

---

## UML: Unified modeling Language

---

Leçon 4 - 02/10/2008  
 Modèle structurel 2/2  
**Luigi DILILLO - LIRMM / UM2**  
[dilillo@lirmm.fr](mailto:dilillo@lirmm.fr)

## Plan

- Cours 4
  - Modèle structurel (2/2)*

## Modèle structurel

*En UML, le modèle structurel ou statique est décrit à l'aide de deux sortes de diagrammes*

- Diagrammes de classes
 

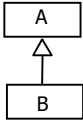
(description de tout ou d'une partie du système d'une manière abstraite, en termes de classes, de structure et d'associations).
- Diagrammes d'objets
 

(description d'exemples de configuration de tout ou partie du système, en termes d'objets, de valeurs et de liens).

## Modèle structurel

### Généralisation / Spécialisation

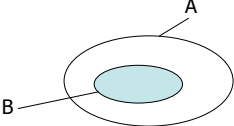
- Elle relie deux éléments de modèle (**classes**, cas d'utilisation, méthodes)



B *spécialise* A ; A *généralise* B

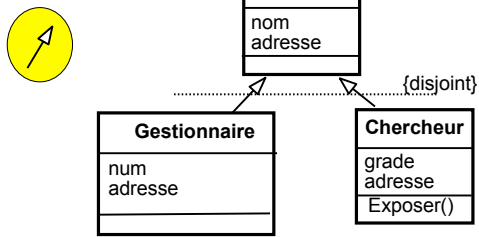
Définition : L'extension de A (ses instances) contient l'extension de B

Conséquence : une instance de B possède les propriétés de A éventuellement sous une forme affinée



## Modèle structurel

### Généralisation / Spécialisation



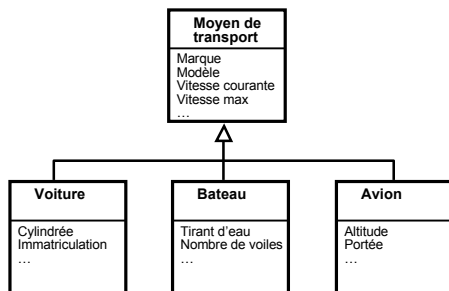
## Modèle structurel

### Généralisation / Spécialisation

- Une sous-classe "hérite" des descriptions de sa super-classe :
  - les déclarations d'attributs,
  - les définitions d'opérations,
  - les associations définies sur la super-classe,
  - les contraintes (on en parle plus tard).
- Une sous-classe peut redéfinir de façon plus spécialisée une partie ou la totalité de la description « héritée ».

## Modèle structurel

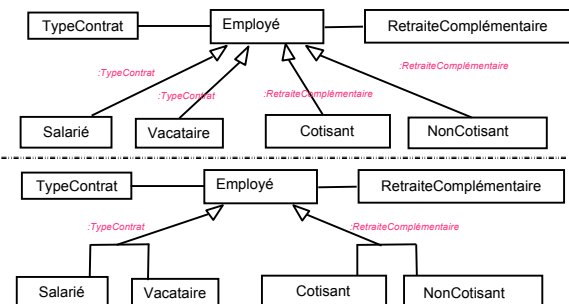
### Généralisation / Spécialisation



## Modèle structurel

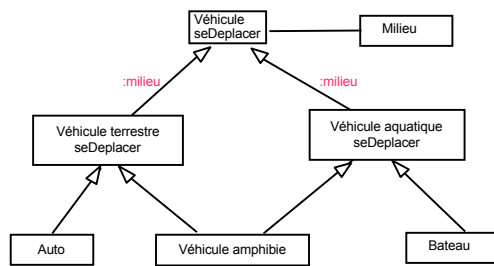
### Généralisation / Spécialisation

Le discriminant : exprime un critère de classification



## Modèle structurel

### Généralisation / Spécialisation multiple



## Modèle structurel

### Composition/Agrégation ou généralisation ?

- Agrégation
  - lien entre instances
  - un arbre d'agrégation est composé d'objets qui sont parties d'un objet composite
- Généralisation
  - lien entre classes

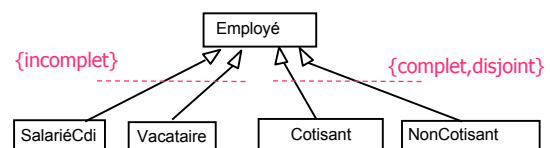
## Modèle structurel

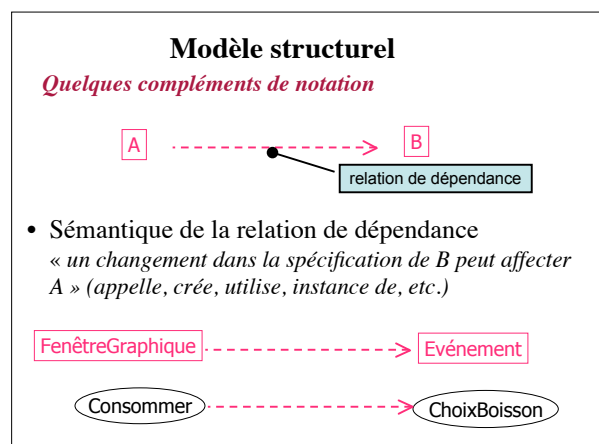
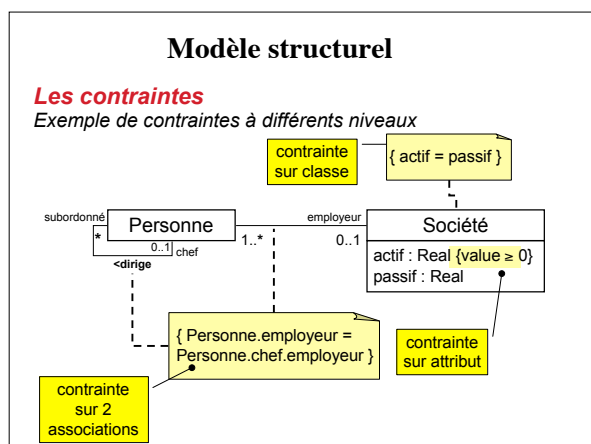
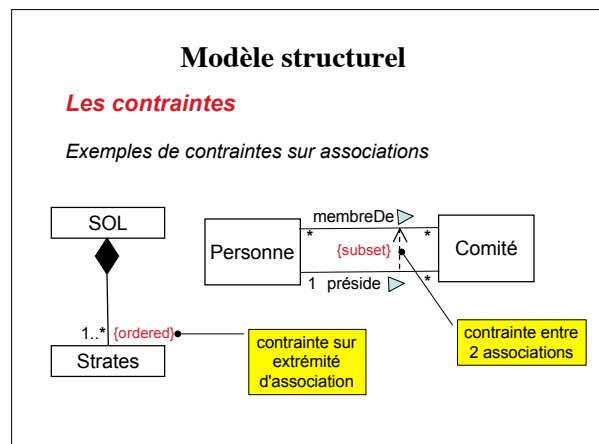
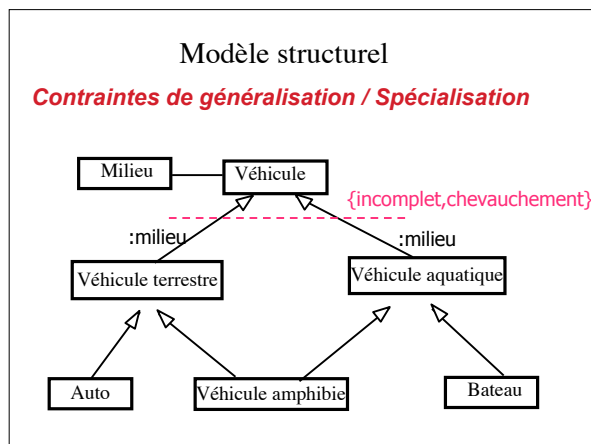
### Les contraintes

- Les *contraintes* sont des prédicats, pouvant porter sur plusieurs éléments du modèle statique, qui doivent être vérifiés à tout instant.
- Les contraintes permettent de rendre compte de détails à un niveau de granularité très fin dans un diagramme de classe. Elles peuvent exprimer des **conditions** ou des **restrictions**.
- En UML, les contraintes sont exprimées sous forme textuelle, entre accolades et de préférence en OCL (Object Constraint Language)  
Ex. {ordered}
- Les contraintes sont **héritées**.

## Modèle structurel

### Contraintes de généralisation / Spécialisation

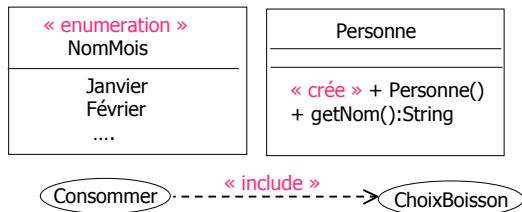




## Modèle structurel

### Quelques compléments de notation

- Un **stéréotype** est un label qui permet d'apporter une précision supplémentaire à un élément de notation (classe, relation, ...)



## Modèle structurel

### Classes abstraites

(notation italiques ou avec mot-clef {abstract})

- Une *classe abstraite* est une classe non instanciable, c'est à dire qu'elle n'admet pas d'instances directes.
- Une classe abstraite est une description d'objets destinée à être « héritée » par des classes plus spécialisées.
- Pour être utile, une classe abstraite doit admettre des classes descendantes *concrètes*.
- La factorisation optimale des propriétés communes à plusieurs classes par généralisation nécessite le plus souvent l'utilisation de classes abstraites.

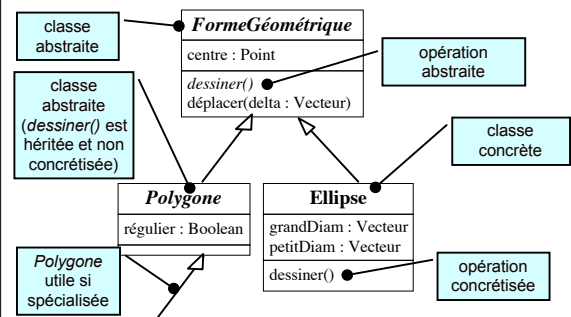
## Modèle structurel

### Opérations abstraites

- Une *opération abstraite* est une opération n'admettant pas d'implémentation : au niveau de la classe dans laquelle est déclarée, on ne peut pas dire comment la réaliser.
- Les opérations abstraites sont particulièrement utiles pour mettre en œuvre le polymorphisme.
- Toute classe concrète sous-classe d'une classe abstraite doit "concrétiser" toutes les opérations abstraites de cette dernière.

## Modèle structurel

### Classes abstraites



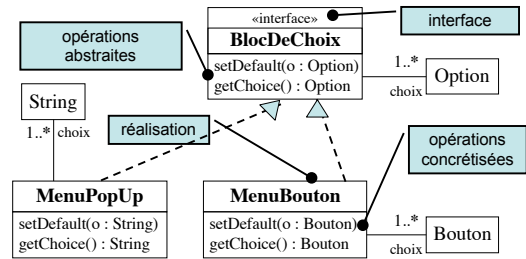
## Modèle structurel

### Interfaces

- Une *interface* est une collection d'opérations utilisée pour spécifier un service offert par une classe.
- Une interface être vue comme une classe sans attributs et dont toutes les opérations sont abstraites.
- Une interface est destinée à être “réalisée” par une classe (celle-ci en hérite toutes les descriptions et concrétise les opérations abstraites).
- Une interface peut en spécialiser une autre, et intervenir dans des associations avec d'autres interfaces et d'autres classes.

## Modèle structurel

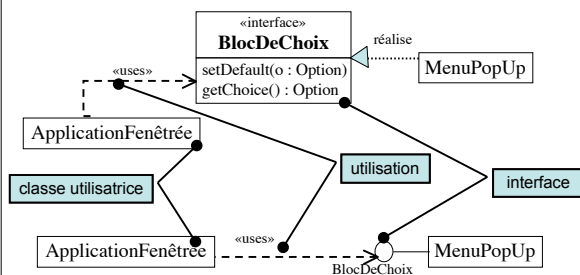
### Interfaces



## Modèle structurel

### Interfaces

Deux notations pour l'utilisation d'une interface



## Modèle structurel

### Heuristiques d'élaboration du modèle structurel

- Bien comprendre le problème
- Faire simple
- Bien choisir les noms
- Bien expliciter les associations
- Ne pas trop “généraliser”
- Relire
- Documenter

De nombreuses révisions sont nécessaires !