Modèles Statiques

Diagrammes d'objets et de classes

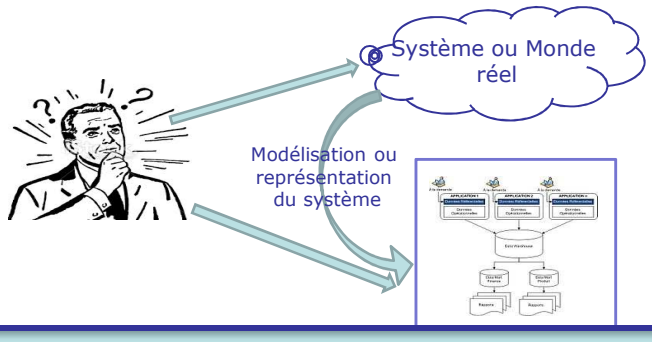
Pr. Larbi Kzaz

Octobre 2018

Introduction

Paradigme ou Approche Objet: de quoi s'agit-il?

- ✓ C'est une façon de Voir les choses ou d'aborder des problèmes. Et qui repose sur un fondement défini: modèle théorique, courant de pensée, etc.
- ✓ En *Géni Logiciel*, le paradigme ou encore *Approche Objet*, traduit une façon particulière de voir et d'aborder un *Système*, appelé aussi *'Monde réel'*, par les différents *Acteurs* du GL: *Analystes, Concepteurs, Développeurs*.



Introduction

Paradigme ou Approche Objet:

Le paradigme *Objet* repose sur l'idée suivante:

- ✓ Tout système est une collection d'éléments et d'entités du monde réel pouvant être concrètes (chose physique, équipement, être humain) ou abstraites (chose immatérielle, concepts, notions, idées, etc.). Les entités du système ont une existence propre; elles jouissent d'une certaine autonomie, agissent et interagissent entre elles.
- ✓ Un *OBJET* est une *représentation abstraite* d'une ENTITE du monde réel.

Remarques:

- ✓ La représentation du système traduit ce que voit et perçoit un INDIVIDU en fonction de ses PROPRES CAPACITÉS COGNITIVES ET SES PRÉOCCUPATIONS.
- ✓ Des individus différents peuvent ainsi produire des représentations (modèles) différentes d'un même système.

Introduction

Exemples:

- Domaine Scolarité du Système « Ecole ou université »

  Objets: Etudiant, Professeur, Matière, Salle de cours, Séance, Examen, etc.

- Domaine Commercial du Système « Entreprise »

  Objets: Client, Fournisseur, Article, Bon de Commande, Facture, etc.

- Domaine RH du Système « Entreprise »

  Objets: Salarié, Grille des Salaires, Fonctions, Postes de travail, Congé, Formation, etc.

- Domaine Commercial du Système « Agence Immobilière»

  Objets: Bien immobilier, Propriétaire, Locataire, Rendez-vous de Visite, Contrat de Vente, Contrat de Location, etc.

Concept d'Objet

Notion d'Objet:

C'est une représentation abstraite d'un élément concret ou abstrait du monde réel.

Modélisation

- ETUDIANT
- PROFESSEUR
- SALLE DE COURS
- MATIERE
- SEANCE
- ……

éléments du Monde réel

Représentation des objets du monde réel

Concept d'Objet

Caractérisation d'un objet:

Tout objet est caractérisé par:

- ✓ *Une Identité:*

  C'est une donnée « naturelle » ou « artificielle » (créée spécialement), permettant de distinguer un objet des autres objets du système.

- ✓ *Des Attributs:*

  Ce sont des données (Variables, Propriétés) associées à un objet et dont les valeurs peuvent varier dans le temps. Les valeurs des attributs d'un objet à un instant précis constituent l'*ETAT* de l'Objet, et permettent le suivi de son évolution dans le système.

- ✓ *Un Comportement:*

  Il s'agit de l'ensemble des opérations qu'un objet est capable de réaliser. Ces opérations sont appelées aussi *Méthodes*.

Concept d'Objet

Exemples:

Proposer des *attributs* et des *méthodes* pour chacun des objets suivants:

Objets Informatique (Système d'Exlpoitation) :

- Fenêtre: Window
- Fichier
- Disque

Objets Mécanique :

- Roue
- Moteur

Objet du domaine RH :

- Salarié

Objet du domaine Banque :

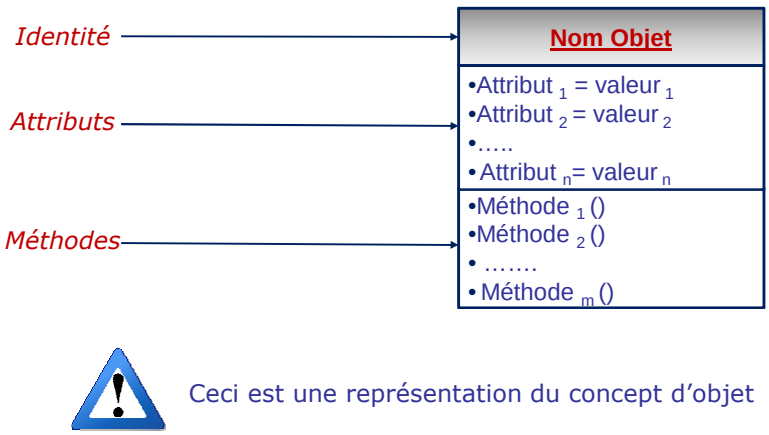
- Compte Bancaire

Objet du domaine Scolarité:

- Etudiant

Concept d'Objet

Représentation d'un objet:



Ceci est une représentation du concept d'objet

Concept d'Objet

Etat d'un objet:

A chaque instant chaque *attribut* d'un objet possède une *valeur*. L'ensemble des valeurs des différents attributs d'un objet définissent son *état* à un instant donné.

Objet
•Attribut $_1$ = valeur $_1$ •Attribut $_2$ = valeur $_2$ •….. •Attribut $_n$= valeur $_n$
•Méthode $_1$ () •Méthode $_2$ () • ……. • Méthode $_m$ ()

Ma voiture
•numéroMatricule $_1$ = "1478 -ب – 8 " •marque= "DACIA " •dateMiseEnCirculation= "13/04/2016 " •compteurKilométrique= 20470
•démarrerMoteur() •arrêterMoteur() •incrémenterCompteur() • calculerAger() • controlerNiveauHuile()

Concept de Classe

Notion de Classe:

Une *Classe* regroupe un *Ensemble d'objets semblables*. Les objets d'une classe ont les *mêmes attributs* (caractéristiques) et possèdent les *mêmes méthodes* (même comportement)

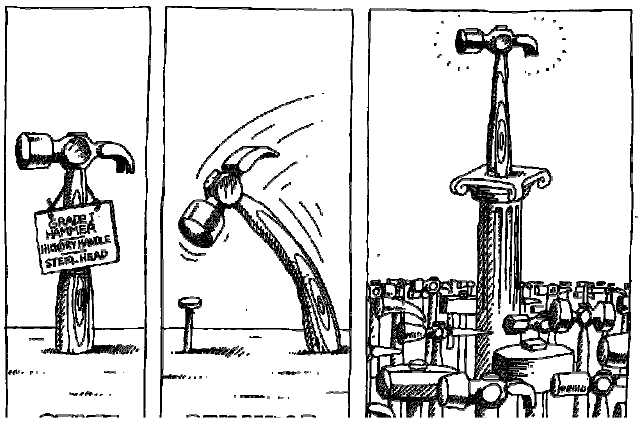
- Un objet est un *élément* d'une classe.
- On dit aussi qu'un Objet est une *Instance* d'une classe.
- Deux objets d'une même classe ont des *identités différentes*. (Principe d'Identité)

Quelle analogie pourrait-on faire avec:

- le moule,
- la statuette, et
- la fabrication d'une statuette?

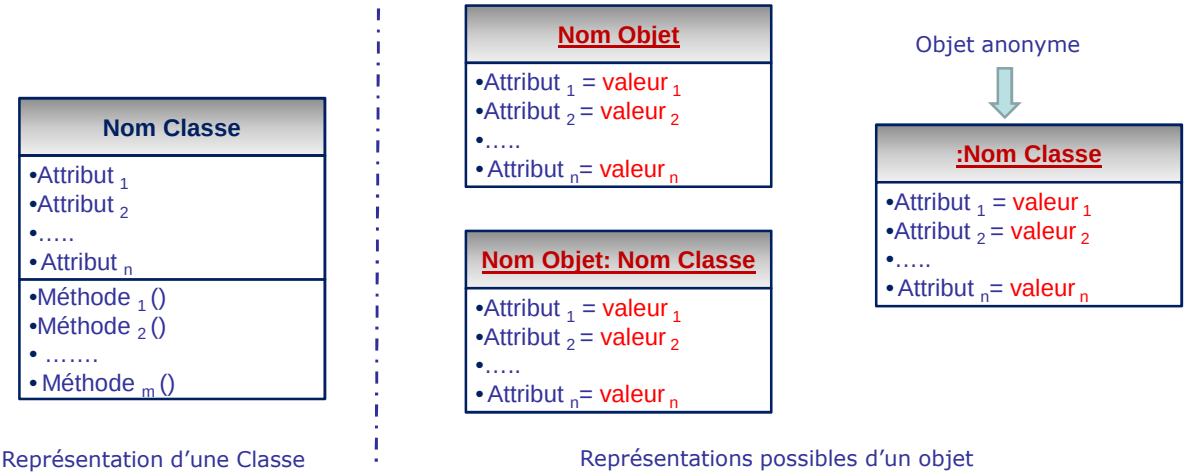
Concept de Classe

Exemple:



Concept de Classe

Représentation des Objets et des Classes:



Représentation d'une Classe

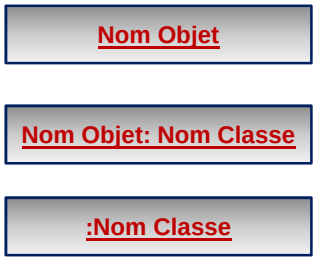
Représentations possibles d'un objet

Objets ET Classes

Représentation simplifiée des Objets et des Classes:

Durant les premières étapes de modélisation on pourra pour des raisons pratiques, se limiter aux représentations simplifiées suivantes:

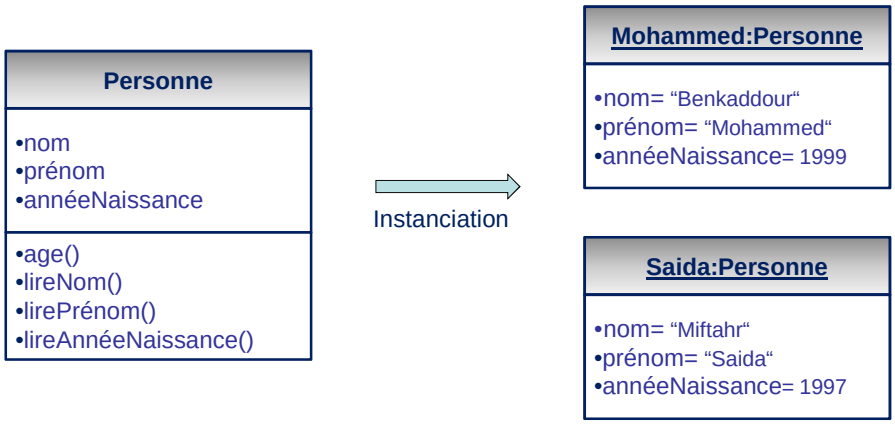
Représentation simplifiée d'une Classe



Représentations simplifiées d'un objet

Objets ET Classes

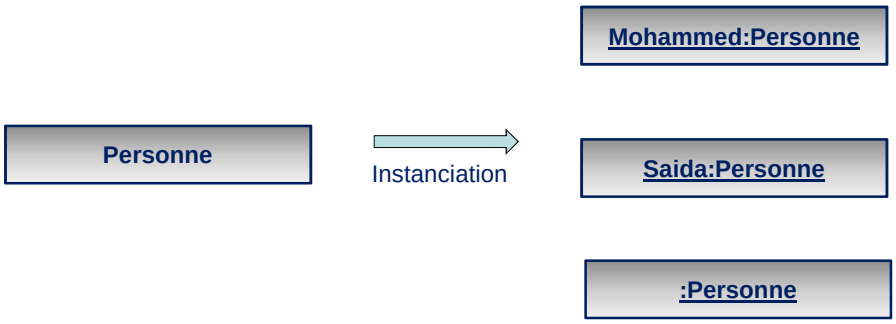
Exemple:



Deux instances de la classe Personne

Objets ET Classes

Exemple:

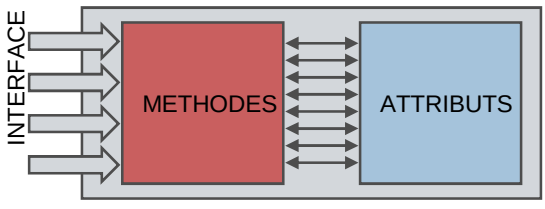


Représentations simplifiées des classes et des objets

Encapsulation et Visibilité

Principes:

- ✓ L' *Encapsulation* consiste à cacher et les attributs d'une classe et à les rendre inaccessibles aux autres classes du système.
- ✓ La seule façon d'accéder aux attributs d'une classe est de lui *envoyer un message* (*appel de méthode*), qui va déclencher l'exécution de l'une de ses méthodes.



Encapsulation et Visibilité

Principes:

- ✓ L'accès à un attribut consiste à:
  - Lire ou Récupérer sa valeur; cela permet de connaitre l'état d'un objet, instance d'une classe.
  - Ecrire ou Modifier sa valeur; cela permet de modifier l'état d'un objet, instance d'une classe.
- ✓ Des méthodes de type LireAttribut(), EcrireAttribut(), InitialiserAttribut(), AffecterAttribut() font partie des méthodes associées Classes.

Encapsulation et Visibilité

Niveaux de visibilités:

Elle définit les niveaux de visibilité, ou encore les droits d'accès, des attributs et méthodes d'un objet. On distingue trois niveaux de visibilité:

- ✓ *Publique* (+) *:* L'attribut, ou la méthode, est publique; elle est accessible aux autres objets du système.
- ✓ *Privé* (-) *:* L'attribut, ou la méthode, est privé; il n'est accessible que par les méthodes de l'objet.
- ✓ *Protégé* (#) *:* L'attribut, ou la méthode, est protégé; il n'est accessible que par les méthodes des objets héritiers.

Remarque:

- Les attributs et les méthodes publiques forment la partie Interface de la Classe.

Attributs et Méthodes de Classe

Définition:

Par défaut, chaque instance d'une classe possède sa propre copie des attributs de la classe. Les valeurs des attributs peuvent donc différer d'un objet à un autre. Chaque objet a son propre état.

Parfois, on a besoin de définir un attribut qui garde *une valeur unique* et *partagée* par toutes les instances de la classe.

Cet attribut s'appelle **attribut de classe**

Attributs et Méthodes de Classe

Définition:

Les *attributs* et *les méthodes de classe* sont des attributs et des méthodes qui *n'ont pas de sens pour les objets* lorsqu'ils sont pris individuellement; ils sont définis et sont *significatifs au niveau de la classe* dans son ensemble.

Les *attributs* et *les méthodes de classe* sont aussi appelés dans le jargon des *langages de programmation* des *attributs* ou des *méthodes statiques*. (static en Java ou en C++)

Une *méthode de classe* ne peut manipuler que des *attributs de classe* . Elle n'a pas accès aux *attributs de **la** classe* (i.e. des instances de la classe).

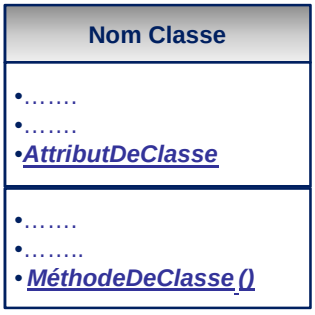
L'accès à une *méthode de classe* ne nécessite pas l'existence d'une instance de cette classe.

Exemple:

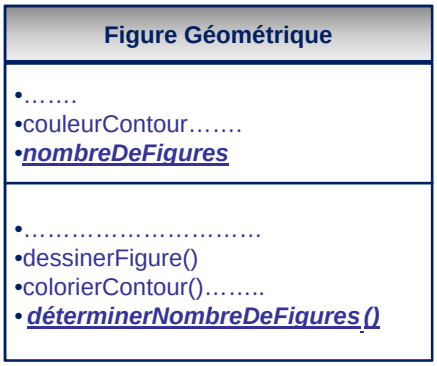
- Le nombre d'instances d'une classe est un *attribut de classe*.
- La méthode qui calcule le nombre d'instances d'une classe est une *méthode de classe*.

Attributs et Méthodes de Classe

Notation:



Exemple:



Relations

Relations entre classes:

Les Objets du système peuvent avoir des relations; on distingue quatre types de relations:

- **Héritage** http://uml.free.fr/cours/p15.html
- **Agrégation**
- **Composition**
- **Association**

Relations : Héritage

Présentation:

- *L'héritage est une relation qui permet de **modéliser des hiérarchies**, lorsqu'elles existent, entre les classes.*

Elle permet de:

- ✓ *Gérer la complexité, en ordonnant les objets au sein d'arborescences de classes, d'abstraction croissante.*
- ✓ *Etablir une sorte de classification entre un ensemble d'objets ayant des attributs et des méthodes en communs et d'autres qui ont des attributs et des méthodes spécifiques.*

- *Deux approches sont possibles:*
  - ✓ *Approche **Ascendante**:*

    Elle Consiste à partir d'objets spécifiques pour aboutir à des objets plus généraux.

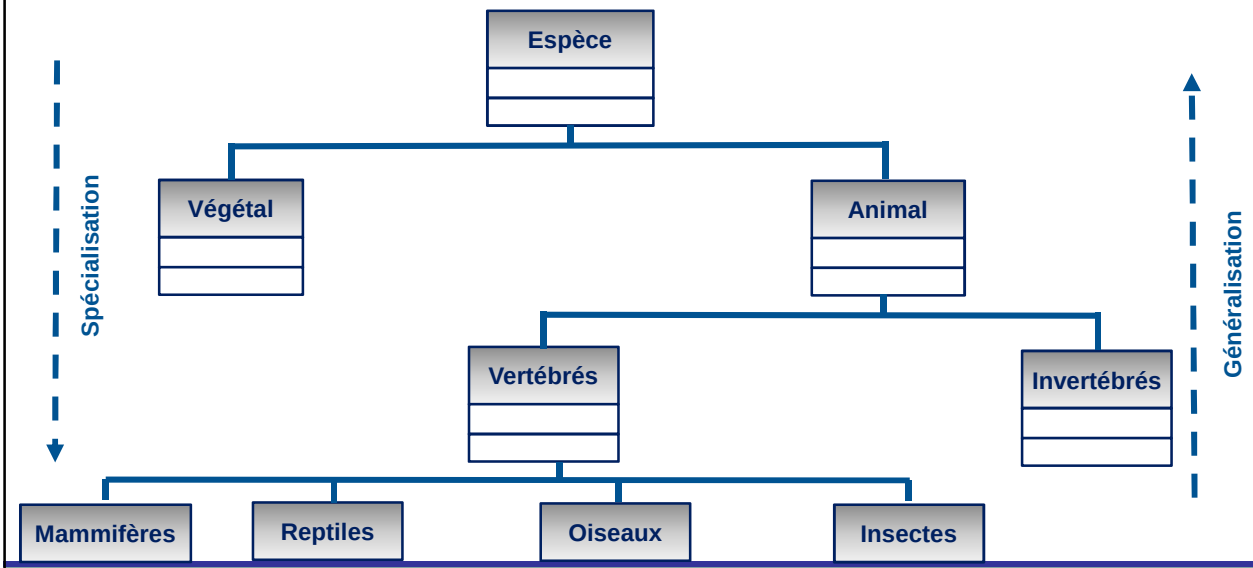
    *Cette approche porte le nom de **Généralisation.***
  - ✓ *Approche **Descendante**:*

    Elle Consiste à partir d'objets Généraux pour aboutir à des objets spécifiques.

    *Cette approche porte le nom de **Spécialisation**.*

Relations : Héritage

Exemple : Classification des espèces:

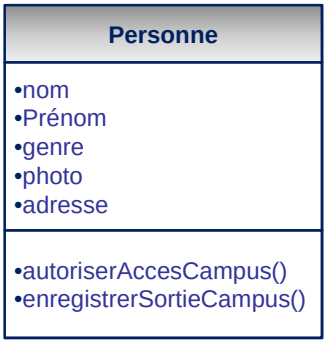


Relations : Héritage

Exemple Introductif:

Supposons qu'on souhaite modéliser l'ensemble des personnes qui fréquentent notre campus.

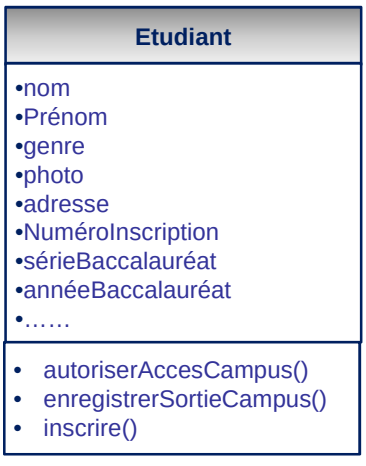
Première approche: On représente toute la population par une classe Unique « Personne »

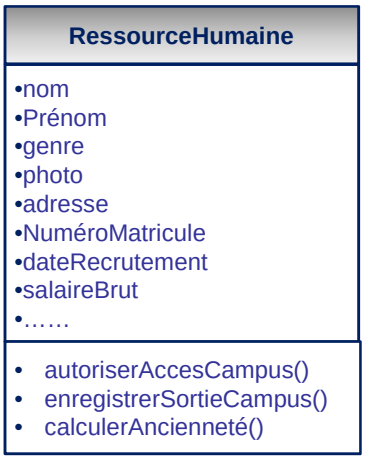


Relations : Héritage

Exemple Introductif:

✓ *Deuxième Approche: On représente cette population par **deux classes**:*





Relations : Héritage

Première approche

On remarque que parmi les personnes, certains ont des attributs, resp. des méthodes, spécifiques.

Ainsi, les étudiants ont, en plus des attributs de la classe PERSONNE, *un numéro d'inscription, une série de baccalauréat et l'année de son obtention, etc.*

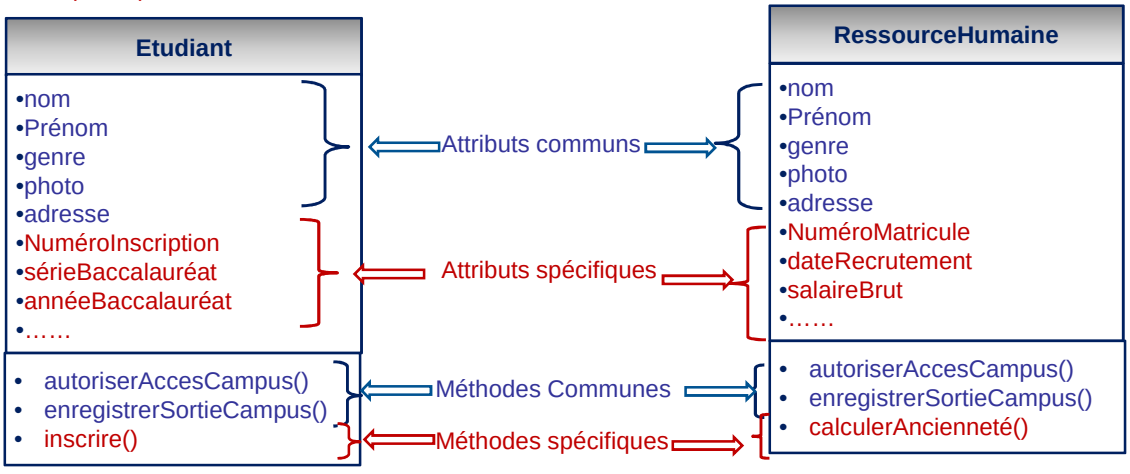
De même, le personnel administratif et enseignant, dispose d'un numéro matricule, d'une date de recrutement, d' un salaire Brut, etc.

On décide alors de créer deux nouvelles classes.

Relations: Héritage

Première approche:

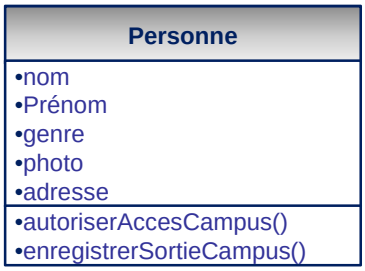
Les attributs des deux nouvelles classes comportent, en plus des attributs de la classe PERSONNE. Des attributs spécifiques*.*



Relations : Héritage

Première approche:

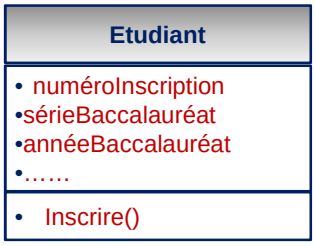
*La classe **« Personne »** ne contiendra plus que **les attributs et les méthodes communes** aux deux classes.*

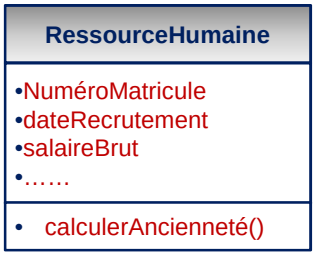


Relations : Héritage

Première approche:

Les deux classes Etudiant et Ressource Humaine contiendront les attributs et les méthodes qui leur sont spécifiques.

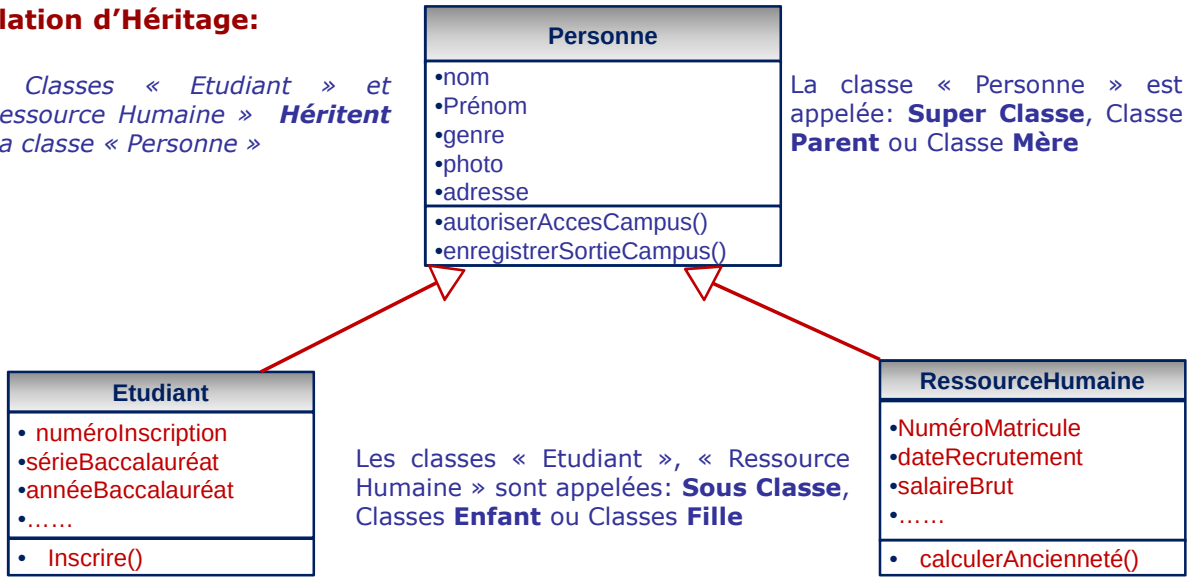




Relations : Héritage

Relation d'Héritage:

*Les Classes « Etudiant » et « Ressource Humaine » **Héritent** de la classe « Personne »*



La classe « Personne » est appelée: **Super Classe**, Classe **Parent** ou Classe **Mère**

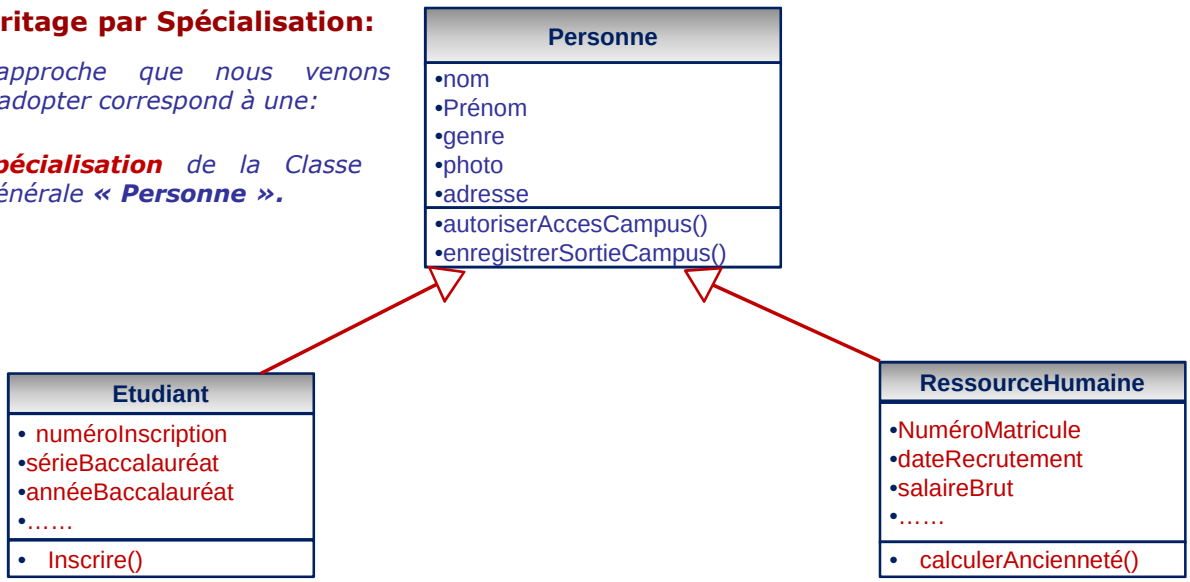
Les classes « Etudiant », « Ressource Humaine » sont appelées: **Sous Classe**, Classes **Enfant** ou Classes **Fille**

Relations: Héritage

Héritage par Spécialisation:

L'approche que nous venons d'adopter correspond à une:

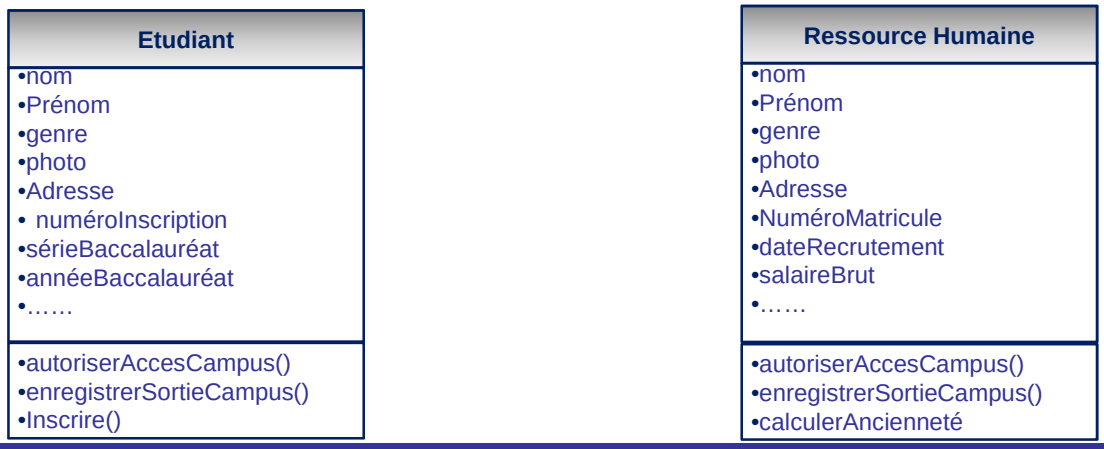
Spécialisation** de la Classe Générale **« Personne ».



Relations: Héritage

Deuxième Approche:

- ✓ *La deuxième approche applique une démarche inverse:*
- ✓ *Elle consiste à observer dès le départ que la population qui fréquente le campus est composée d'étudiants et de ressources humaines:*



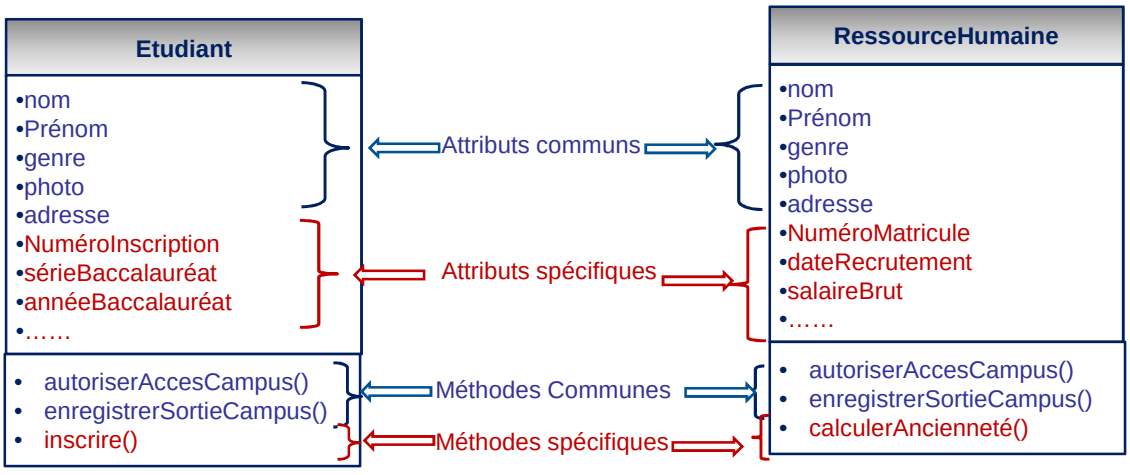
Relations: Héritage

Deuxième approche:

On constate par la suite que les deux classes ont:

*Des Attributs et des méthodes **communes***

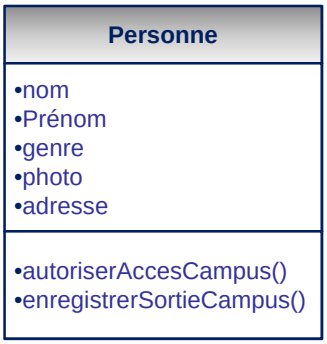
*Des Attributs et des méthodes **propres***



Relations: Héritage

Deuxième Approche:

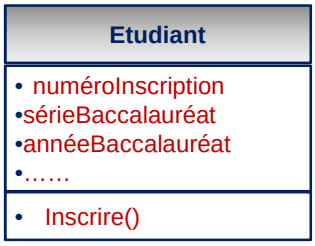
*Ce constat nous amène à créer une nouvelle classe **« Personne »** qui regroupera les attributs et les méthodes communes.*

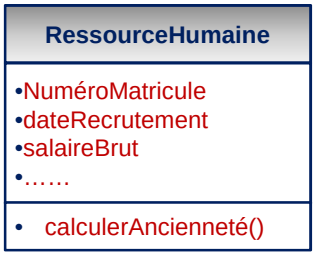


Relations : Héritage

Deuxième approche:

Et de ne conserver dans les classes « Etudiant » et « Ressource Humaine » que les attributs qui leur sont spécifiques..



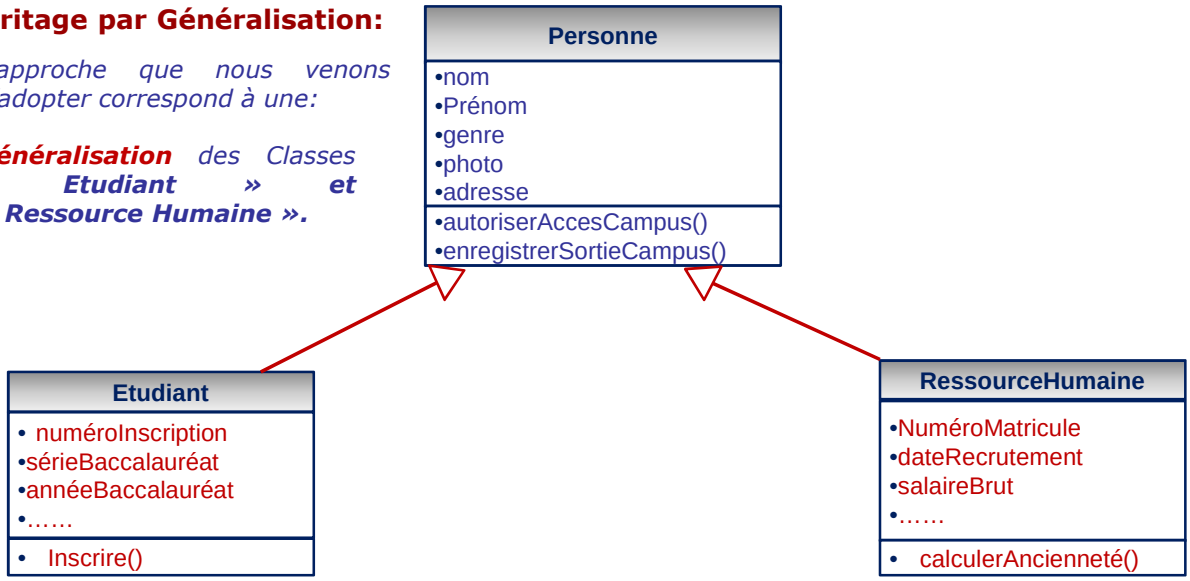


Relations: Héritage

Héritage par Généralisation:

L'approche que nous venons d'adopter correspond à une:

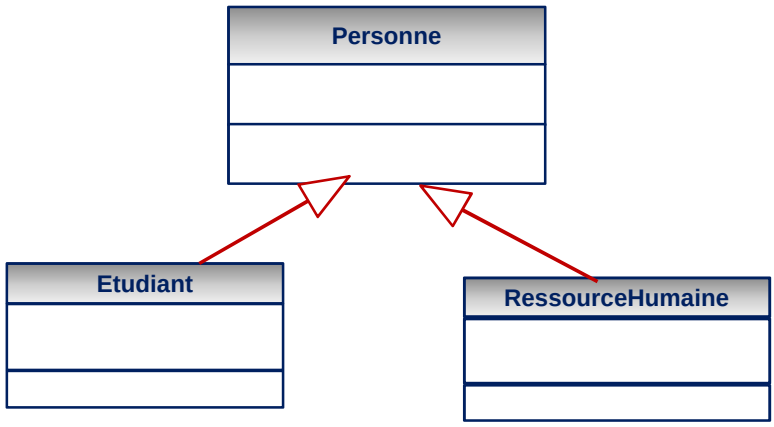
***Généralisation** des Classes « **Etudiant** » **et** « **Ressource Humaine** ».*



Relations: Héritage

Constat:

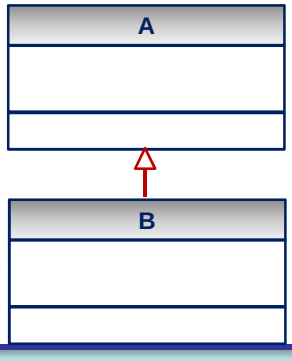
Les deux démarches aboutissent au même Modèle.



Relations: Héritage

Propriétés:

- ✓ *Une instance (Objet) de la classe B est aussi une instance (Objet) de la classe A.*
- ✓ *Les instances (Objets) de la classe B partagent les mêmes attributs et méthodes que celles de des instances (Objets) de la classe A.*
- ✓ *La classe B est une **spécialisation** de la classe A.*
- ✓ *La classe A est une **généralisation** de B.*



- ✓ *La classe A est une **Super Classe**.*
- ✓ *La classe A est une Classe **Parent**.*
- ✓ *La classe A est une Classe **Mère**.*
- ✓ *La classe B est une **Sous Classe**.*
- ✓ *La classe B est une Classe **Enfant**.*
- ✓ *La classe B est une Classe **Fille**.*

Relations: Héritage

Exercices:

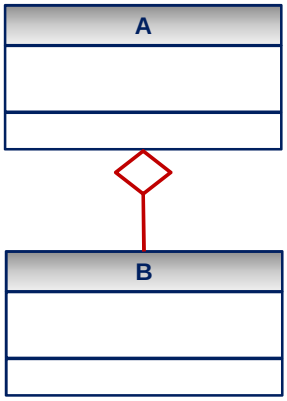
- ✓ *Modéliser le parc de véhicules d'une entreprise de location de moyens de transport.*
- ✓ *Modéliser les modes de paiement*
- ✓ *Modéliser les objets d'un Bibliothèque graphique.*
- ✓ *Modéliser le fonds documentaire d 'une Bibliothèque.*
- ✓ *Modéliser les messages échangés via un réseau social.*

Relations : Contenance (Agrégation/Composition)

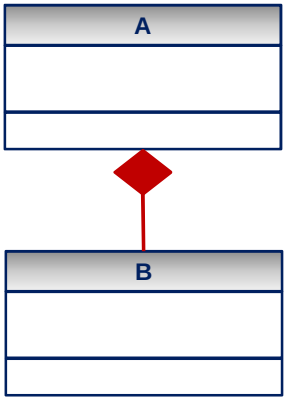
Présentation:

UML permet de représenter la relation de contenance moyennant deux types de relations:

- ✓ **Agrégation.**



- ✓ **Composition**



Relations : Contenance (Agrégation/Composition)

Présentation:

Ce type de relation exprime le fait qu'une Classe A **Contient** une autre classe B

Exemples:

- ✓ Une voiture contient (possède, est composée de):
  - Roues
  - Moteur
  - Carrosserie
  - Plaque d'immatriculation.
- ✓ Un dessin contient un ensemble de figures géométriques
- ✓ Une présentation PowerPoint est composé de Slides
- ✓ Une équipe de foot ball est composée d'un ensemble de joueurs.
- ✓ Une classe (groupe) est composée d'un certain nombre d'étudiants.

Relations : Contenance (Agrégation/Composition)

Agrégation:

L'agrégation exprime une relation de type « **avoir un** , **Possède**», une instance de A (un objet de A) peut « avoir un » une instance de B (un objet de B).

- les objets de B peuvent exister indépendamment des objets de A.
- La suppression d'une instance de A n'entraine pas nécessairement la suppression des instances de B qui lui sont rattachés.
- La même instance de B peut faire partie de (être commune à) plusieurs instances de A.

Exemple: Quel type de relation existe-t-il entre:

- Un étudiant et sa classe.
- Une voiture, son moteur et sa carrosserie.
- Un joueur et son équipe.

Relations : Contenance (Agrégation/Composition)

Représentation UML:

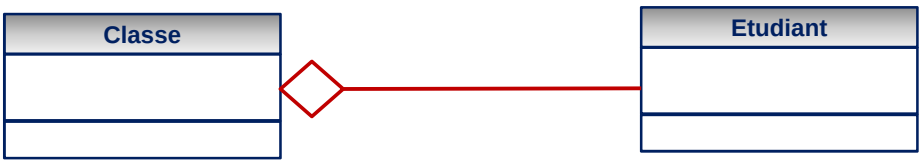
✓ Une instance de la classe « A » Contient des instances de la classe « B ».

Conteneur Agrégé Ensemble Contenu Agrégat Elément

Relations : Contenance (Agrégation/Composition)

Exemples:

- ✓ Une classe (groupe) a un certain nombre d'étudiants.



- ✓ Une association est composée de personnes (ses membres).

Relations : Contenance (Agrégation/Composition)

Composition:

La composition exprime une relation de type « **Faire partie de** », une instance de la classe B (un objet de la classe B) « Fait partie » d'une instance de la classe A (un objet de la classe A).

- les objets de B ne peuvent exister indépendamment des objets de A.
- La suppression d'une instance de A entraine la suppression des instances de B qui lui sont rattachés.
- La même instance de B ne peut faire partie de (être commune à) plusieurs instances de A.

Exemples: Quel type de relation existe-t-il entre:

- Une Voiture et sa plaque d'immatriculation
- Une voiture, son moteur et sa carrosserie.

Relations: Association

Présentation:

Les Objets du système peuvent avoir liens exprimant l'existence d'une certaine relation.

Soit deux objets du système reliés par une certaine relation

Exemple:

Dans le domaine « Scolarité » on observe les faits suivants:

L'étudiant Karim est inscrit dans la filière Ingénierie

Quels sont les objets mentionnés dans ce fait et quel lien existe-t-il entre eux ?

Relations: Association

Présentation:

Dans le domaine « Scolarité » on observe d'autres faits similaires au précédents:

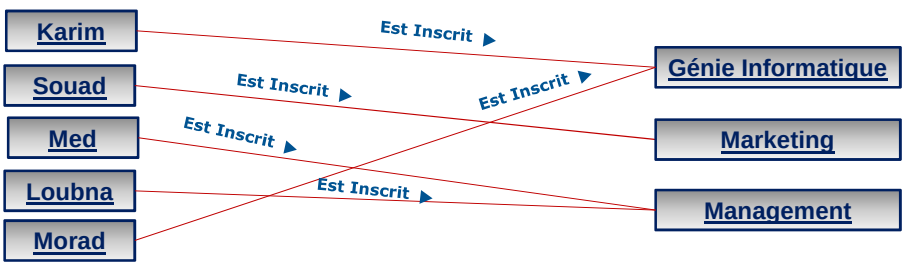
L'étudiante Souad est inscrite dans la filière Marketing

L'étudiant Mohammed est inscrite dans la filière Management

L'étudiante Loubna est inscrite dans la filière Management

L'étudiant Morad est inscrit dans la filière Génie Informatique

Représentons l'ensemble de ces faits:



Relations: Association

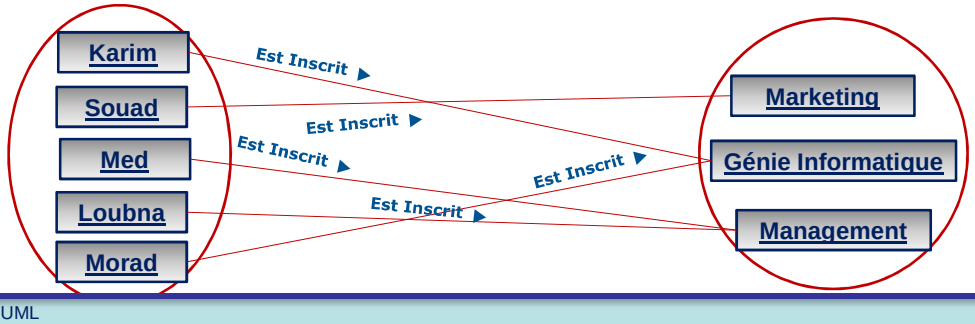
Présentation:

L'analyse de ce schéma conduit au constat suivant:

Les objets: { Karim, Souad, Med Loubna, Morad } sont semblables et peuvent être regroupés.

Les objets: { Génie Informatique, Marketing, Management }

sont également semblables et peuvent être regroupés.

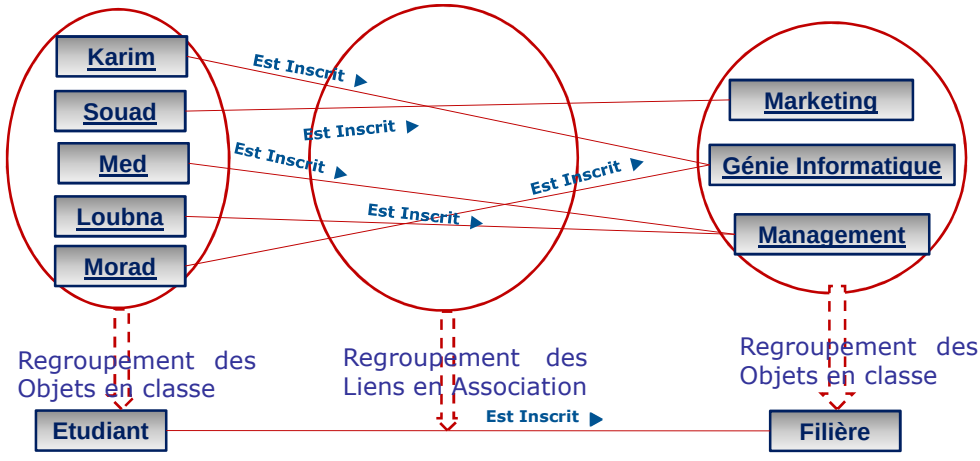


Relations: Association

Présentation:

On peut faire le même constat au niveau des liens entre les objets:

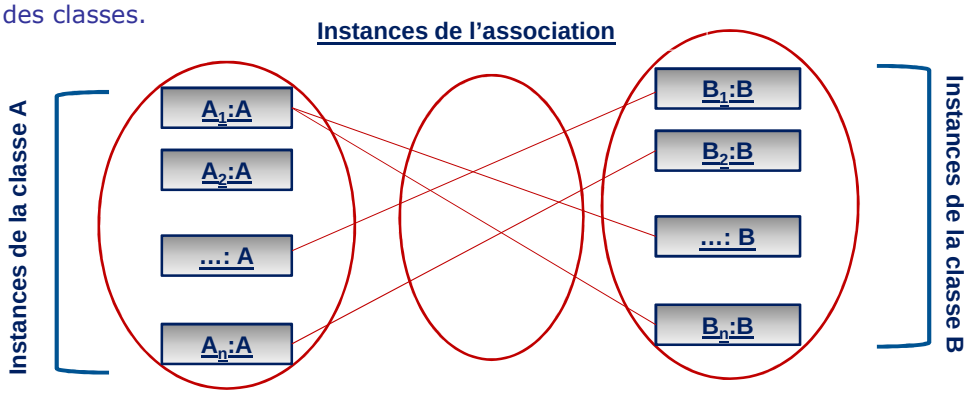
Tous les liens sont semblables et décrivent le même type de relation



Relations: Association

Présentation:

D'une manière plus générale: Une association est un ensemble de liens semblables reliant des objets, instances des classes.



Représentation ensembliste d'une association

Relations: Association

Arité des Associations: Association binaire.

Les exemples vus concernent les associations dites binaires.

Une association binaire est une association dont chaque instance relie deux objets différents.

Représentation d'une association binaire (Arité = 2)

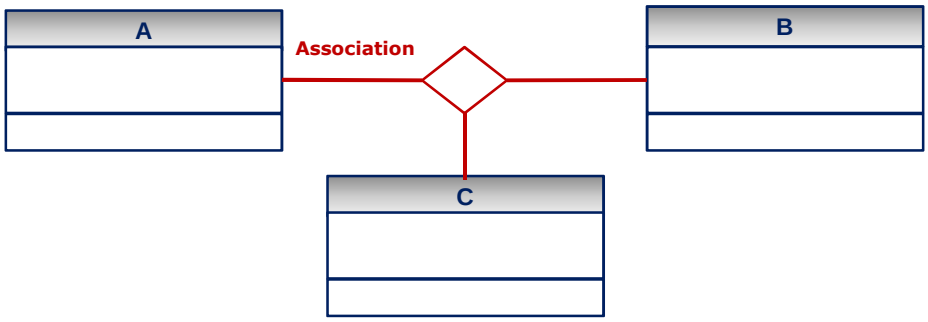
L'arité d'une association est **le nombre d'objets différents** reliés par chaque **instance de l'association.**

L'arité d'une association **n'est pas toujours égale au nombre de classes qu'elle relie.**

Relations: Association

Association ternaire:

Chaque instance de l'association relie trois objets (instances)



Représentation d'une association ternaire (reliant trois classes)

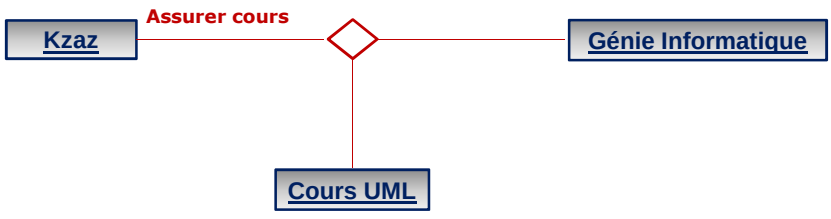
Relations: Association

Association ternaire:

Dans le domaine planification des cours on relève le fait suivant:

Le Professeur Kzaz assure le cours d'UML pour la filière Génie Informatque

Quels sont les objets mentionnés dans ce fait et quel lien existe-t-il entre eux

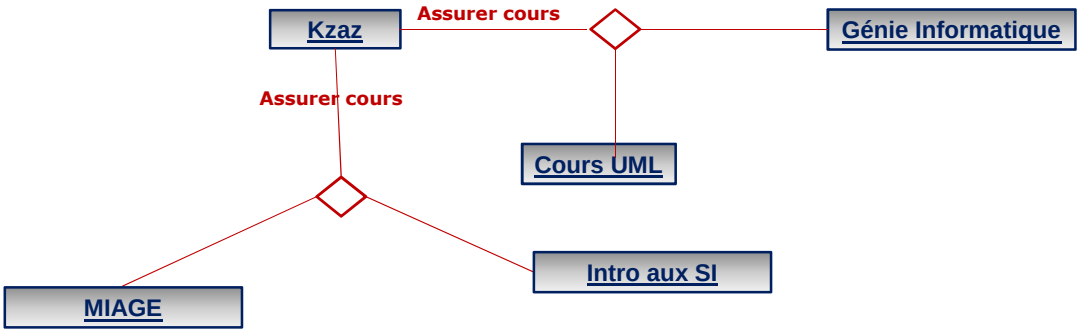


Relations: Association

Association ternaire:

Exemple; soit le fait suivant:

Le Professeur Kzaz assure aussi le cours Intro aux SI pour la filière MIAGE

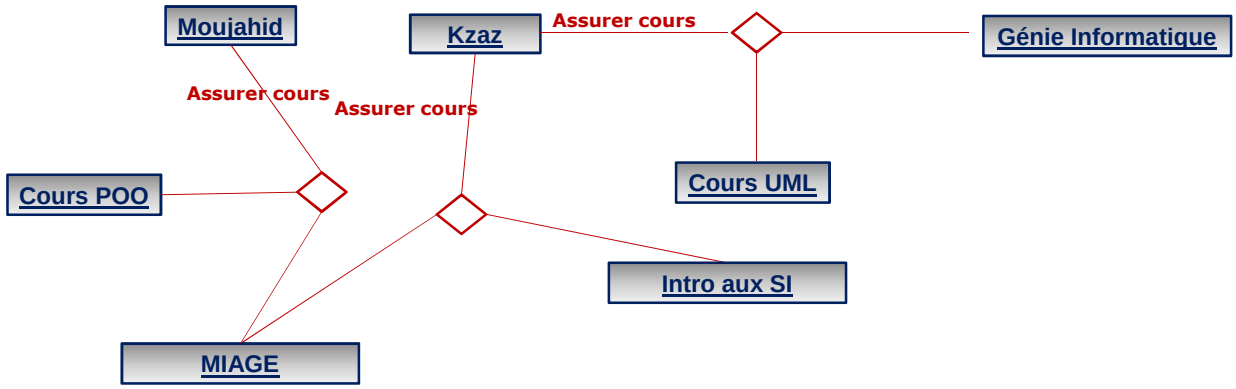


Relations: Association

Association ternaire:

Un autre fait similaire:

Le Professeur Moujahid assure le cours Programmation Orientée de la filière MIAGE



Relations: Association

Association ternaire:

Le diagramme précédent met en relation des objets qu'on peut regrouper en 3 sous- ensembles:

Les objets: { Kzaz, Moujahid } Sont semblables; ils peuvent être regroupés en une seule classe: Professeur

Les objets: { Génie Informatique, MIAGE } Sont semblables; ils peuvent être regroupés en une seule classe: Filière

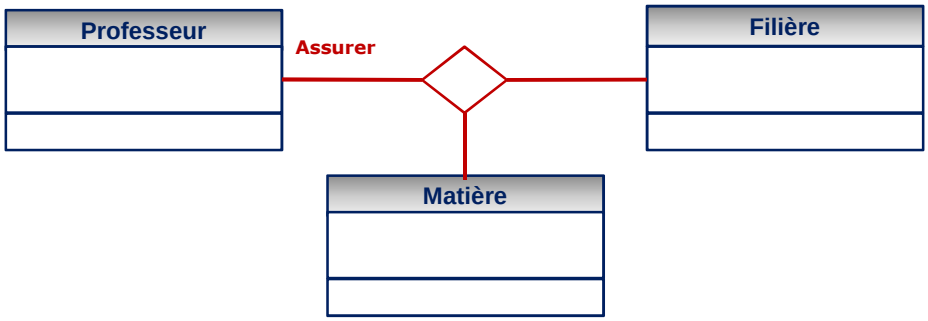
Les objets: { UML, POO } Sont semblables; ils peuvent être regroupés en une seule classe: Matière

Les objets précédents sont reliés par trois liens similaires: « Assurer Cours »; ils peuvent être regroupés en une seule association.

Relations: Association

Association ternaire:

Les diagrammes précédents donnent lieu à un seul diagramme des classes:

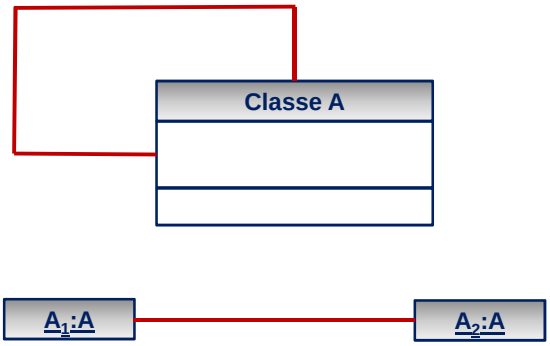


Représentation d'une association ternaire (reliant trois classes)

Relations: Association

Association réflexive:

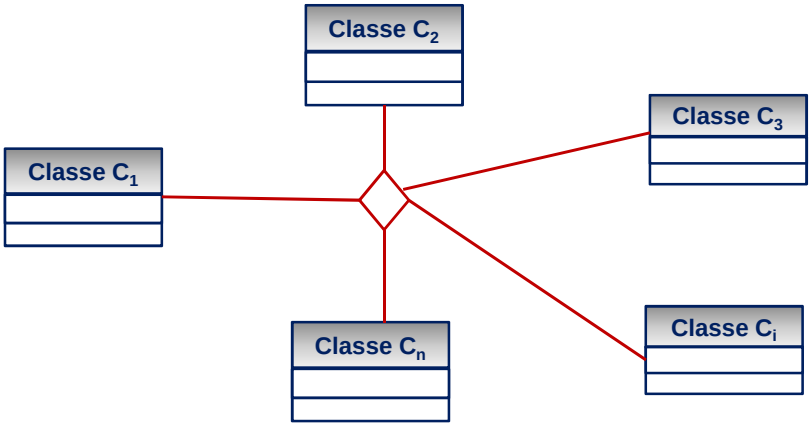
Une association réflexive relie des objets différents d'une même classe.



Deux instances de la classe A sont reliées

Relations: Association

Association n-aire:



Relations: Association

Exemples:

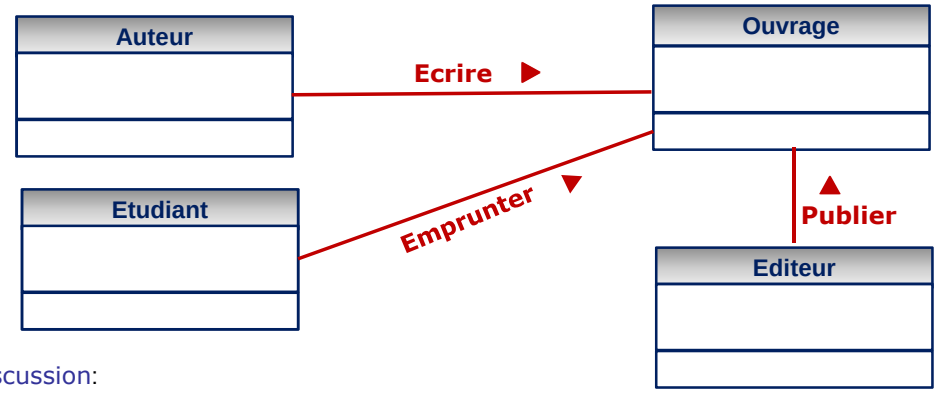
Modéliser les phrases suivantes:

- ✓ Un auteur écrit des ouvrages.
- ✓ Un éditeur publie des ouvrages.
- ✓ Un étudiant emprunte un ouvrage.
- ✓ Un professeur assure une matière pour une classe.
- ✓ Une séance de cours est assurée par un professeur pour une classe d'étudiants dans une salle pendant un certain horaire.
- ✓ Une personne propriétaire loue un appartement à une autre personne locataire.

Relations: Association

Exemples:

- ✓ Un auteur écrit des ouvrages
- ✓ Un étudiant emprunte un ouvrage
- ✓ Un éditeur publie des ouvrages.



- ✓ Discussion:

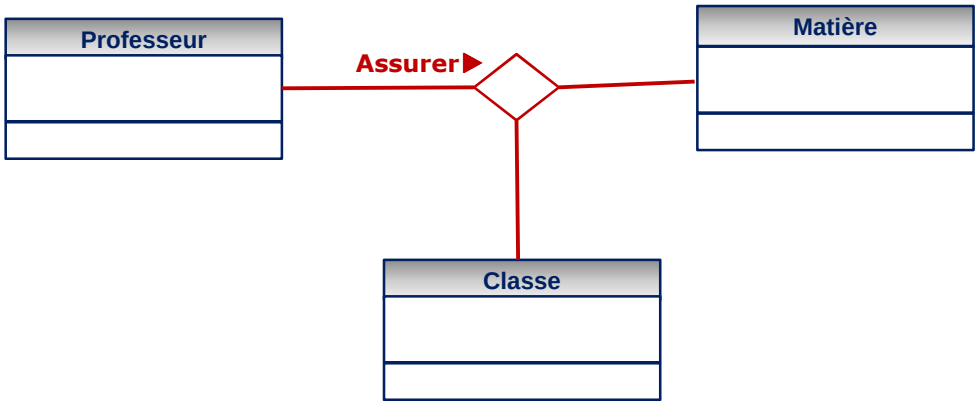
Editeur: attribut de la classe Ouvrage!

Auteur: attribut de la classe Ouvrage!

Relations: Association

Exemples:

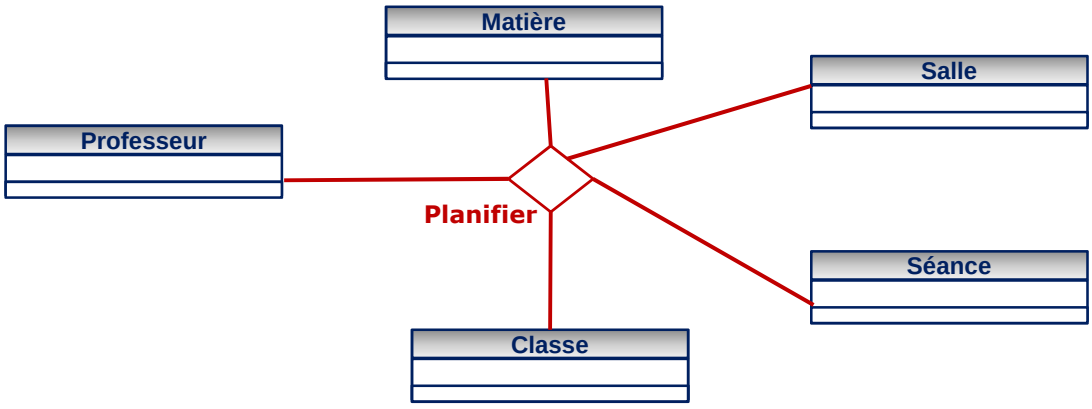
- ✓ Un Professeur assure une matière pour une classe



Relations: Association

Exemples:

- ✓ Une séance de cours est assurée par un professeur pour une classe d'étudiants dans une salle pendant un certain horaire.



Relations: Classe-Association

Exemple introductif:

✓ Un étudiant emprunte un ouvrage auprès de la bibliothèque. Lors de l'emprunt, les dates d'emprunts et de restitution sont fixées.

Soit le diagramme des classes suivant:

Placer les attributs suivants sur ce diagramme: – dateEmprunt – dateRestitution

✓ Ces deux attributs ne caractérisent ni l'étudiant ni l'ouvrage.

✓ Ils caractérisent l'emprunt. Ils ne sont définis qu'au moment de l'emprunt.

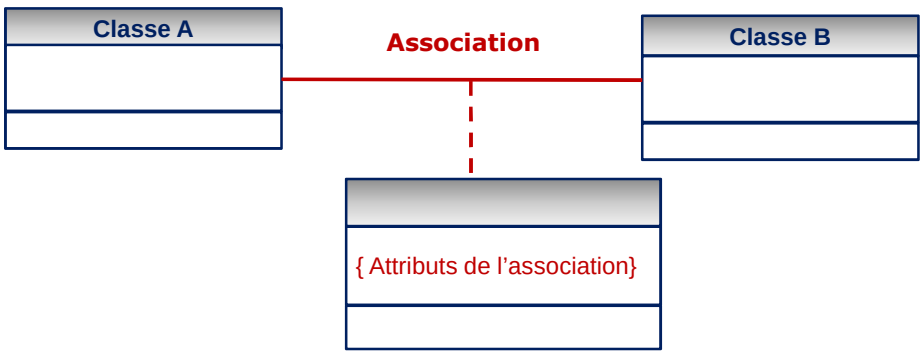
✓ Ce sont donc: des attributs de l'association « Emprunter »

des attributs de l'association « Emprunter »

Relations: Classe-Association

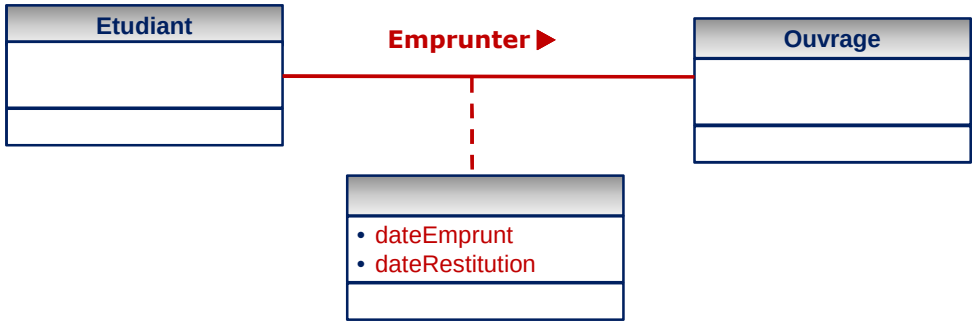
✓ Une classe-Association est une association porteuse d'attributs.

✓ Représentation des classes association.



Relations: Classe-Association

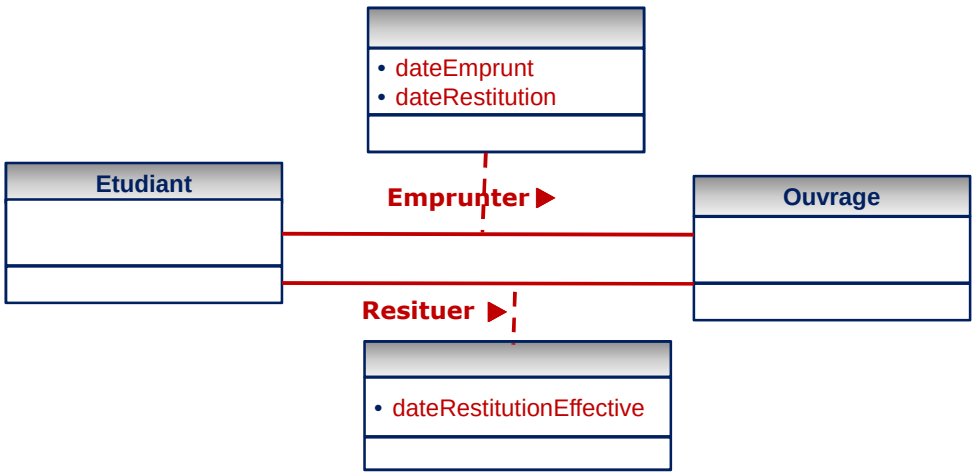
✓ Exemple:



Relations: Classe-Association

✓ Compléter le modèle pour prendre en compte la date de restitution effective de l'ouvrage.

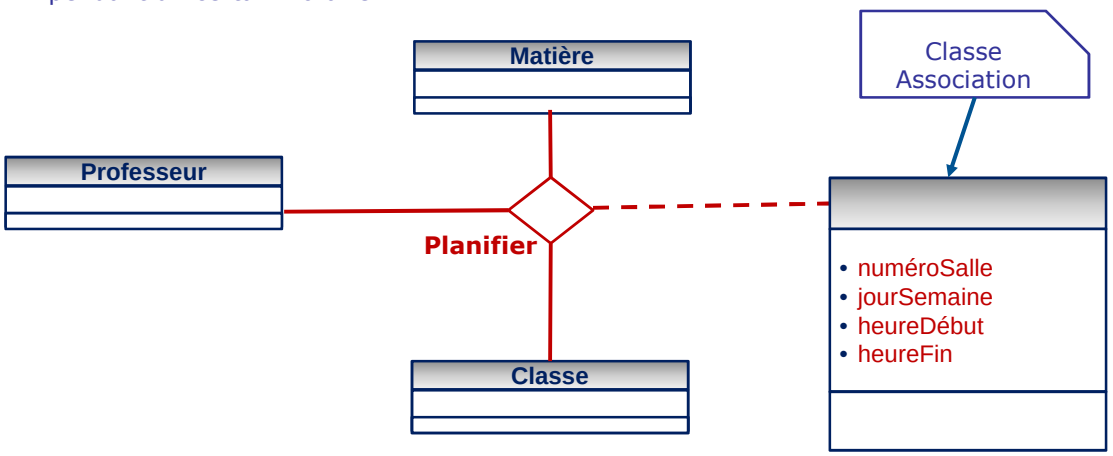
✓ La date de restitution effective peut être différente de la date de restitution réglementaire.



Relations: Classe-Association

Exemple:

✓ Une séance de cours est assurée par un professeur pour une classe d'étudiants dans une salle pendant un certain horaire.

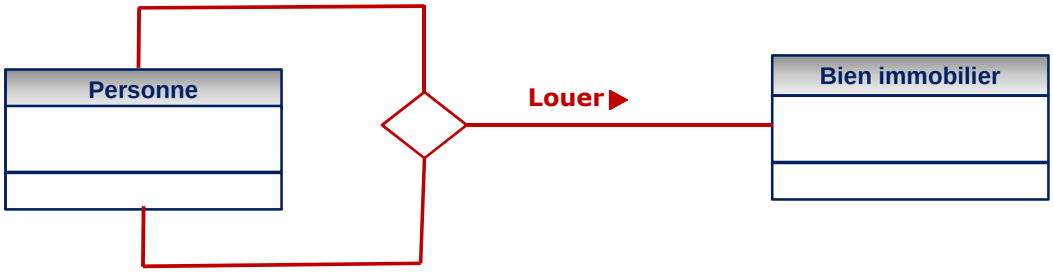


Relations: Classe-Association

Exemple:

✓ On considère le SI d'une agence immobilière. Elle reçoit de la part des propriétaires des biens à louer, et reçoit de la part des locataires des demandes de location. On souhaite modéliser le fait suivant:

Une personne, propriétaire d'un bien, loue un bien immobilier, à une autre personne, qui devient le locataire du bien.

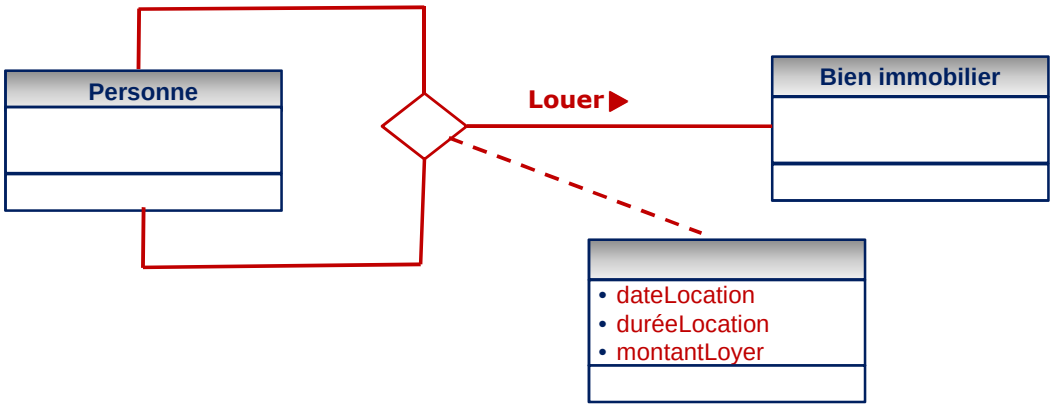


Relations: Classe-Association

Exemple:

✓ On souhaite enrichir le modèle précédent par les attributs suivants:

Date Début de location, Durée de la location, et montant du loyer.

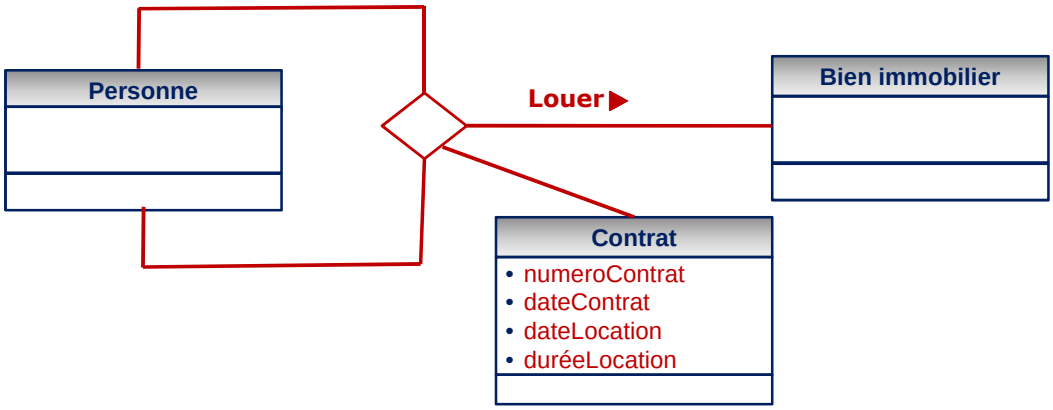


Relations: Classe-Association

Exemple:

✓ On souhaite intégrer au modèle les données sur le contrat:

Numéro de contrat, date du contrat.



Classe Abstraite

Définition:

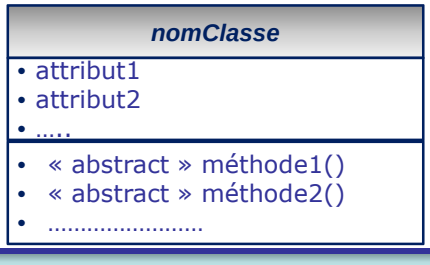
✓ Une **classe abstraite** est une classe dont l'implémentation n'est pas complète et qui ne peut être instanciée. Une classe abstraite contient des **méthodes abstraites**.
Une **méthode est dite abstraite** lorsqu'on connaît son entête, mais pas la manière dont elle peut être réalisée (i.e. on connaît sa déclaration, mais pas sa définition).

Une classe abstraite sert essentiellement à factoriser des méthodes et attributs communs à plusieurs classes, et ce dans une relation d'héritage

Une classe abstraite sert de base pour la définition d'autres classes, les sous classes.

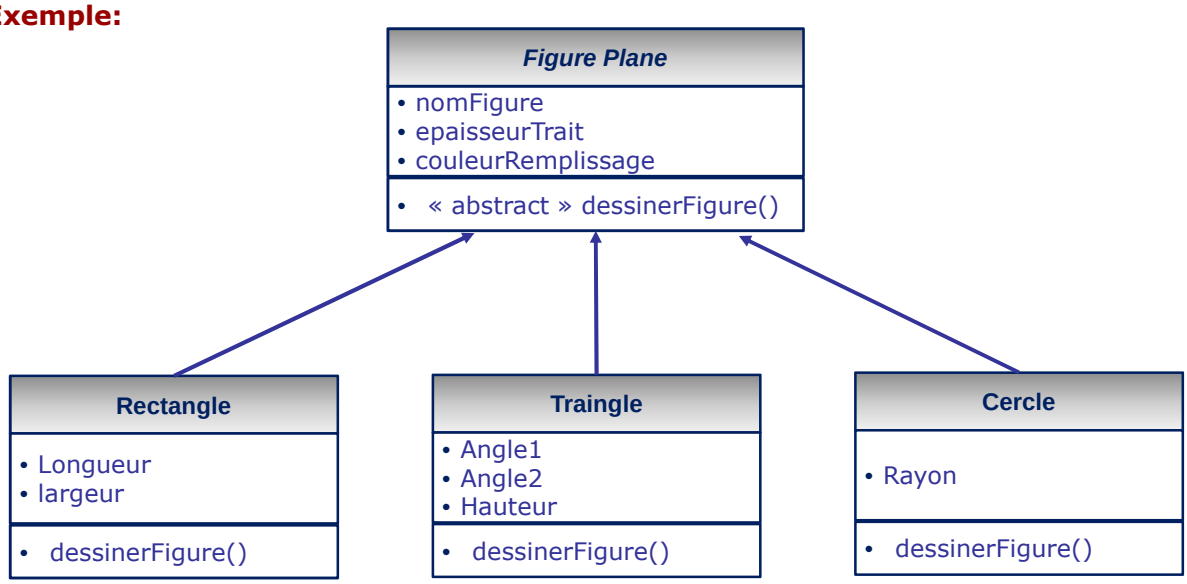
Notation: le nom d'une classe abstraite est écrit en italique

Les méthodes abstraites sont précédées du stéréotype « abstract ».



Classe Abstraite

Exemple:



Classe Interface

Définition:

✓ Une **classe interface** est une classe complètement abstraite.

Une **classe interface** ne contient pas d'attributs; elle ne contient que des méthodes abstraites, c'est à dire des méthodes non implémentées dans la classe.

Les classes interfaces sont définissent des ensembles d'opérations que d'autres classes doivent implémenter.

Une classe interface sert essentiellement à factoriser des méthodes communes à plusieurs classes, et ce dans une relation d'héritage.

Une classe interface permet de structurer et simplifier les modèles.

Notation: le nom d'une classe interface est précédé du mot « Interface ».

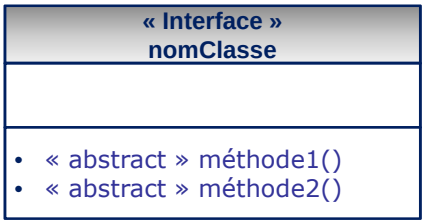


Diagramme des Classes et Diagramme d'Objets

Diagramme des classes:

✓ Un **diagramme des classes** relatif à un domaine est un schéma représentant les classes et les associations du domaine.

✓ Le diagramme des classes traduit les **règles structurelles du domaine** .

✓ Le **diagramme des classes** est une **abstraction** des **diagrammes d'objets du domaine**.

✓ Les diagrammes des classes d'un domaine permettent :

- La compréhension du domaine d'étude.
- Sa délimitation. et
- L'évaluation de la complexité du système à construire.

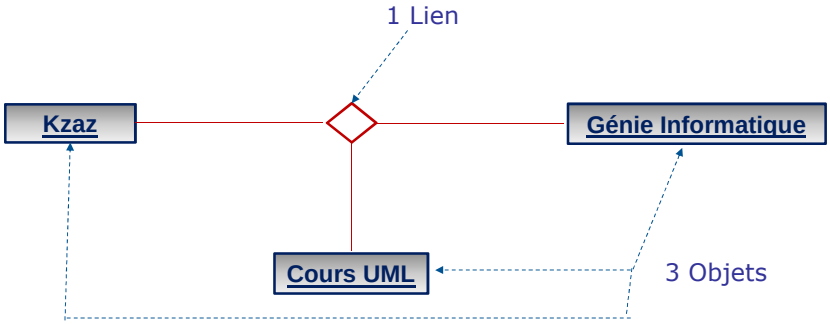
✓ Les diagrammes des classes constituent une base :

- Du contrat (cahier des charges) passé ente le maître d'ouvrage (le commanditaire du système) et le maître d'œuvre (le constructeur du système).
- Pour la conception et l'implémentation du futur système informatisé.

Diagramme des Classes et Diagramme d'Objets

Diagramme d'objets

- ✓ Un diagramme représente un fait particulier d'un domaine d'étude.
- ✓ Un diagramme d'objets représente un ensemble d'objets et les liens qui les relient.

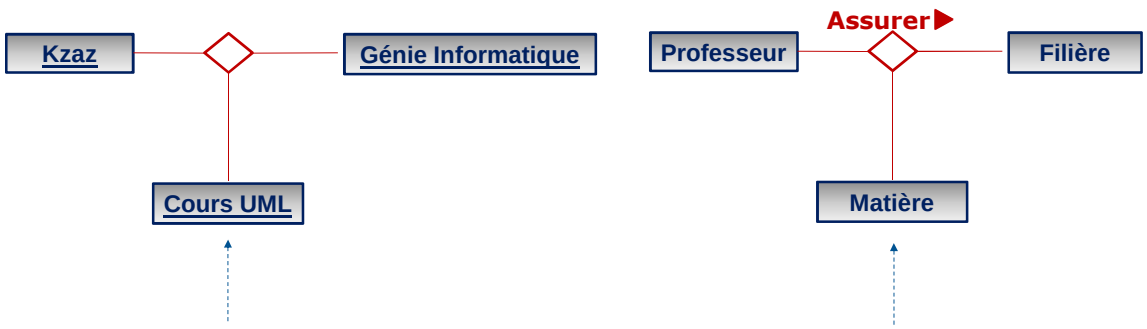


Un diagramme d'objets

Diagramme des Classes et Diagramme d'Objets

Diagramme d'objets

- ✓ Un diagramme d'objet est une instance d'un diagramme des classes.
- ✓ Un diagramme des classes est une abstraction d'un ensemble de diagrammes d'objet.



Ce diagramme d'objets est une instance de ce diagramme des classes

Diagramme des Classes et Diagramme d'Objets

Intérêt des diagrammes d'objets:

- ✓ Les diagrammes d'objets ont un intérêt très limité par rapport aux diagrammes des classes
- ✓ Ils permettent cependant de :
  - Comprendre à l'aide d'exemples concrets (Objets et liens), des situations précises de la réalité, qui semblent être complexes à modéliser directement à l'aide des diagrammes des classes.
  - Illustrer un diagramme des classes complexe et aider ainsi à la lecture de ces diagrammes.

On fera donc appel aux diagrammes d'objets uniquement dans des **situations particulières**.

Les diagrammes des Classes sont t **indispensables** pour construire le système.

Règles de Domaine

Définition:

- ✓ La modélisation d'un système consiste à fournir une représentation de ses caractéristiques considérées comme étant pertinentes par rapport à l'objectif recherché.
- ✓ Les systèmes que nous modélisons sont régis par des règles qu'il est nécessaire d'inclure dans le modèle.
- ✓ On appelle **règle de domaine**, toute règle ou contrainte qui régit la structure ou le fonctionnement du domaine modélisée.
- ✓ La politique de l'entreprise, les lois physiques, les lois civiles, la législation fiscale sont des exemples de règles de domaine.
- ✓ Lorsque le domaine étudié est un **domaine de gestion,** on parle alors de **règles de gestion** ou de **règles métier** du domaine.

Exemples:

- Règles commerciales,
- Règles à appliquer aux ressources humaines,
- Règles fiscales et comptables,
- Etc.

Règles de Gestion

Exemple:

Règles de gestion régissant le domaine «Gestion des Etudes »:

- ✓ Tout étudiant est inscrit dans une filière.
- ✓ Un étudiant ne peut être inscrit, pendant la même année, dans des filières différentes.
- ✓ Une filière est dirigée par un professeur.
- ✓ Un professeur peut n'être responsable d'aucune filière.

Règles de gestion régissant le domaine «Gestion des équipes commerciales »:

- ✓ Le marché est découpé en zones commerciales.
- ✓ Un commercial peut être responsable d'une zone.
- ✓ Les commerciaux sont répartis sur les zones.
- ✓ Les commerciaux peuvent être spécialisés dans certaines catégories de produits.

Règles de Gestion

Exemples:

Règles de gestion régissant le domaine «Bibliothèque »:

- ✓ Un lecteur ne peut pas emprunter plus de 5 ouvrages, il peut à un moment donné n'avoir emprunté aucun ouvrage.
- ✓ Un ouvrage est écrit par au moins un auteur, il peut être écrit par plusieurs auteurs.
- ✓ Un auteur écrit au moins un ouvrage.
- ✓ Un ouvrage est publié par un éditeur unique.

Règles de Gestion

Expression des règles de gestion:

Les règles de gestion s'expriment sur les diagrammes des classes à l'aide des concepts suivants:

- **Valeurs de Multiplicité**.
- Contraintes sur les associations: **Inclusion et Exclusion**
- **Dépendances fonctionnelles** entre les attributs et leur répartition sur les classes et classes-associations.

Valeurs de Multiplicité

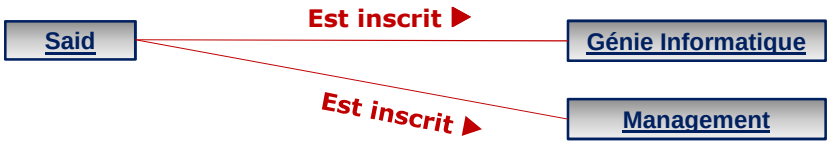
Définition:

- ✓ Les valeurs de multiplicité permettent d'exprimer certaines règles de gestion du domaine modélisé.
- ✓ Elles expriment des contraintes (règles) qui régissent les LIENS entre les OBJETS du domaine.
- ✓ Elles sont portées par les extrémités des ASSOCIATIONS reliant les CLASSES du domaine.

Valeurs de Multiplicité

Exemple:

- ✓ Soit la règle de gestion :Tout étudiant est inscrit dans une filière unique.
- ✓ Le diagramme d'objets ci-après respecte-t-il cette RG?

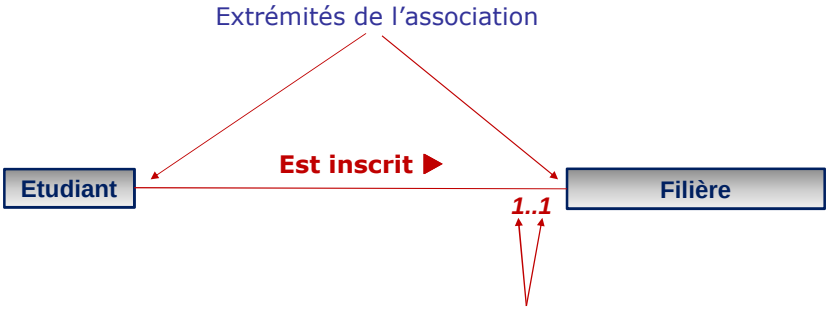


- ✓ Le système ne doit pas donc tolérer l'existence d'un telle situation.

Valeurs de Multiplicité

Représentation:

- ✓ Les valeurs de multiplicité sont portées par les extrémités des associations entre les classes.

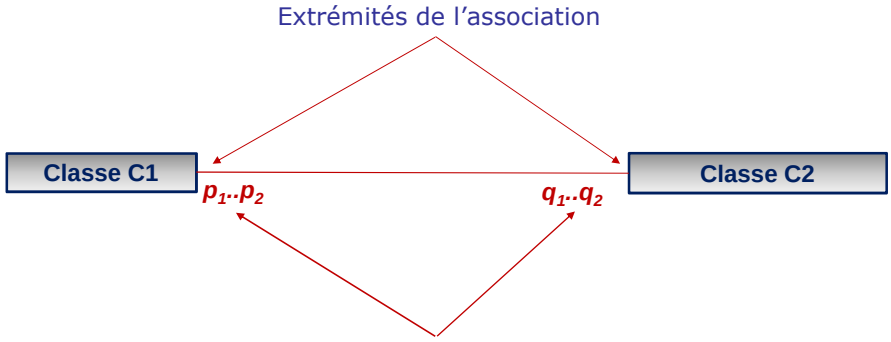


Le couple de valeurs de multiplicité traduisant la règle de gestion précédente.

Valeurs de Multiplicité

Représentation:

✓ A chaque extrémité de l'association est affecté un couple de valeurs de multiplicité.



2 couples de valeurs sont associée aux 2 extrémités de l'Association

Valeurs de Multiplicité

Interprétation et lecture:

✓ Cas des associations binaires.

✓ Chaque objet instance de la classe C1, est lié à travers l'association à :

- Au moins q1 objets instances de la classe C2.
- Au plus q2 objets instances de la classe C2.

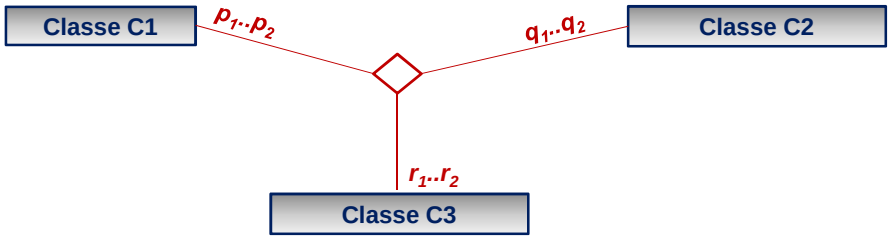
✓ Chaque objet instance de la classe C2, est lié à travers l'association à :

- Au moins p1 objets instances de la classe C1.
- Au plus p2 objets instances de la classe C1.

Valeurs de Multiplicité

Interprétation et lecture:

- ✓ Cas des associations ternaires.
- ✓ A chacune des 3 extrémités de l'association est affecté un couple de valeurs de multiplicité.



- ✓ Chaque couple d'objets instances des classes C1 et C2 , est lié à travers l'association à :
  - Au moins r1 objets instances de la classe C 3.
  - Au plus r2 objets instances de la classe C3.

Valeurs de Multiplicité

Valeurs usuelles:

p = 0

- Un objet de la classe C1 peut n'être en relation avec aucun objet de la classe C2.
- Des objets de la classe C1 peuvent exister sans pour autant être reliés avec des objets de la classe C2.
- La participation des objets de la classe C1 à la relation est **FACULTATIVE**.

Valeurs de Multiplicité

Valeurs usuelles:

p = 1

- **TOUT** objet de la classe C1 est relié à **AU MOINS UN** objet de la classe C2.
- Des objets de la classe C1 ne peuvent exister sans être reliés à des objets de la classe C2.
- La participation des objets de la classe C1 à la relation est **OBLIGATOIRE**.

Valeurs de Multiplicité

Valeurs usuelles:

q = 1

- **TOUT** objet de la classe C1 est relié à **AU PLUS UN** objet de la classe C2.
- Tout objet de la classe C1 ne peut être relié à plus d'un objet de la classe C2.
- Le nombre de liens que peut avoir un objet de la classe C1 est limité à **UN**.

Valeurs de Multiplicité

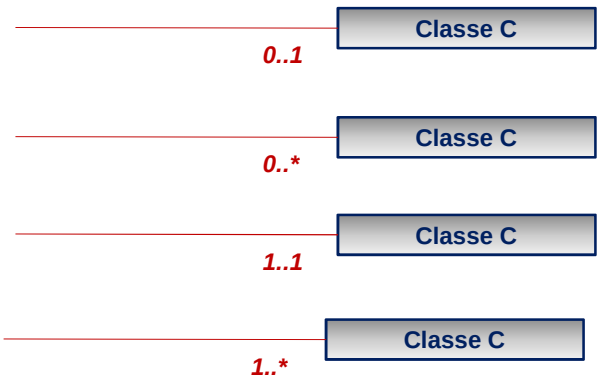
Valeurs usuelles:

*q = **

- **TOUT** objet de la classe C1 peut être relié à **PLUSIEURS** objets de la classe C2.
- Tout objet de la classe C1 peut être relié à un nombre quelconque d'objets de la classe C2.
- Le nombre de liens que peut avoir un objet de la classe C1 est illimité.

Valeurs de Multiplicité

4 Couples de valeurs usuelles:



Zéro à 1

Zéro à Plusieurs

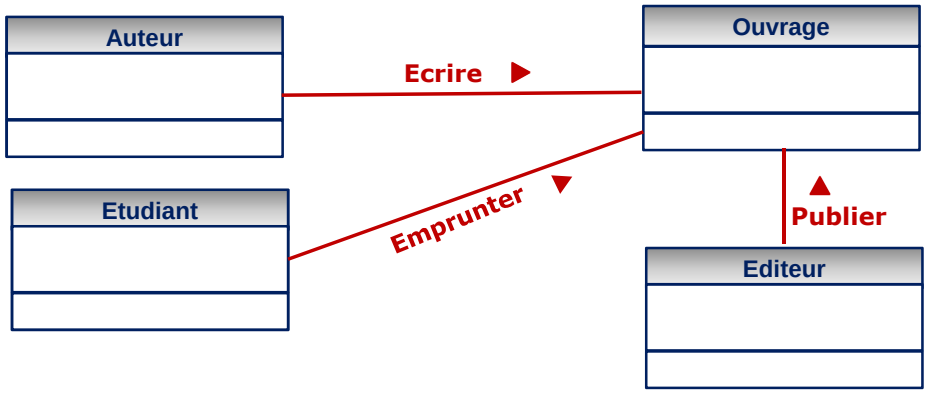
Un à un

Un à Plusieurs

Valeurs de Multiplicité

Applications:

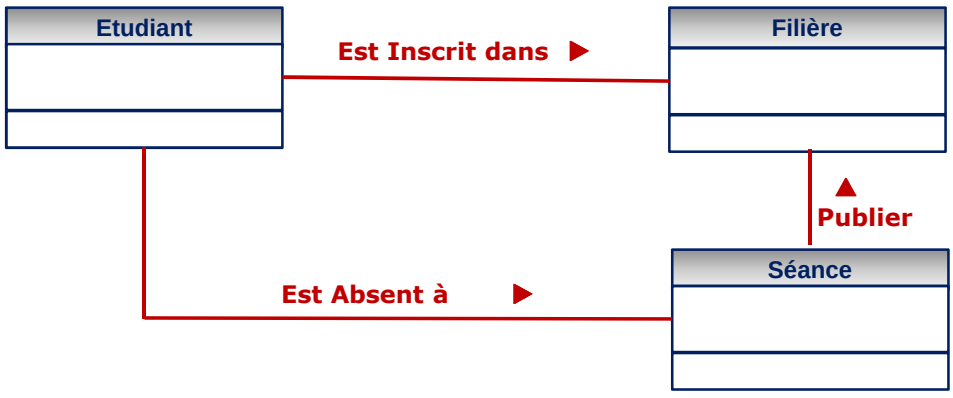
✓ Déterminer les valeurs de multiplicité des diagrammes suivants:



Valeurs de Multiplicité

Applications:

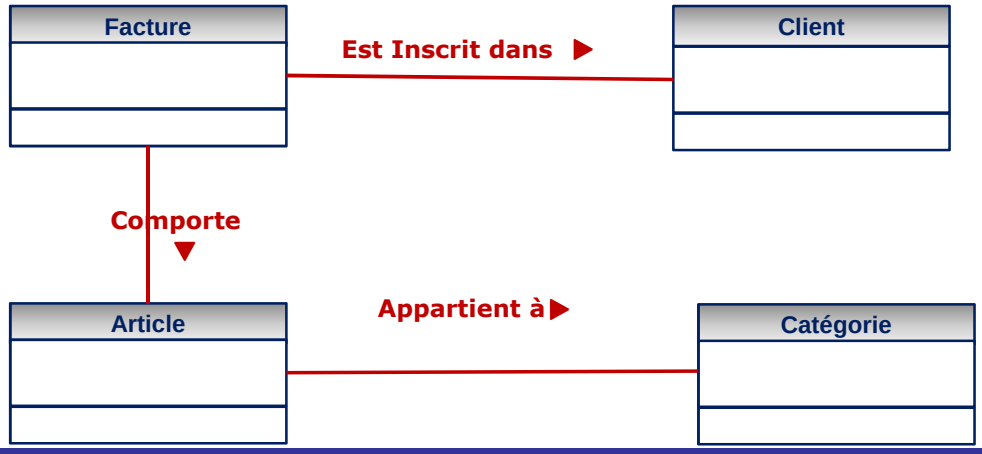
✓ Déterminer les valeurs de multiplicité des diagrammes suivants:



Valeurs de Multiplicité

Applications:

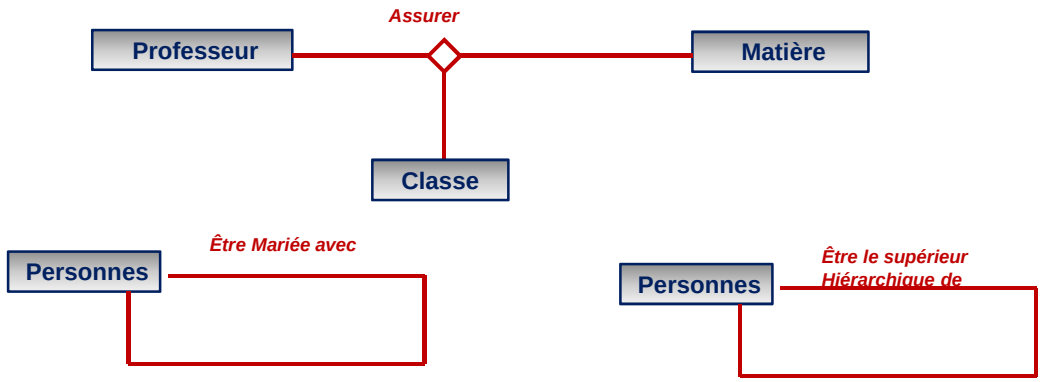
✓ Déterminer les valeurs de multiplicité des diagrammes suivants:



Valeurs de Multiplicité

Applications:

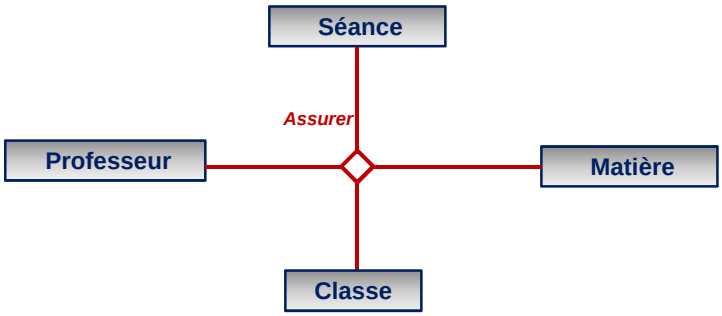
✓ Déterminer les valeurs de multiplicité des diagrammes suivants:



Valeurs de Multiplicité

Applications:

✓ Déterminer les valeurs de multiplicité des diagrammes suivants:



Contraintes sur les associations: Inclusion

Exemple introductif:

Soit la règle de gestion régissant les professeurs et les départements :

✓ Un chef de département est nécessairement membre du département dont il est le chef.

Soient la règle de gestion suivante régissant le contrats d'assurnace :

✓ Tout individu qui contracte un contrat d'assurance est nécessairement assuré.

Contraintes sur les associations: Inclusion

Exemple introductif:

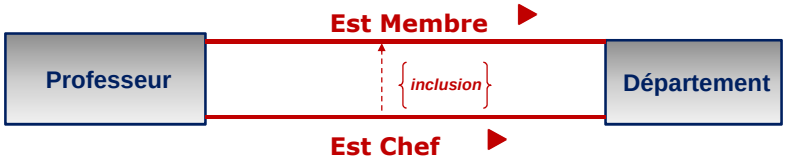
- ✓ Soit D_i une instance de la classe Département, et P_j une instance de la classe Professeur:
- ✓ La règle de gestion:

  Un chef de département est nécessairement membre du département dont il est le chef.

Implique que chaque couple d'objets (D_i, P_j) reliés par un lien de l'association « Être chef », est relié par un lien de l'association « Être membre ».

On dit que l'association « Être Chef » est incluse dans l'association « Être membre ».

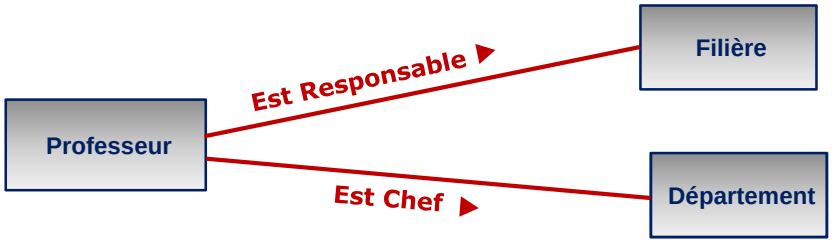
La contrainte d'inclusion n'est applicable qu'aux associations reliant les mêmes classes.



Contraintes sur les associations: Exclusion

Exemple introductif:

- ✓ Un professeur ne peut être à la fois responsable d'une filière et chef de département.



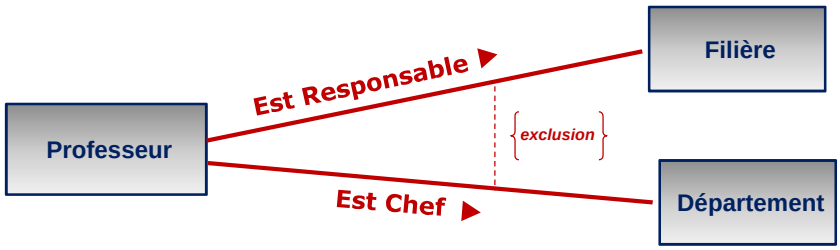
- ✓ Soit P_i une instance de la classe Professeur, Dj une instance de la classe Département et F_k une instance de la classe Filière:
- ✓ La règle de gestion énoncée:

Implique que si un objet P_i est relié par un lien de l'association « Être chef » à une instance D_j, il ne peut alors être relié par un deuxième lien de l'association « Diriger» à aucune instance F_k, et réciproquement.

Contraintes sur les associations: Exclusion

Exemple introductif:

✓ Un professeur ne peut être à la fois responsable d'une filière et chef de département.



✓ On dit que les associations « Être Chef » et « Etre Responsable » sont exclusives.

✓ La contrainte d'exclusion peut porter sur des associations reliant différentes classes.

Dépendances Fonctionnelles

Introduction:

✓ La dépendance fonctionnelle est une notion qui permet d'exprimer des contraintes qui portent sur :

- Les attributs
- Les classes et les associations

✓ Les dépendances fonctionnelles sont très utilisées en bases de données pour:

- La conception systématique du schéma de la base.
- La normalisation du schéma de la base.

Caractéristiques des attributs

Concept d'attribut:

- ✓ Un attribut est une information pertinente sur les objets d'une classe, ou bien sur les liens d'une association .
- ✓ Le suivi d'un objet et de son évolution, durant sa vie, se fait à travers la connaissance des valeurs de ses attributs.
- ✓ D'où la nécessité de mémoriser ses valeurs dans la future base de données du système.
- ✓ Un attribut possède un nom et un type.

Exemples :

- Numéro d'un Vol
- Nom d'un aéroport
- Nom d'un Passager.
- Date de départ d'un vol.
- Heure de départ d'un vol.
- État d'une réservation.

Caractéristiques des attributs

Types d'attribut:

- ✓ Le type d'un attribut délimite l'ensemble dans le quel il peut prendre ses valeurs.
- ✓ Exemples de types simples :

  Numérique, date, monétaire, logique, texte etc.

- ✓ Exemples d'attributs:

  Date élection : date.

  Coefficient matière : Numérique entier.

  Salaire perçu : Monétaire.

  Adresse : Texte.

  Admis : Logique.

Caractéristiques des attributs

Caractéristiques des attributs:

✓ Un attribut peut être :

- Atomique.
- Élémentaire.
- Composé.
- Calculé (Dérivé).

Caractéristiques des attributs

Attributs atomiques:

✓ Un attribut a est dit atomique s'il ne peut pas s'écrire sous la forme:

$$a_1 + a_2 + \ldots + a_n$$

Avec:

- a_1 , a_2 , …., a_i ,… a_n des attributs significatifs.
- L'opérateur « + » est l'opérateur de Concaténation (juxtaposition des chaînes de caractères).

✓ Autrement dit:

un attribut est dit atomique s'il ne peut pas être obtenu par juxtaposition d'autres attributs.

Caractéristiques des attributs

Attributs atomiques:

✓ Exemples:

- Montant d'une facture. — atomique
- Coefficient d'une matière. — atomique
- Nom d'une personne. — atomique
- Le relevé d'identité bancaire. — Non atomique

RIB = code banque + code localité + numéro de compte principal + Clé RIB

✓ Qu'en est-il de l'atomicité des attributs suivants :

- Adresse d'un client.
- Date naissance .

Caractéristiques des attributs

Attributs Composés:

✓ Un attribut a est dit composé s'il peut s'écrire sous la forme :

$$a_1 + a_2 + \ldots + a_n$$

Avec:

- a_1 , a_2 , ..., a_i ,... a_n des attributs significatifs.
- L'opérateur « + » est l'opérateur de Concaténation (juxtaposition des chaînes de caractères).

✓ Autrement dit:

un attribut est dit composé s'il peut être obtenu par juxtaposition d'autres attributs.

✓ un attribut non atomique est composé.

Caractéristiques des attributs

Attributs Elémentaires:

✓ Un attribut a est dit élémentaire si ses valeurs ne peuvent pas être obtenues (calculées) à partir des valeurs d'autres attributs.

✓ Exemples:

- Coefficient d'une matière.
- Note obtenue par un étudiant dans un contrôle.
- Tarif d'un article.
- Quantité vendue d'un article à un client.
- Montant du versement effectué par un client sur son compte.

Caractéristiques des attributs

Attributs Calculés (Dérivés):

✓ Un attribut a est dit élémentaire si ses valeurs ne peuvent pas être obtenues (calculées) à partir des valeurs d'autres attributs.

✓ Exemples:

- Moyenne d'un étudiant.
- Classement d'un étudiant.
- Montant d'une facture.
- Nombre d'articles facturés.
- Solde d'un compte.

Caractéristiques des attributs

Propriétés des attributs:

- Tout attribut composé est calculé.
- Tout attribut élémentaire est atomique.

Dépendances Fonctionnelles entre attributs

But des Dépendances Fonctionnelles:

- ✓ Représenter certaines contraintes issues des règles de gestion du domaine.
- ✓ Les df constituent un moyen d'expression de certaines règles qui régissent le domaine modélisé.
- ✓ Elles constituent un outil formel pour la répartition des attributs entre les classes et classes-associations.

Dépendances Fonctionnelles entre attributs

Définition:

✓ On dit qu'une attribut ***b*** dépend fonctionnellement d'un autre attribut ***a***, si et seulement si:

A chaque valeur de ***a*** il lui correspond au plus une valeur de ***b***.

Notation: ***a*** ⟶ ***b***.

✓ Exemples:

Numéro Inscription ⟶ ***Nom élève***

Référence Article ⟶ ***Tarif de Vente***

Numéro de Compte ⟶ ***Solde***

Dépendances Fonctionnelles entre attributs

Applications:

Numéro Inscription $\xrightarrow{?}$ ***Note obtenue dans une matière***

Numéro facture $\xrightarrow{?}$ ***Numéro Client***

Référence Article $\xrightarrow{?}$ ***Quantité Commandée***

Dépendances Fonctionnelles entre attributs

Dépendances élémentaires:

✓ On dit qu'une attribut ***b*** dépend fonctionnellement d'un autre attribut ***a***, par une dépendace fonctionnelle élémentaire si et seulement si:

- A chaque valeur de ***a*** il lui correspond au plus une valeur de ***b***.
- Et si ***b*** ne dépend d'aucune autre partie de ***a***.

Notation: ***a*** $\xrightarrow{dfe}$ ***b***.

✓ Exemples:

Numéro Inscription + Code Matière $\xrightarrow{?}$ ***Nom élève***

Référence Article + N° Commande $\xrightarrow{?}$ ***Tarif de Vente***

Numéro de Compte + Nom Client $\xrightarrow{?}$ ***Solde***

Adresse Client $\xrightarrow{?}$ ***Ville Client***

Dépendances Fonctionnelles entre attributs

Dépendances Elémentaires Directes:

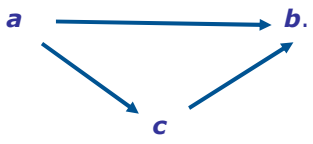
✓ On dit qu'une attribut ***b*** dépend fonctionnellement d'un autre attribut ***a***, par une dépendance fonctionnelle élémentaire directe si et seulement si:

- ***b*** est en dépendance fonctionnelle élémentaire avec ***a***.
- Et s'il n'existe aucun autre attribut ***c*** , tel que

c dépend de ***a*** et et ***b*** dépend de ***c***

Notation: ***a*** $\xrightarrow{dfd}$ ***b***.

✓ Autrement dit, il n'existe aucun autre attribut c tel que :



Dépendances Fonctionnelles entre attributs

Dépendances Elémentaires Directes:

✓ Exemple: Considérons la règle commerciale suivante:

Le taux de remise accordée par une entreprise à ses clients dépend de leur catégorie.

La df suivante est-elle directe ?

Numéro Client → ***Taux de Remise***

Non car il existe un attribut Code Catégorie tels que :

Dépendances Fonctionnelles entre attributs

Dépendances Elémentaires Directes:

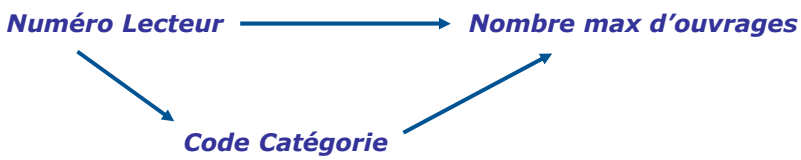
✓ Exemple: Considérons la règle suivante issue du domaine gestion des emprunts dans une bibliothèque :

Le nombre maximum d'ouvrages qu'un lecteur peut emprunter est fonction de sa catégorie

La df suivante est-elle directe ?

Numéro Lecteur → ***Nombre max d'ouvrages***

Non car il existe un attribut Code Catégorie tels que :



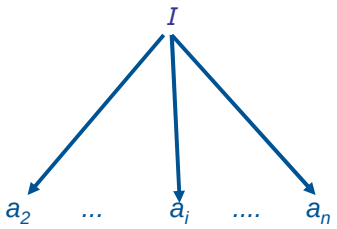
Identifiants

Identifiant d'une classe:

✓ Définition: L'Identifiant d'une classe est un attribut, ou bien un groupe d'attributs tel que :

Tous les autres attributs de la classe sont en dépendance fonctionnelle élémentaire directe avec l'identifiant.

Classe
- ***I***   - *a2*   - .....   - *an*



Identifiants

Identifiant d'une classe:

✓ Conséquences:

- A chaque valeur de l'identifiant correspond au plus une seule instance (objet) de la classe
- Deux instances (objets) distinctes d'une même classe ont nécessairement deux valeurs distinctes de l'identifiant.

✓ Exemples:

- Le ***numéro d'inscription*** est un identifiant de la classe **« Élèves ».**
- Le **numéro d'un lecteur** est un identifiant de la classe « **Lecteur** ».
- Le **code d'une matière** est un identifiant de la classe « **Matières** ».
- **L'intitulé d'une matière** ne peut être est un identifiant de la classe « **Matières** ».
- **La référence d'un article** est un identifiant de la classe « **articles** ».

Proposer un identifiant pour la classe « **Ouvrages** » du domaine bibliothèque.

Règles de Vérification du modèle

But: Obtenir un modèle "correct" « normalisé ».

1. Tous les attributs du modèle doivent être atomiques
2. Chaque classe du modèle doit posséder un identifiant.
3. Pour chaque objet, instance d'une classe, chacun des attributs doit avoir une valeur et une seule
4. Un attribut ne peut appartenir qu'à une seule classe ou classe-association.
5. Il ne doit pas y avoir de dépendance fonctionnelle transitive entre les attributs d'une classe.
6. Les attributs d'une classe-association doivent être en dépendance fonctionnelle élémentaire avec l'identifiant de la classe association.

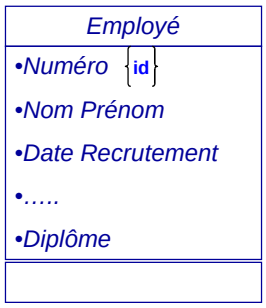
Règles de Vérification du modèle

Règle n° 3:

- ✓ Pour chaque objet, instance d'une classe, chacun des attributs doit avoir une valeur et une seule

Soit la règle de gestion suivante, issue du domaine Gestion des Ressources Humaines :

Un employé peut ne pas avoir de diplôme, comme il peut en avoir plusieurs.

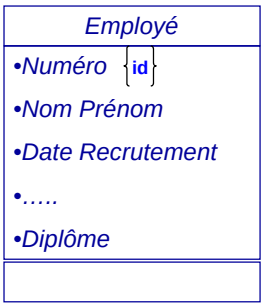


La classe « Employé » respecte-t-elle la règle n° 3?

Règles de Vérification du modèle

Règle n° 3:

La classe « Employé » respecte-t-elle la règle n° 3?



Non le modèle ne respecte pas la règle n° 3:

L'attribut « Diplôme » peut ne pas avoir de valeur pour certaines instances de la classe : Celles qui représentent les employés sans diplôme.

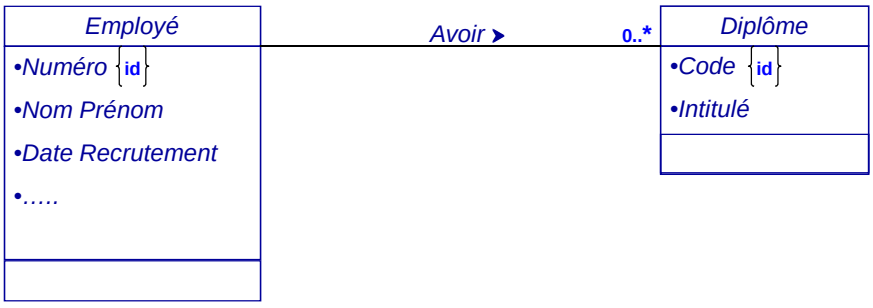
L'attribut « Diplôme » peut avoir plusieurs valeurs pour certaines instances de la classe : Celles qui représentent les employés ayant plusieurs diplômes.

Corriger le modèle.

Règles de Vérification du modèle

Règle n° 3:

Le modèle corrigé est :



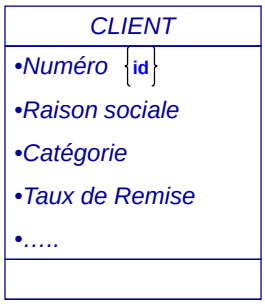
Règles de Vérification du modèle

Règle n° 5:

✓ Il ne doit pas y avoir de dépendance fonctionnelle transitive entre les attributs d'une classe.

Soit la règle de gestion suivante, issue du domaine Gestion commerciale :

Les clients sont répartis en catégories, le taux de remise accordé à un client lors d'une vente dépend exclusivement de sa catégorie.



La classe « Client » respecte-t-elle la règle n° 5 ?

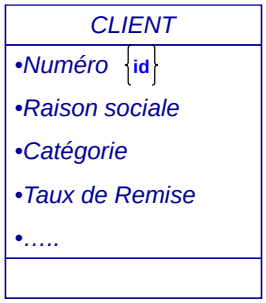
Règles de Vérification du modèle

Règle n° 5:

✓ Il ne doit pas y avoir de dépendance fonctionnelle transitive entre les attributs d'une classe.

La classe « Client » respecte-t-elle la règle n° 5 ?

Non, la règle n° 5 n'est pas respectée :



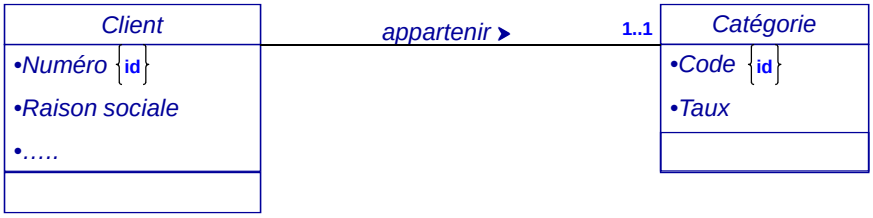
Corriger le modèle.

Règles de Vérification du modèle

Règle n° 5:

✓ Il ne doit pas y avoir de dépendance fonctionnelle transitive entre les attributs d'une classe.

Le modèle corrigé :



Réfléchir aux situations suivantes:

L'entreprise vient de modifier le taux de remise pour une certaine catégorie.

L'unique client d'une certaine catégorie ne fait plus partie de nos clients.

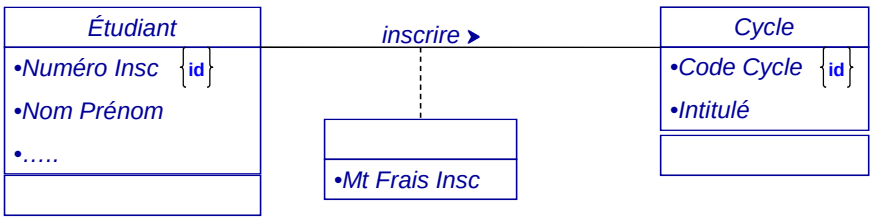
Règles de Vérification du modèle

Règle n° 6:

✓ Les attributs d'une classe-association doivent être en dépendance fonctionnelle élémentaire avec l'identifiant de la classe association.

Soit la règle de gestion suivante, issue du domaine Scolarité :

Les étudiants peuvent s'inscrire dans différents cycles, lorsqu'un étudiant se présente pour s'inscrire, il choisit le cycle et règle la totalité des frais d'inscription. Le montant des frais dépend du cycle choisi.

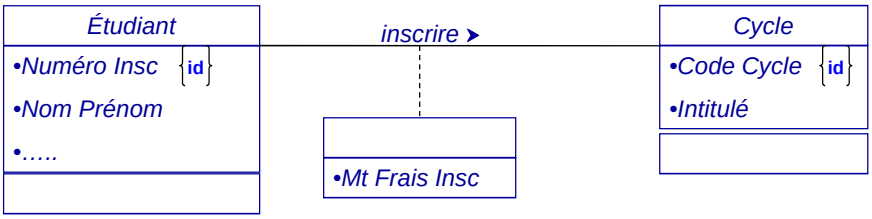


Le modèle proposé respecte-t-il la règle n° 6?

Règles de Vérification du modèle

Identifiant d'une classe association

L'identifiant d'une classe-association est composé des identifiants des différentes classes reliées par l'association.



L'identifiant de la classe association « Inscrire » est : Numéro Insc + Code Cycle.

Le modèle proposé respecte-t-il la règle n° 6?

Règles de Vérification du modèle

Règle n°6:

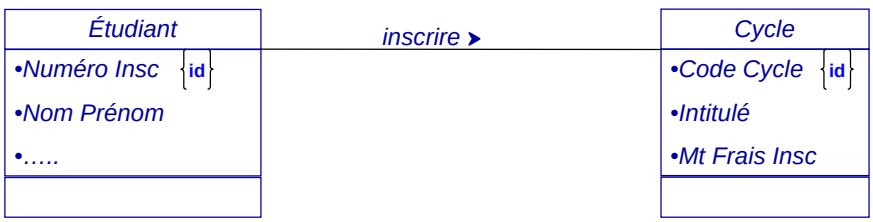
✓ Les attributs d'une classe-association doivent être en dépendance fonctionnelle élémentaire avec l'identifiant de la classe association.

La df suivante est-elle élémentaire ? *Numéro Insc + Code Cycle* —?→ *Mt Frais insc*

Non, car le Montant des frais d'inscription ne dépend pas de l'étudiant, mais du cycle.

Code Cycle → *Mt Frais Insc*

Corriger le modèle.



Démarche de construction du modèle statique

Présentation:

- ✓ Le modèle statique est une représentation abstraite de la structure du système modélisé.
- ✓ Les utilisateurs du système peuvent avoir des visions et des attentes différentes du même système.
- ✓ Le modèle construit doit être le reflet des différentes visions et attentes de l'ensemble de ses utilisateurs.
- ✓ La construction du modèle passe par la connaissance du domaine, la détermination de ses utilisateurs, de leurs visions et attentes.
- ✓ Ces connaissances peuvent être acquises en faisant appel aux techniques et méthodes de:
  - Observation du domaine.
  - Recueil et analyse des documents et des flux.
  - Interview des utilisateurs.

Démarche de construction du modèle statique

Les étapes:

- ✓ La construction du modèle peut se faire en réalisant les étapes suivantes :
  - Construire le référentiel des attributs du domaine. (catalogue ou dictionnaire)
  - Construire le référentiel des règles de gestion du domaine.
  - Déterminer les classes.
  - Déterminer les associations.
  - Affecter les valeurs de multiplicité.
  - Décrire les classes en terme d'attributs et d'opérations.
  - Vérifier le modèle obtenu.