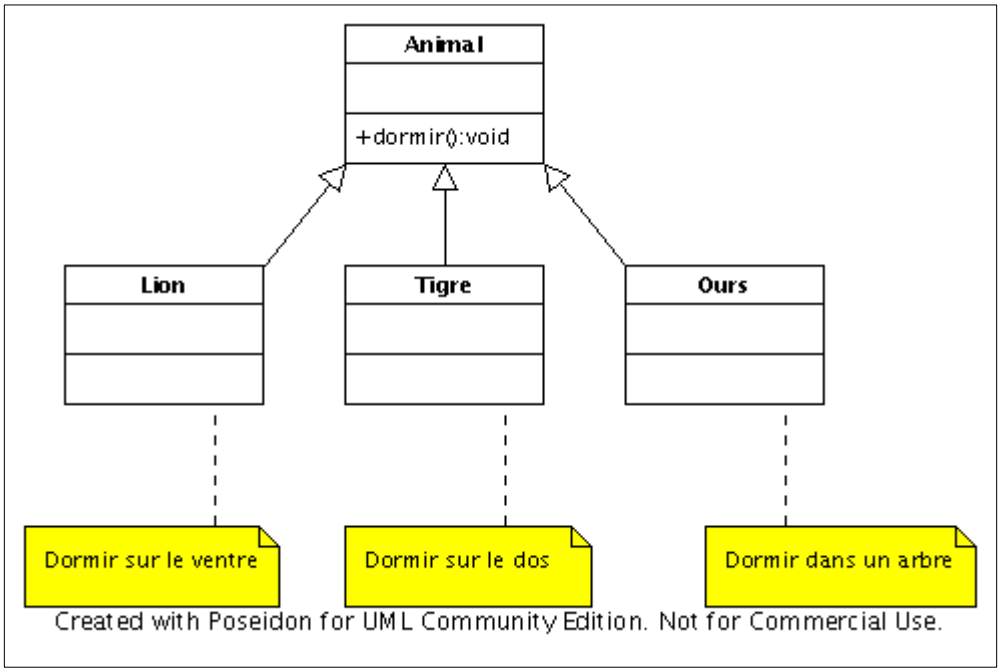
I.C UML - Classes

- vi. Principe de Substutition
  - Héritage = relation de classification
  - Toute Superclasse doit pouvoir être remplacée par une sous-classe
  - Conservation de la sémantique du programme
  - Malgré la réecriture des méthodes
- Polymorphisme
  - Un nom d'objet peut désigner des instances de classes différentes
  - Polymorphisme d'opération
    - Méthode réecrite
    - Le même message peut entraîner plusieurs réactions

I.C UML - Classes

- Polymorphisme - exemple





Sommaire

- **UML**
  - **Ce qu'est UML**
  - **Diagrammes de Cas d'utilisation**
  - **Diagrammes de Classes**
- **UML et les Bases de Données**

II. UML et Bases de Données

- Classes = tables
- Attributs = champs
- Objets = Instances
- Association = relation entre tables
- Mapping direct entre Programmes Objets et Bases de données
  - Persistence

Bilan

- Diagrammes
  - Cas d'utilisation
  - Classes
- Relation avec Merise
  - Prog. Objet vs Prog. Procédurale
  - Persistence implicite des Objets