Conception des systèmes d'information

Diagramme de classes

Mariem Haoues

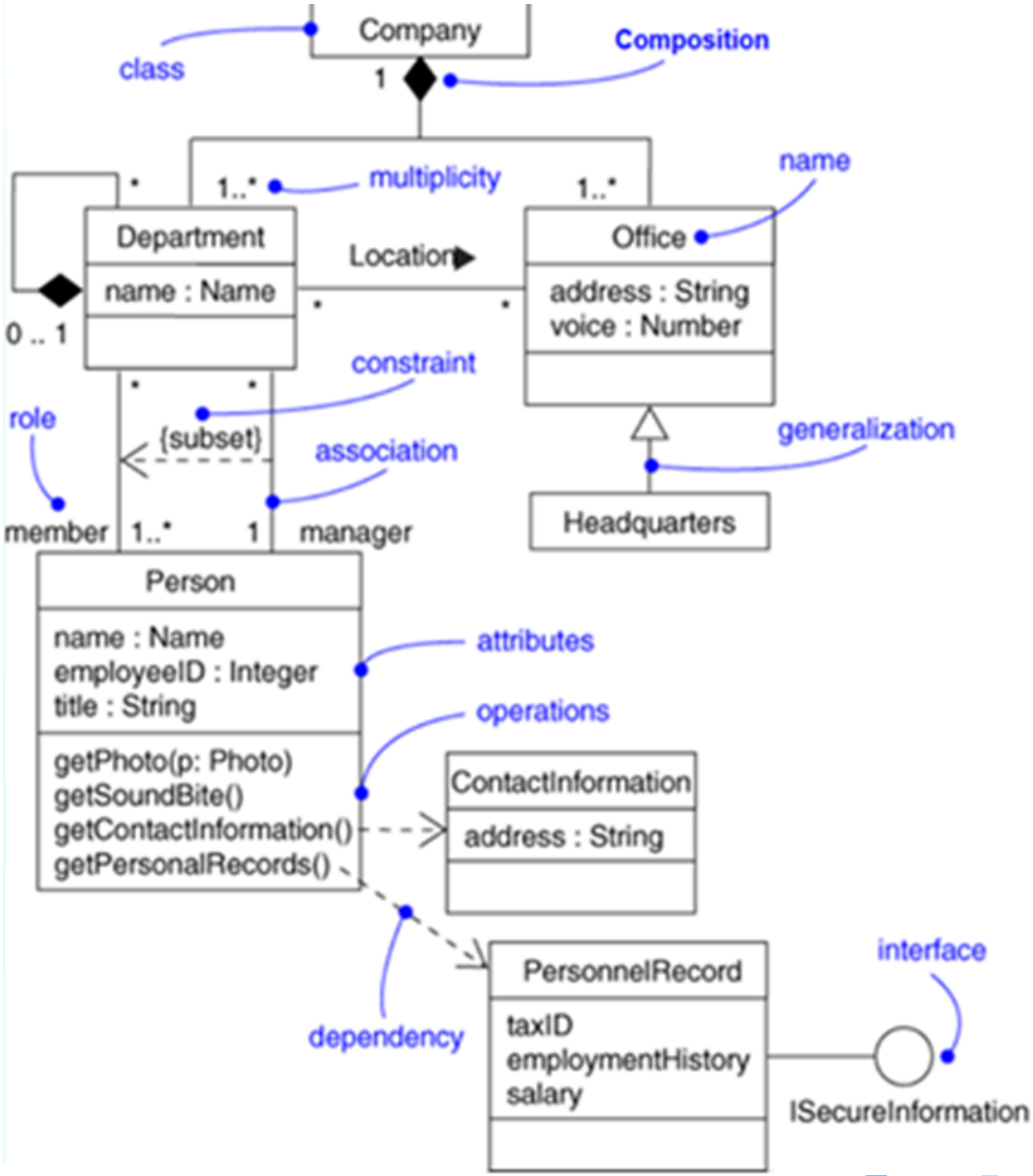
Maître-assistant
FSB, Université de Carthage

10 octobre 2025

Introduction

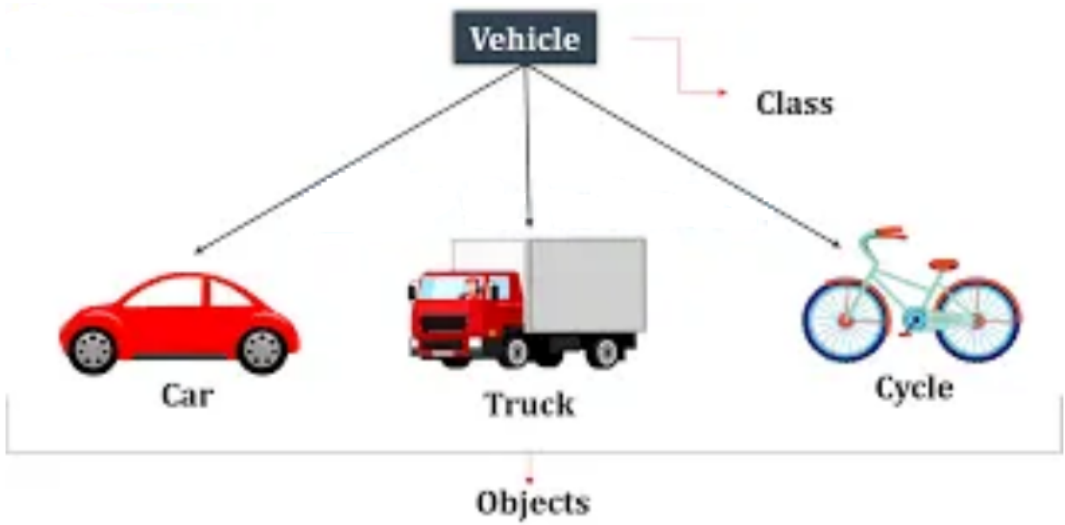
- Les diagrammes de classes expriment la structure statique d'un système en termes de :
  - Classes et relations entre elles, et
  - Un ensemble d'interfaces et de packages, ainsi que leurs relations
- Élément central du développement orienté objet :
  - En analyse, il décrit la structure des entités manipulées par les utilisateurs
  - En conception, il représente la structure du code orienté objet ou, à un niveau plus détaillé, les modules du langage de développement
- **Comment le représenter ?**
  Le diagramme de classes met en œuvre des classes contenant des attributs et des opérations, reliées par des associations ou des généralisations

Introduction



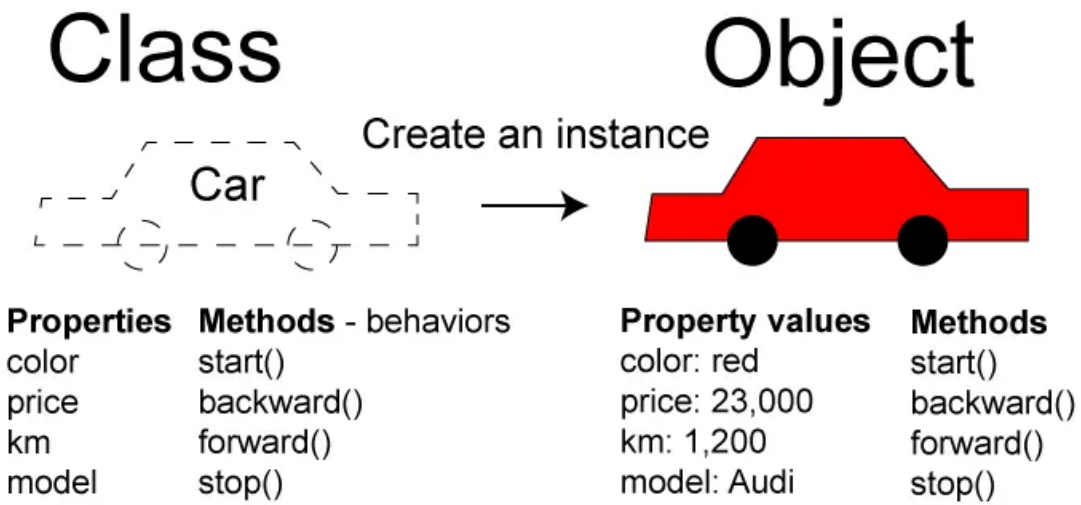
Classe et Objet

- Une classe représente la description abstraite d'un ensemble d'objets partageant les mêmes caractéristiques
  Exemples : Voiture, Personne
- Un objet est une entité avec des limites bien définies, ayant une identité et encapsulant un état et un comportement. Un objet est une instance (ou occurrence) d'une classe.
  Exemples : Pascal Roques est une instance de la classe Personne. Le livre « Base de données » est une instance de la classe Livre.



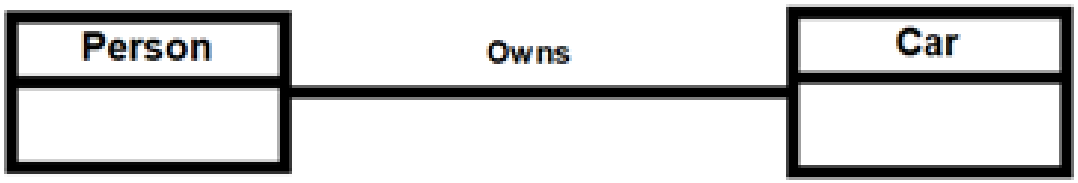
Attribut et Opération

- Un attribut représente un type d'information contenu dans une classe
  Exemples : vitesse actuelle, couleur, numéro d'immatriculation, etc., sont des attributs de la classe Voiture
- Une opération représente un élément comportemental (un service) contenu dans une classe.
  Exemples : pour gérer le solde, la classe CompteBancaire offre les opérations getSolde(), retirer(), déposer(), etc.



Association

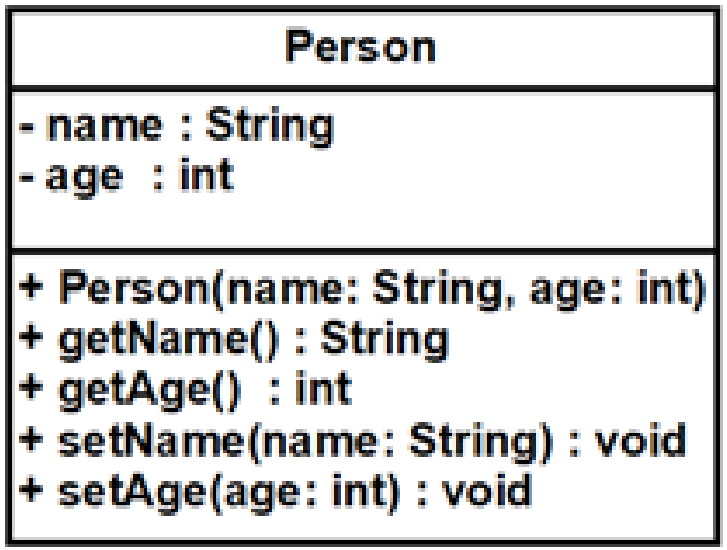
- Une association représente une relation sémantique durable entre deux classes.
  Exemple : Une personne peut posséder une voiture.
  La relation *possède* est une association entre les classes **Personne** et **Voiture**.
  Remarque : même si le verbe nommant une association semble favoriser une direction de lecture, une association entre concepts dans un modèle de domaine est par défaut bidirectionnelle. Ainsi, l'exemple précédent inclut aussi implicitement le fait qu'une voiture est possédée par une personne.



Concepts de base

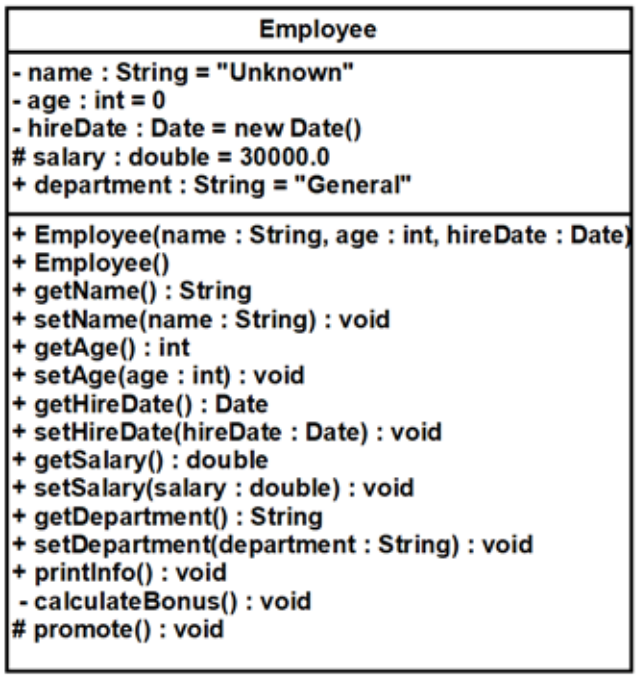
- **Éléments = Classes**
  - Classe concrète
  - Classe abstraite
  - Classe interface
  - Classe paramétrée
- **Relations**
  - Simple
  - Agrégation/Composition
  - Classe d'association
  - Généralisation/Spécialisation

Diagramme de classe et code - Personne




```
public class Personne {
    private String nom;
    private int age;
    // Constructeur
    public Personne(String nom, int age) {
        this.nom = nom;
        this.age  = age;
    }
    // Accesseurs
    public String getNom() {
        return nom;
    }
    public int getAge() {
        return age;
    }
    // Modificateurs
    public void setNom(String nom) {
        this.nom = nom;
    }
    public void setAge(int age) {
        this.age = age;
    }
}
```

Notation d'une Classe



Elle spécifie :

- Le type des variables (Integer, String, Date, etc.)
- Les valeurs par défaut
- Les signatures des opérations
- Le niveau de visibilité :
  - + **public** (accessible par tout utilisateur) – par défaut
  - - **private** (accessible uniquement dans la classe)
  - # **protected** (accessible par les sous-classes)

Notation des Attributs et des Opérations

- **Attribut**
  [visibilité] NomAttribut [: type] [= <valeur par défaut>]
  Exemple : - age : int = 0

- **Opération**
  [visibilité] NomOpération([(Liste des paramètres)]) [: typeRetour]
  Exemple : + getNom() : String

- **Paramètre**
  [direction] Nom : type [= valeur par défaut], direction : in | out | inout (par défaut : in)
  Exemple : in compteur : int = 1

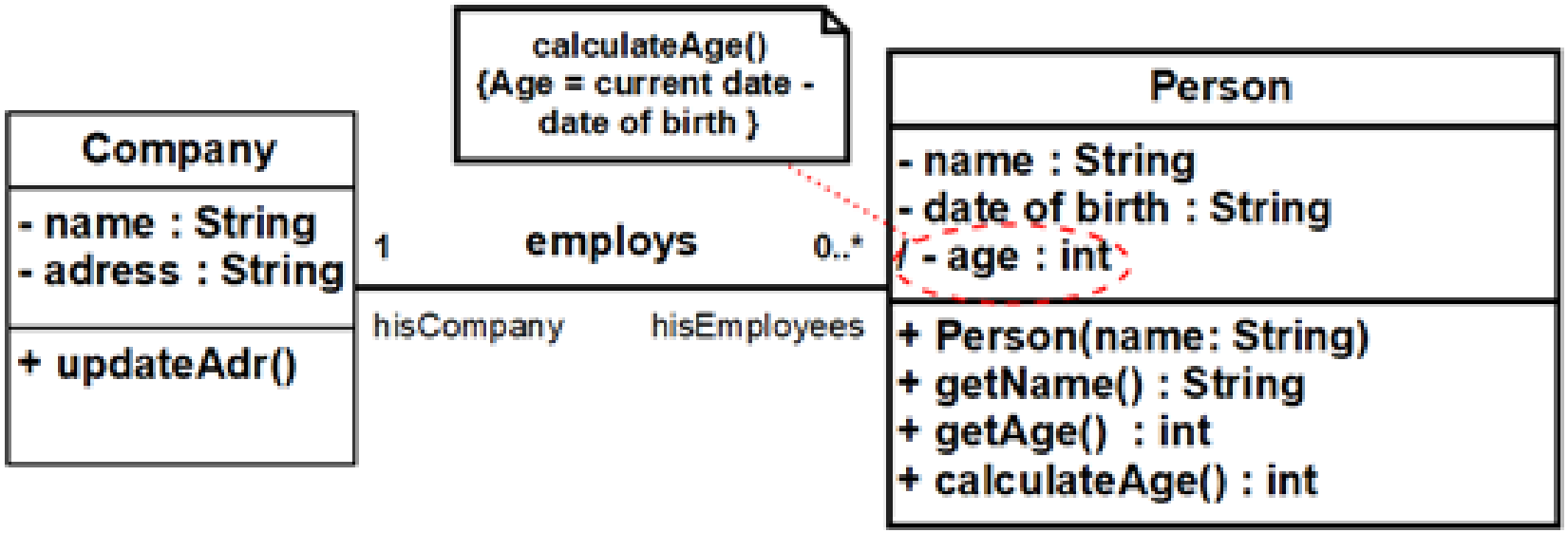
Directions de paramètres : in, out, inout

- **in** – Paramètre d'entrée (par défaut)
  - La méthode reçoit la valeur, mais ne modifie pas la variable de l'appelant.
  - **Exemple :** [language=Java] void afficherCompteur(int compteur) // compteur est utilisé mais la valeur originale reste inchangée
- **out** – Paramètre de sortie
  - La méthode écrit une valeur dans le paramètre ; la valeur initiale est ignorée.
  - **Exemple :** [language=Java] void obtenirValeurSuivante(int[] resultat) resultat[0] = 42 ; // affecte la valeur pour l'appelant
- **inout** – Paramètre d'entrée/sortie
  - La méthode lit et peut modifier la valeur ; les changements sont visibles pour l'appelant.
  - **Exemple :** [language=Java] void incrementerDeUn(int[] nombre) nombre[0] = nombre[0] + 1 ; // modifie la valeur de l'appelant

Attribut dérivé

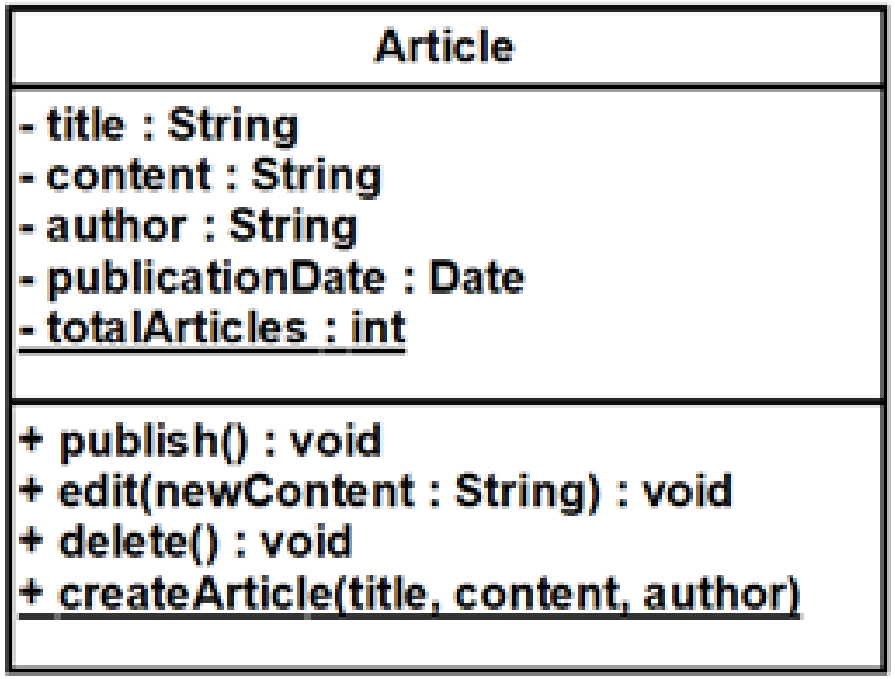
- Un attribut dérivé (noté /Attribut) est un attribut calculé. Cela signifie qu'il peut être déterminé à tout moment à partir d'autres informations du système.
  Exemple : pour une personne, l'attribut *Âge* est un attribut calculé à partir de sa *DateDeNaissance*.
- **En revanche :** si une *QuantitéStock* est mise à jour par ajout ou retrait sans conserver l'historique des transactions, alors *QuantitéStock* n'est pas un champ calculé.
  L'attribut *QuantitéStock* est dit modifiable (par défaut, tous les attributs le sont).
- {`frozen`} est réservé aux attributs qui, une fois initialisés, ne peuvent plus être modifiés.

Exemple


calculateAge()
{Age = current date - date of birth }
Company
- name : String
- adress : String
+ updateAdr()
1
employs
0..*
hisCompany
hisEmployees
Person
- name : String
- date of birth : String
/ - age : int
+ Person(name: String)
+ getName() : String
+ getAge() : int
+ calculateAge() : int

Attributs et Opérations de Classe

- **Attributs de classe :** décrivent les valeurs partagées par tous les objets de la classe.
- **Opérations de classe :** opérations portant sur la classe elle-même. La plus courante est utilisée pour créer de nouvelles instances de la classe.
- Les attributs et opérations de classe sont soulignés.



Exemple : Classe Article

- **Attributs :**
  - titre : String *(Le titre de l'article)*
  - contenu : String *(Le texte principal ou le corps de l'article)*
  - auteur : String *(Nom de l'auteur ayant rédigé l'article)*
  - datePublication : Date *(La date de publication de l'article)*
- **Attributs de classe :**
  - totalArticles : int *(Compte le nombre total d'instances de la classe Article ; partagé par tous les articles)*
- **Opérations :**
  - publier() : void *(Marque l'article comme publié)*
  - modifier(nouveauContenu : String) : void *(Met à jour le contenu de l'article)*
  - supprimer() : void *(Supprime l'article)*
- **Opérations de classe :**
  - creerArticle(titre, contenu, auteur) : Article
    (Crée une nouvelle instance d'Article et incrémente totalArticles)

Exercice : Application des Concepts

Une **Personne** est caractérisée par son numéro CIN, son nom, son prénom, sa date de naissance, son adresse et son âge. Les attributs de la classe Personne sont privés ; le nom, le prénom et l'âge doivent être accessibles via des opérations publiques. Un objet de la classe Personne peut être créé à partir du nom et de la date de naissance. Il est possible de modifier l'adresse d'une personne.

1. Représentez la classe `Personne` avec tous ses attributs, opérations et la visibilité appropriée.

Les **Employés** et les **Clients** sont deux types de personnes. Un Employé est caractérisé par son matricule, son salaire et son poste. Un Client est caractérisé par son numéro de client, sa catégorie et son solde d'achat cumulé.

2. Enrichissez votre représentation précédente pour inclure ces nouveaux éléments.

Quiz

1. Laquelle des représentations suivantes correspond correctement à un attribut *privé* age de type `int` avec une valeur par défaut de 0 en UML ?

a- + age : int = 0

b- - age : int = 0

c- # age : int = 0

d- / age : int = 0

2. En UML, laquelle des représentations suivantes montre un paramètre d'entrée *count* de type *int* avec une valeur par défaut de 1 ?

a- out count : int = 1

b- in count : int = 1

c- inout count : int = 1

d- count : int

Quiz

3. Une classe Voiture possède les attributs : - *modele : String*, - *annee : int = 2020*, et les méthodes : + *demarrer() : void*, + *arreter() : void*.
Laquelle des affirmations suivantes est vraie ?

a- Tous les attributs sont publics

b- Toutes les méthodes sont privées

c- *annee* a une valeur par défaut de 2020

d- *demarrer()* ne peut pas être appelée depuis l'extérieur de la classe

4. Comment un constructeur est-il généralement représenté dans un diagramme de classes UML ?

a- Comme une méthode normale mais portant le même nom que la classe

b- Avec un symbole # avant le nom

c- Avec un type de retour void

d- Il n'est pas représenté en UML

Thank you for listening !

Mariem Haoues

mariem.houes@fsb.ucar.tn