

Université de Corse
2021-2022
L3 Informatique

UE Conception et Programmation Objet
Modélisation UML

CH2 – Modèle du Domaine



Cours de Mme Evelyn Vittori



1

Modélisation UML
Plan du Cours

CH1 – UML et ACOO

→ CH2 – MODELE DU DOMAINE

CH3 – MODELE DES CAS D'UTILISATION

CH4 – MODELE D'ANALYSE

2

2

CH2 – MODELE DU DOMAINE


→ 2.1 – Présentation

- Objectifs
- Artefacts

2.2 – Formalisme des Diagrammes de classe

2.3 – Diagrammes d'objets

2.4 – Démarche de construction



3


3

MODELE DU DOMAINE

Objectif du modèle du domaine

- Comprendre et décrire les concepts essentiels dans le contexte du système
- « Concepts métiers » : concepts manipulés par les experts du domaine
- Ensemble des données importantes du domaine

Les données que l'on souhaite conserver de manière durable



4

4



MODELE DU DOMAINE

Artefacts du modèle du domaine

+ Diagrammes d'objets



Diagramme de classes du domaine

- Classes
- Relations
 - Associations, agrégations, composition
 - Généralisation
- Attributs

+ Packages

5

5



CH2 – MODELE DU DOMAINE

2.1 – Présentation

2.2 – Formalisme des Diagrammes de classe

- Définition
- Diagrammes de classe et étapes d'ACOO
- Formalisme



2.3 – Diagrammes d'objets

2.4 – Démarche de construction

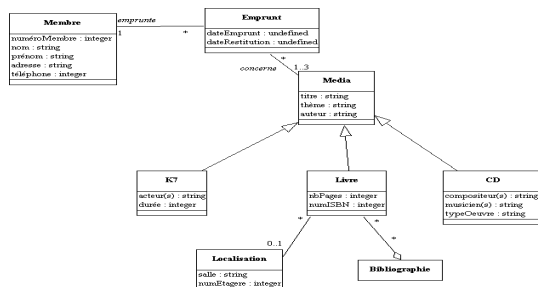
6

6



Définition

- Un **diagramme de classes** exprime la structure d'un système en termes de classes et de relations entre ces classes



7

7



Diagrammes de classes et étapes d'ACOO

Le **diagramme de classes du domaine** est le premier diagramme de classe défini au cours d'une démarche d'ACOO.

Il est ensuite complété pour obtenir le diagramme de **classes d'analyse** lui-même complété pour aboutir au diagramme de **classes de conception**.

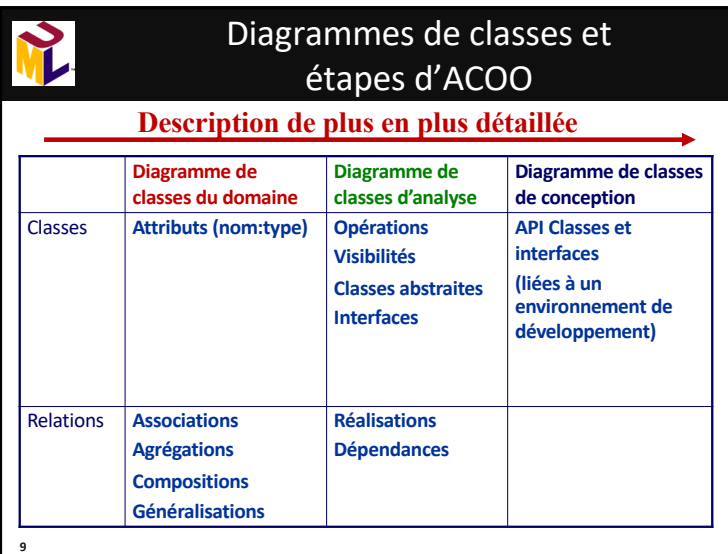
3 - Diagramme de classes de Conception

2 - Diagramme de classes d'Analyse

1 - Diagramme de classes du Domaine

8

8



9

CH2 – MODELE DU DOMAINE

2.1 – Présentation

→ 2.2 – Formalisme des Diagrammes de classe

- Rappel: notion d'objet
- Représentation d'une classe
- Attributs et opérations
- Relations entre classes
- Classes Abstraites et interfaces
- Autres concepts (Conception)

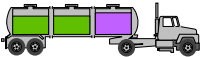
2.3 – Diagrammes d'objets


2.4 – Démarche de construction

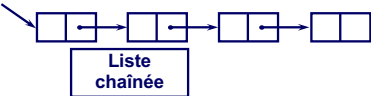
10

10

Notion d'objet
Qu'est-ce qu'un objet?

Entité physique 
Camion

Entité conceptuelle 

Entité logicielle 
Liste chaînée

11

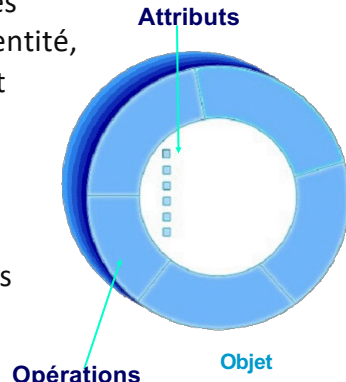
11

Notion d'objet

Objet = entité aux frontières précises qui possède une identité, un état et un comportement

Etat = attributs + liens

Comportement = opérations



12

12

Notion d'objet en JAVA

Variable définie sur une classe

prof001

nom	Nimbus
age	30
discipline	physique
grade	MCF Ech.6
Statut	actif
nbCoursMax	4

- Une **adresse (ou référence)** en mémoire qui permet d'**identifier** l'objet
- Un **état** qui est représenté par un ensemble de valeurs attribuées à ses variables d'instances
- Un **comportement** défini par des fonctions ou sous-programmes appelés méthodes

13

Formalisme du Diagramme de classes Représentation d'une classe

Une **classe** est une **abstraction d'un ensemble d'objets**

- Mise en valeur des caractéristiques les plus importantes.
- Suppression des autres caractéristiques.

attributs (pointing to attributes)

opérations (pointing to methods)

14

Classes en JAVA et en UML

Une classe en JAVA

```

class Vehicule{
  /** l'immatriculation de ce véhicule */
  String immat;
  /** la puissance */
  int puissance;
  /** La consommation de ce véhicule. */
  double poids;

  ...

  void faireLePlein() {
    ....
  }
  void reviser() {
    ....
  }
}

```

Une classe en UML

attributs (pointing to attributes in both)

méthodes (pointing to methods in both)

15

Représentation d'une classe

attributs

opérations

Commence par une majuscule (pointing to the class name)

Non représentées dans un modèle du domaine (pointing to the operations section)

Exemples

Etudiant
 nom
 prenom
 dateNaissance

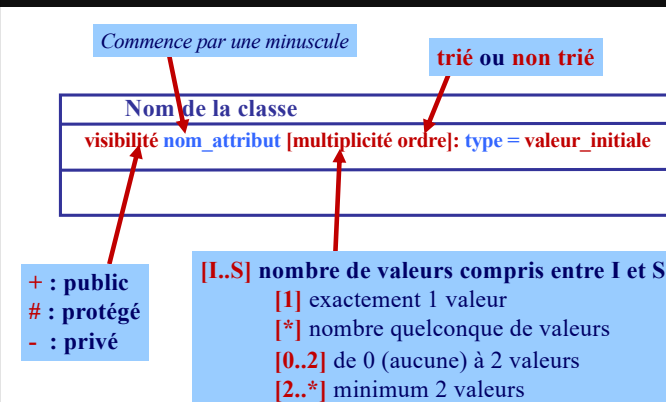
Enseignant

Cours

16



Représentation d'une classe



17

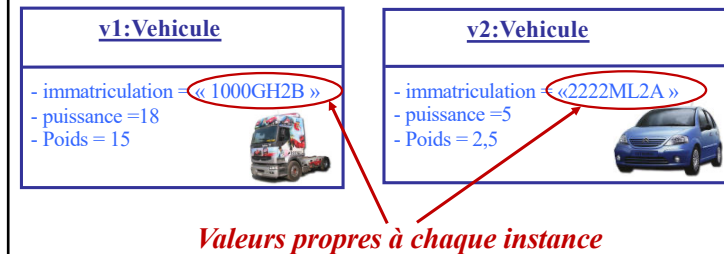
17



Représentation d'une classe

Attribut (d'instance) =

- défini par un nom, un type et éventuellement une valeur initiale
- valeur spécifique pour chaque instance



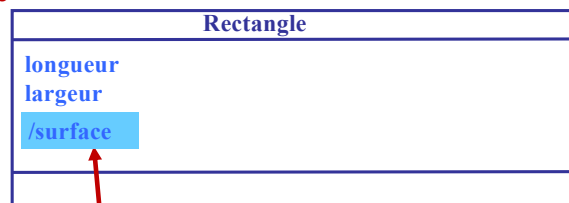
18

18



Représentation d'une classe

Exemple



Attribut dérivé (/nom):
valeur calculée à partir de celles d'autres attributs

19

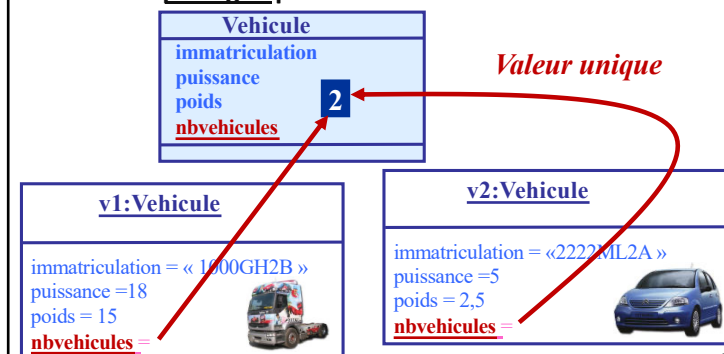
19



Représentation d'une classe

Attribut de classe =

Valeur partagée par toutes les instances de la classe



20

20



Attributs et méthodes de classe en Java



```

public class Vehicule{
    private String immatriculation;
    private int nbPlaces;
    Private int age;
    private static int nbVehicule=0;
    private static int totalNbPlaces=0;
    Public Vehicule(String immatriculation,
        int nbPlaces){
        this.immatriculation=immatriculation;
        this.nbPlaces=nbPlaces;
        this.age=0;
        nbVehicule++;
        totalNbPlaces=totalNbPlaces+nbPlaces;
    }
    public void augmenterAge(){ age++;}
    public static void afficherNbVehicule(){
        System.out.println("Nombre de Vehicules "+ nbVehicule);
    }
}
    
```

instance

```

class TestVehicule{
    public static void main(String[] args){
        Vehicule v1=new
        Vehicule("2222CX23" , 6);
        v1.augmenterAge();
        Vehicule.afficherNbVehicule();
    }
}
    
```

classe

21



Représentation d'une classe

Exemple

Employé

```

- iD : Entier
- nom : Chaîne
- adresseEmail [1..5 non trié] : Chaîne = " Pas d'adresse email"
- numeroTelephone [1..* trié] : Chaîne
#motDePasse : Chaîne
#nbrEmploye: entier = 0
    
```

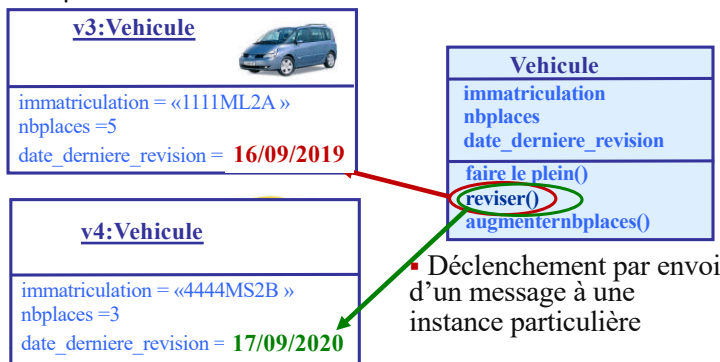
Attribut de classe (nom souligné):
valeur partagée par tous les objets d'une classe

22



Représentation des opérations

- Les **opérations** sont attachées à une classe.
- L'exécution d'une opération porte sur une instance particulière.



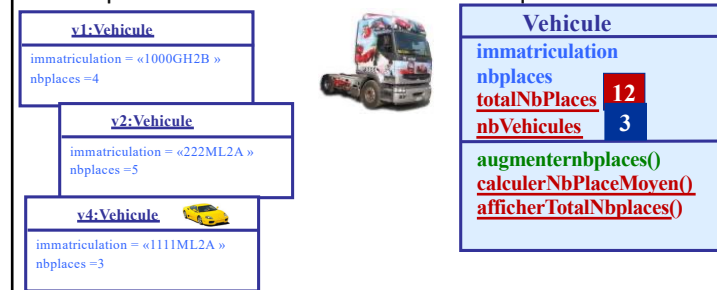
23



Représentation des opérations

Opérations de classe =

- Exécution déclenchée par un message envoyé à la classe.
- Opération qui ne concerne pas une instance particulière mais la classe
- Manipulation des attributs de classe uniquement



24

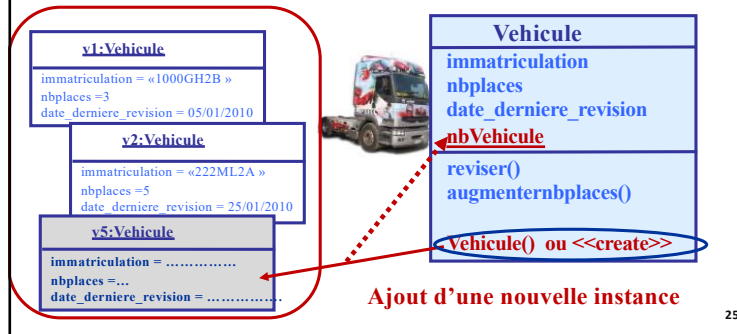
24



Représentation des constructeurs

Mécanisme d'instanciation =

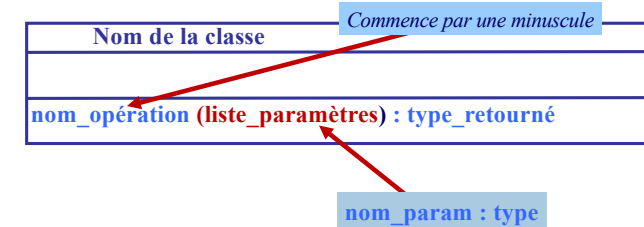
- Activation de l'opération de création d'instance de la classe: Constructeur



25



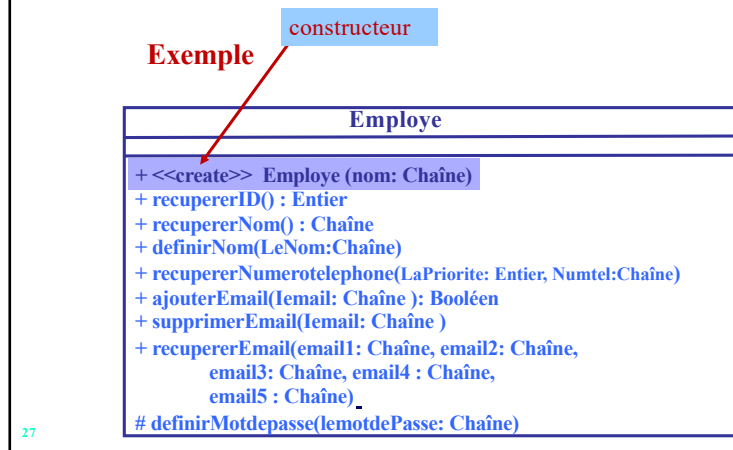
Représentation des opérations



26



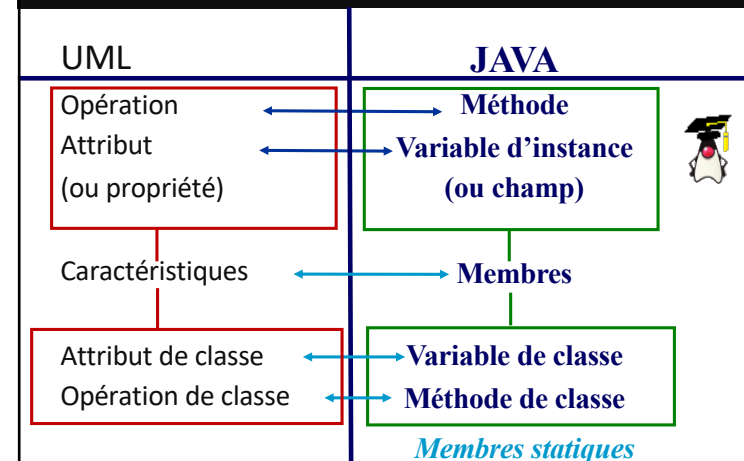
Représentation des opérations



27



Attributs et Opération - Terminologie



28



Modélisation: Distinction Classe/Attribut

Comment savoir si une entité doit être modélisée par une classe ou un simple attribut?

- Attribut = **valeur simple**
 - Entité qui ne peut être caractérisée que par sa valeur (« on ne peut lui demander que sa valeur »)
- Classe = **objet**
 - Entité caractérisée par plusieurs autres entités (« on peut lui poser plusieurs questions »), il s'agit d'un objet qui possède plusieurs attributs et des liens avec d'autres objets.

29

29



Exercice: Classe ou attribut ?

Parmi les éléments, lesquels sont des objets qui appartiennent à des classes, lesquels sont de simples valeurs d'attributs caractérisant d'autres objets ?

- la guerre de 100 ans
- le réfrigérateur dans le coin de la pièce
- la couleur rouge
- une transaction boursière
- Le temps d'exécution d'un programme
- Amadeus Mozart
- l'heure de départ d'un vol
- le chiffre 3

30

30



Exercice: Classe, attribut ou opération?

Parmi les éléments suivants, lesquels peuvent être modélisés par une classe? par un attribut ? par une opération?

	Classe	Attribut	Opération
Un immeuble			
Une longueur			
Une ville			
Une superficie			
Une couleur			
Une personne			
Une date de naissance			
Un age			

Il y a plusieurs réponses possibles, l'intérêt est le questionnement!!

31

31



Relations entre Classes

UML définit quatre principaux types de relations entre classes

- Association
 - Association simple
 - Agrégation
 - Composition
- Généralisation (Héritage)
- Réalisation
- Dépendance

Concepts de niveau analyse/conception

32

32



Associations

ligne pleine entre classes

Exemple d'association binaire



Précise le sens de la lecture (optionnel)

33

33



Nommage des Associations

- Nom de l'association en italique au milieu de la ligne
- Forme verbale active ou passive



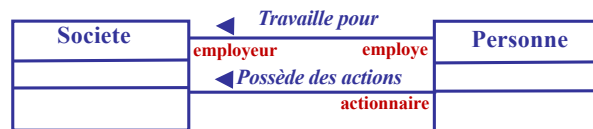
34

34



Nommage des rôles

- Un rôle définit la manière dont une classe intervient dans une association



- Le rôle est indispensable lorsqu'il y a plusieurs associations entre 2 classes

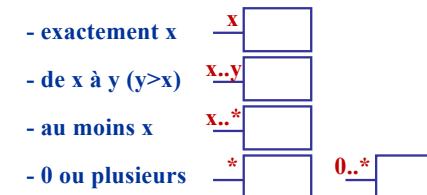
35

35



Cardinalité des associations

- La cardinalité (ou multiplicité) d'une association précise le nombre d'instances qui participent à une relation

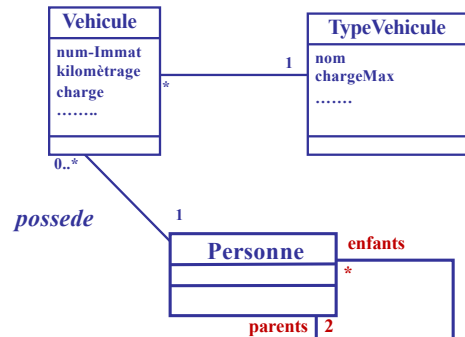


36

36



Cardinalité des associations



Association reflexive

37

37



Traduction des associations

Les associations sont traduites par l'ajout d'attributs dans les classes lors de la programmation

- La multiplicité définit le **style d'attribut**:
 - multiplicité 0 ou 1: références ou pointeurs
 - multiplicité > 1: listes, ensembles, ...
- La multiplicité définit le **caractère obligatoire ou facultatif** de l'attribut:
 - multiplicité 0.. : les opérations devront tester la présence de la relation avant de l'utiliser
 - multiplicité >0: l'attribut correspondant aura une (ou plusieurs) valeur(s) obligatoire(s).

38

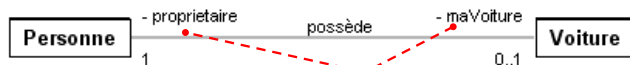
38



Traduction des associations

Multiplicité ≤1

- Les attributs ajoutés pour représenter l'association sont du type des classes associées.



```
public class Personne {
    private Voiture maVoiture;
}
```

```
public class Voiture {
    private Personne propriétaire;
}
```



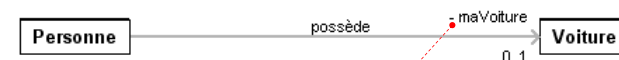
39



Traduction des associations

Multiplicité ≤1

- Association **unidirectionnelle**:
L'attribut ne sera ajouté que dans une seule classe.



```
public class Personne {
    private Voiture maVoiture;
    ...
}
```



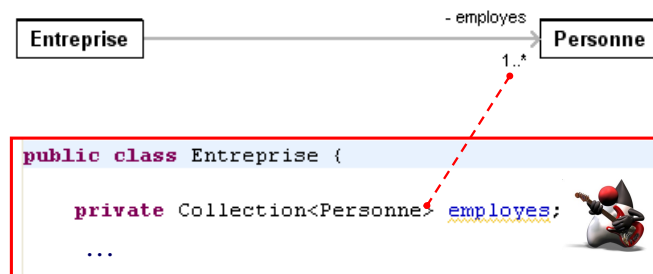
40



Associations entre classes

Multiplicité >1

- L'attribut ajouté pour représenter l'association est de type tableau ou collection.

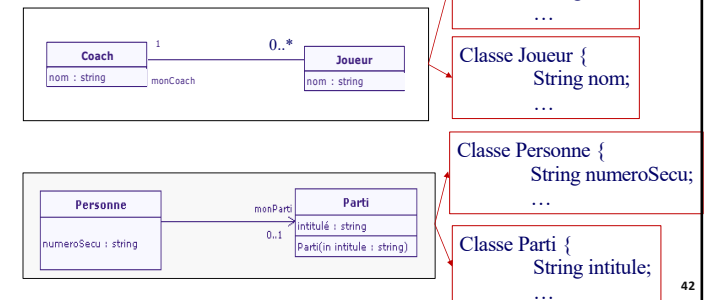


41



Exercice: Traduction des associations

- Pour chacun des diagrammes suivants indiquer les attributs à ajouter dans les classes lors de la phase d'implémentation

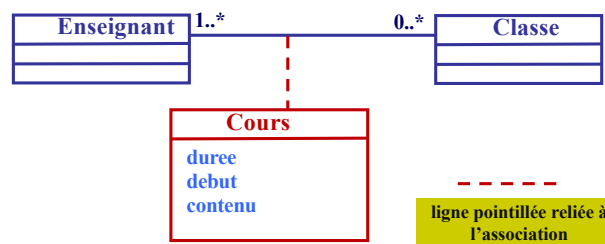


42



Classe-Association

- Une association porteuse d'attributs est représentée par une **classe associative** ou **classe-association**.



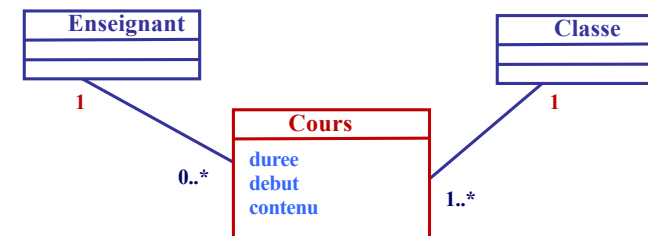
43

43



Classe-Association

- Un diagramme de classe comportant une classe-association peut toujours être remplacé par un diagramme équivalent sans classe association:



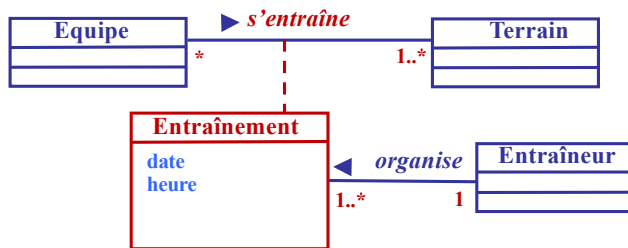
44

44



Exercice: Classe-Association

- Définissez un diagramme de classe équivalent au diagramme suivant mais ne comportant pas de classe association.




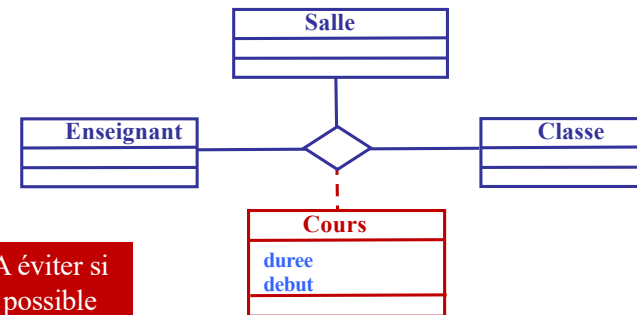
45

45



Association n-aire

- Une association n-aire  implique au moins 3 classes



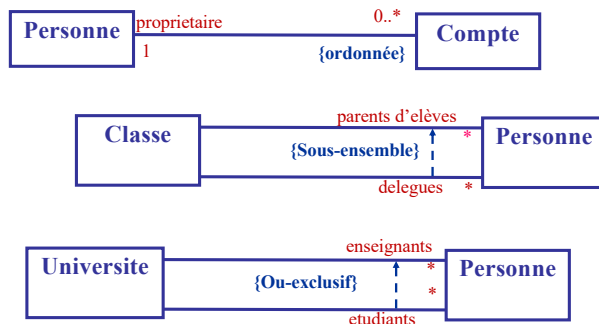
46

46



Contraintes sur les associations

- Une contrainte porte sur une association ou sur un groupe d'associations **{Contrainte}**



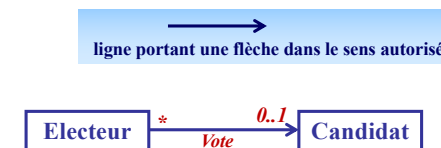
47

47



Navigabilité des Associations

- Par défaut, une association est navigable dans les deux sens
- Lors de l'analyse, on peut exprimer le fait qu'un seul sens ne devra être implémenté :



48

48

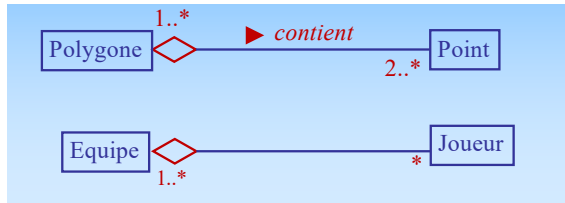


Agrégation

- Une **agrégation** est une association non symétrique dont la sémantique évoque une relation de contenance



Exemples



49

49

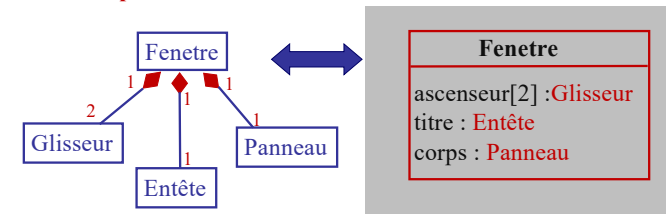


Composition

- Une **composition** est une agrégation avec des contraintes fortes sur les cardinalités et les durées de vie composant/composé



Exemple



50

50



Généralisation -Spécialisation

- Généralisation :
relation « EST UN » ou « EST UNE SORTE DE »
- Factorisation des éléments communs d'un ensemble de classes
- Super-classe** = abstraction de ses sous classes
- généralisation / spécialisation= deux points de vue antagonistes du concept de classification

51

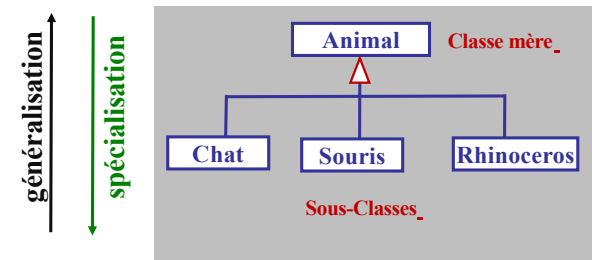
51



Généralisation -Spécialisation

- Relation EST DE ou EST UNE SORTE DE Δ

Notion d'héritage



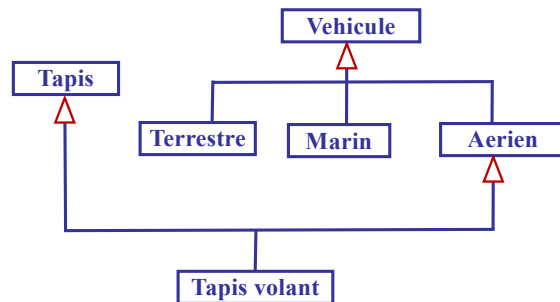
52

52



Généralisation -Spécialisation

■ Notion d'héritage multiple



53

53



Exercice: Instanciation ou spécialisation?

Pour chacune des phrases suivantes, indiquez si la relation décrite est une instanciation ou une spécialisation :

- une toyota est une voiture
- un appartement est une habitation
- Lady est un chien
- un singe est un animal
- Ajaccio est une ville

54

54



Dépendances

- L'**association** représente une connexion structurelle (représentée par des **attributs** références) entre les objets des classes associées.

Identifiée dans le modèle du domaine

- La **dépendance** est une relation d'usage. Elle traduit l'utilisation temporaire (dans une méthode) d'objets de la classe dont on dépend.

Identifiée dans le modèle d'analyse

55

55



Dépendances

- variable locale
- paramètre de méthode
- résultat de méthode



56

56



Dépendances

- Création d'un objet d'une autre classe en tant que variable locale d'une méthode.



- Objet d'une autre classe comme paramètre de méthode.



57

57



Dépendances

- Objet d'une autre classe comme résultat de méthode.



58

58



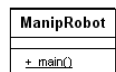
Dépendances (Résumé)

- Une dépendance est une relation non structurée entre classes (communication momentanée, limitée dans le temps):



Au moins une méthode de A :

- contient une **variable locale** de type B
- possède un **paramètre** de type B
- renvoie un **résultat** de type B



Variable locale de type Robot

```
public class ManipRobot {
    public static void main(String[] args) {
        Robot r = new Robot("Toto", 10, 20, Robot.NORD);
        r.changerOrientation(Robot.SUD);
        r.deplacer();
    }
}
```



59



Classes Abstraites et Interfaces

Classe abstraite

- Une classe abstraite est une classe qui n'a pas d'instances directes.

Nom en italique

Nom de la classe abstraite
{abstraite}

Propriété {abstraite=vrai}

60

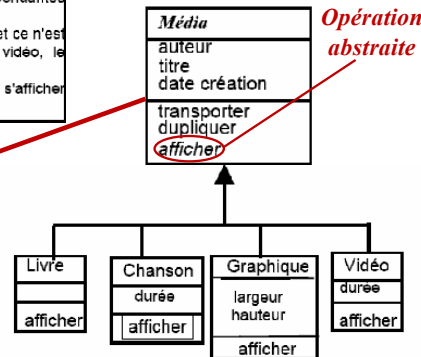
60



Classes abstraites

Un média peut être transporté, dupliqué, affiché. Le transport et la duplication sont indépendantes du type du média (copie de fichiers). Par contre, tout média peut être affiché et ce n'est pas la même chose pour l'audio, la vidéo, le graphisme, le texte. Un média ne peut pas définir comment s'afficher tant qu'il ne sait pas ce qu'il est.

Il n'y a pas d'instance de la classe média. Un média existe en tant que livre, chanson, graphique, vidéo.



61



Interfaces

- Une interface est une classe qui ne possède que des opérations abstraites



- ✓ Description du comportement visible d'une classe, d'un composant: *liste d'opérations publiques (services de l'interface)*
- ✓ Généralisation possible entre les interfaces

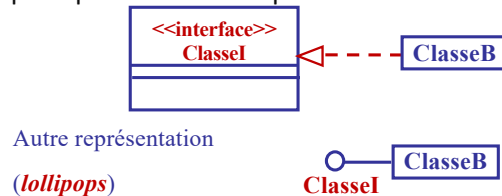
62



Interfaces

Les classes peuvent être reliées aux classes interfaces par deux sortes de relations :

- relation de **réalisation** (implémentation)
Une classe B réalise (ou implémente) une classe I interface si elle fournit un ensemble de méthodes qui implémentent les opérations de l'interface.

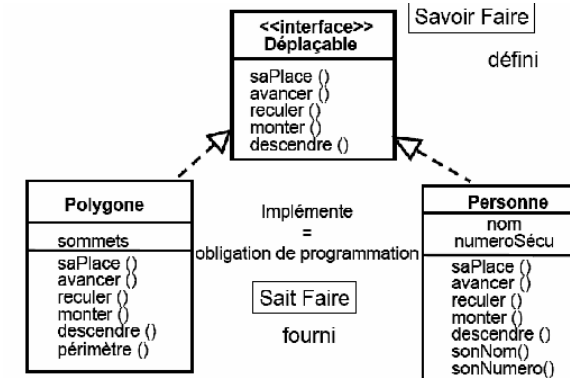


63

63



Interfaces



64

64



Interfaces

Relation de dépendance (utilisation)

- Une classe A requiert une interface I si elle a besoin (ou utilise) d'une instance d'une classe qui implémente cette interface pour travailler.



Autre représentation (*lollipop*)



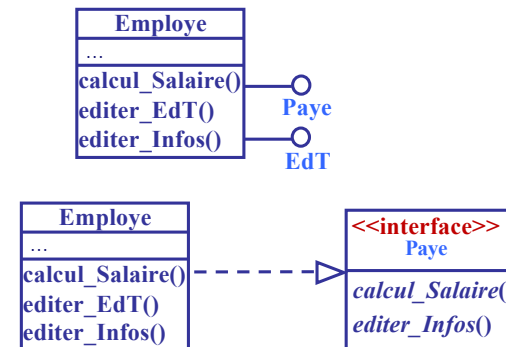
65

65



Interfaces

Exemple



66

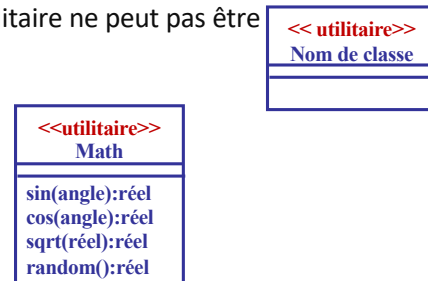
66



Autres Concepts (Conception)

- Classes utilitaires
 - Regroupement de variables et opérations de classes
 - Une classe utilitaire ne peut pas être

Exemple



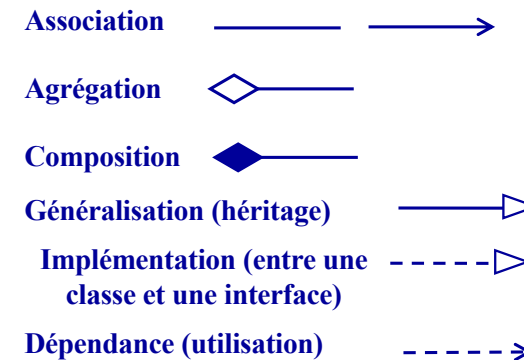
67

67



Formalisme du Diagramme de classes

Relations entre classes (résumé)



68

68



Exercice: Association, Agrégation ou Composition

Pour chacune des associations suivantes, indiquez s'il s'agit d'une association simple, d'une agrégation ou d'une composition

- Une université emploie des enseignants
- Une personne possède une voiture
- Une maison comporte des pièces
- Un zoo contient des animaux
- Une voiture possède des roues et un châssis
- Une page web comporte des liens et des images
- Un livre comporte des pages

69