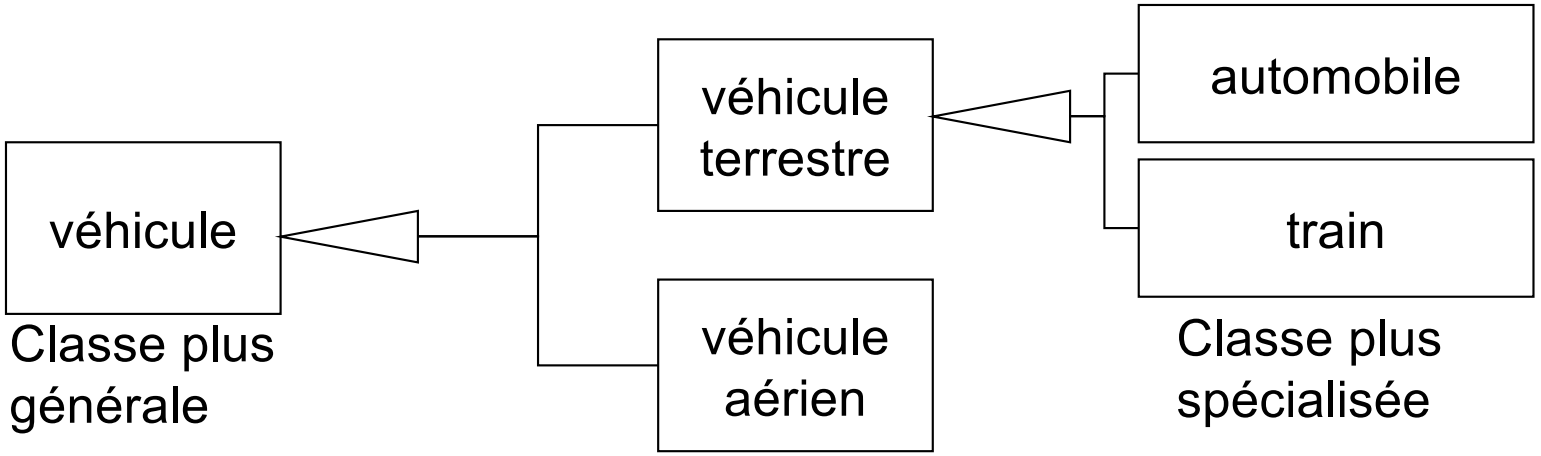
SyML/UML
Hiérarchies de blocs/classes

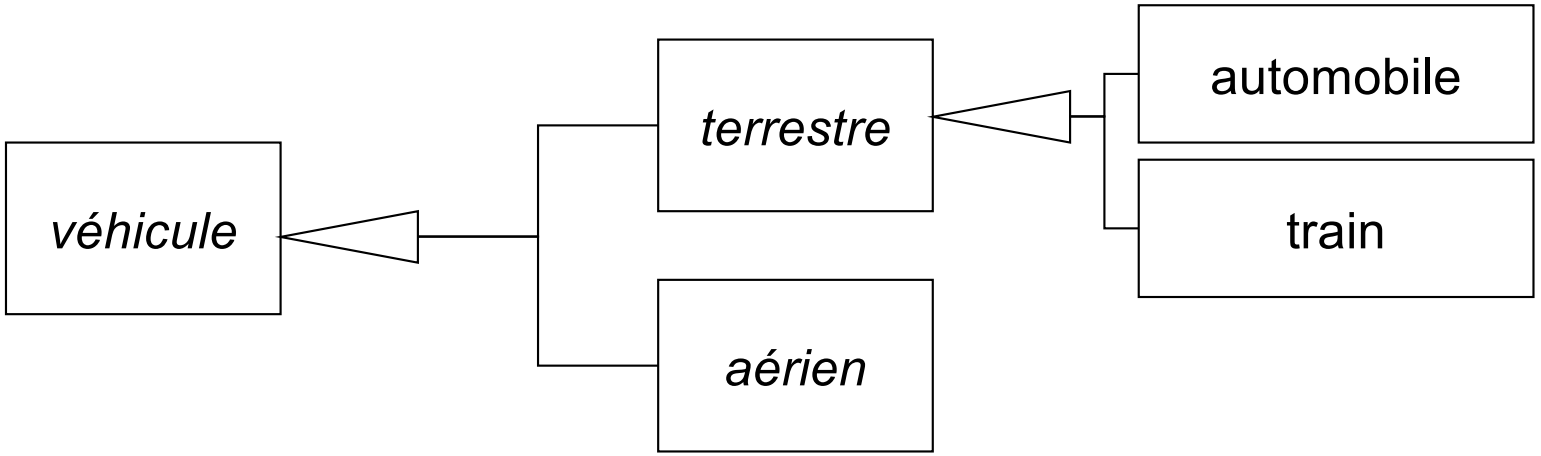
Hiérarchies de classes

- Arborescences de classes d'abstraction croissante :



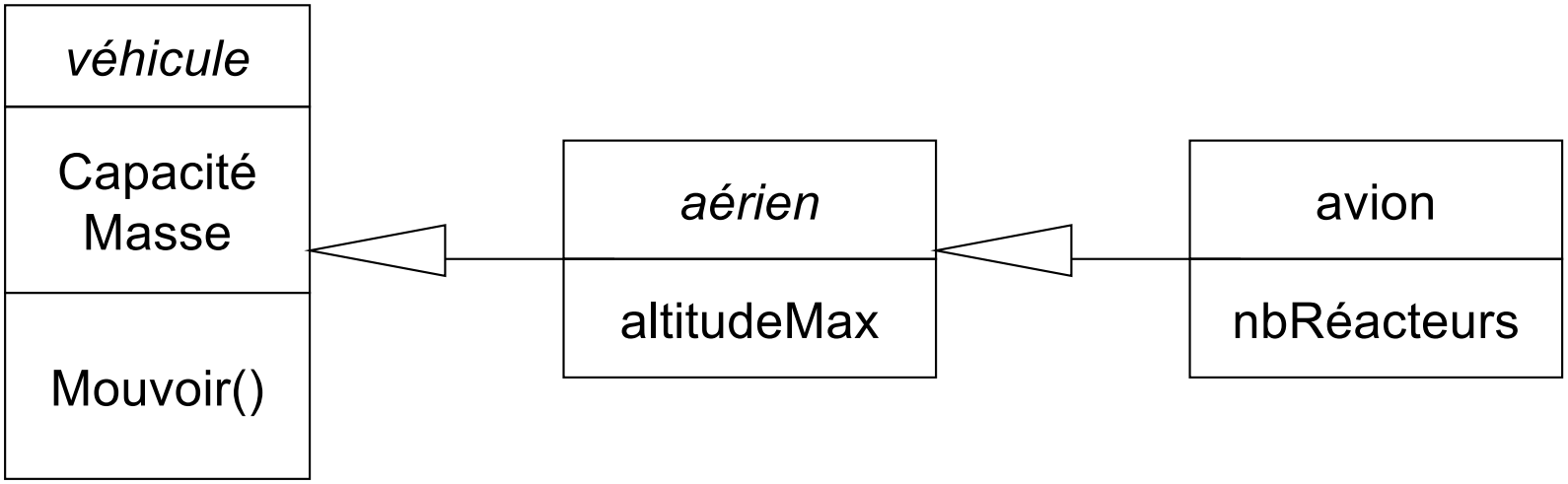
Classes abstraites

- Les classes les plus générales peuvent être définies comme abstraites à savoir non instanciables directement
- Elles ne servent que de spécification des propriétés générales communes à leurs sous-classes
- Par convention le nom des classes abstraites est en italique (avec la mention {abstrait} facultative)



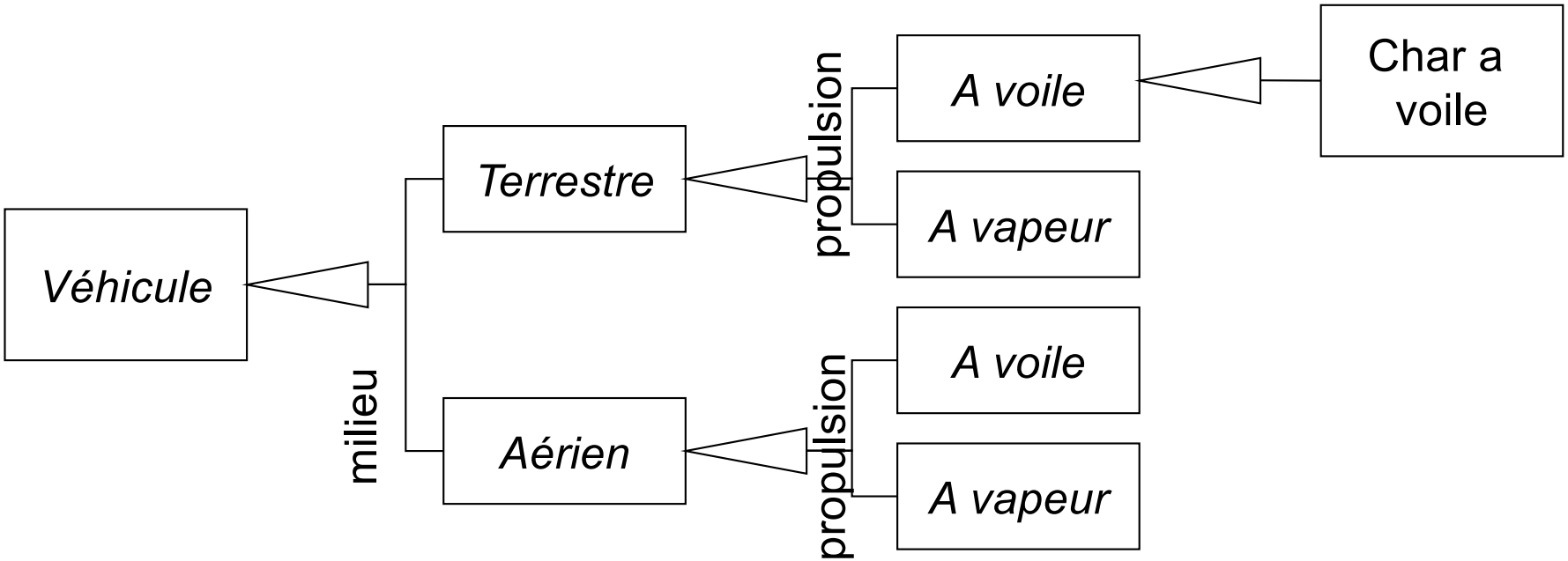
Propagation des propriétés

- Les classes plus spécialisées reprennent les propriétés (attributs et opérations) des classes plus générales et peuvent rajouter les leurs propres :



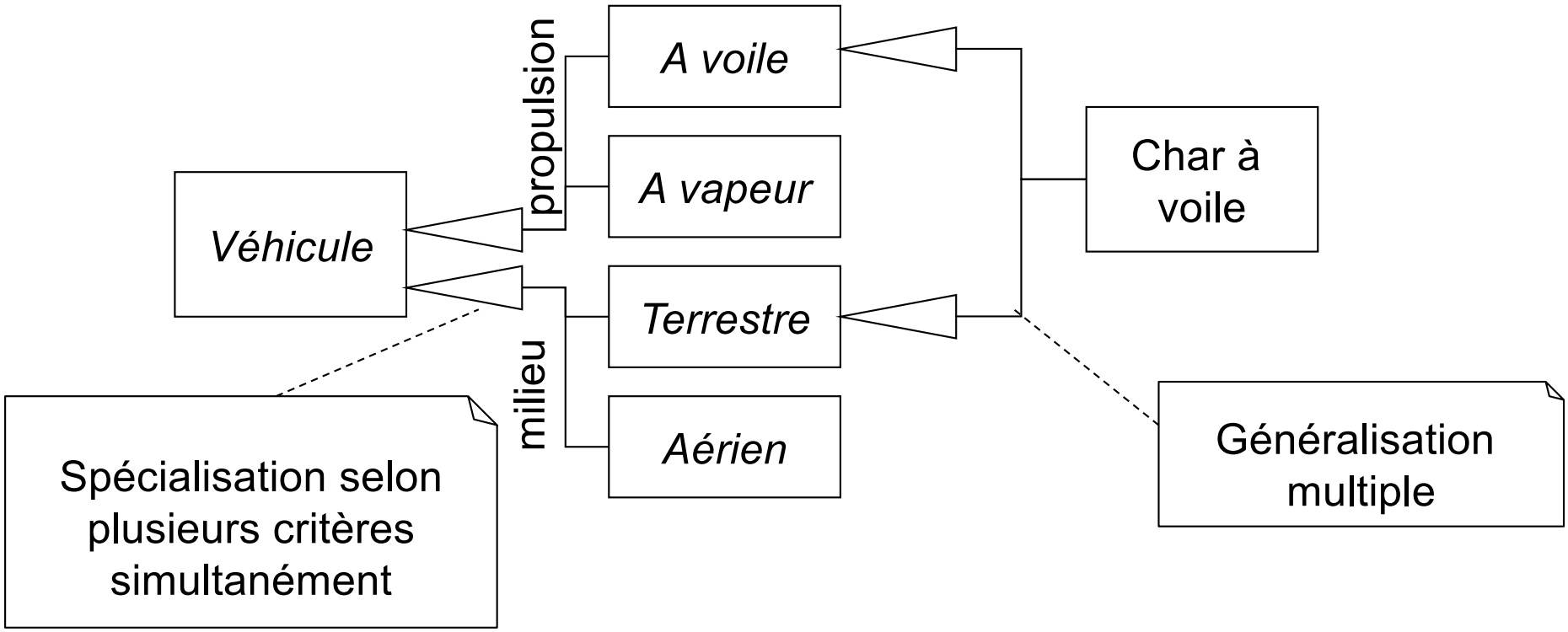
Critères de spécialisation

- Lorsqu'un critère ou discriminant sert à spécialiser une classe générale, il est indiqué sur la relation de généralisation. La classification suivant plusieurs critères successifs peut alors induire des duplicata préjudiciables à la maintenabilité du modèle :



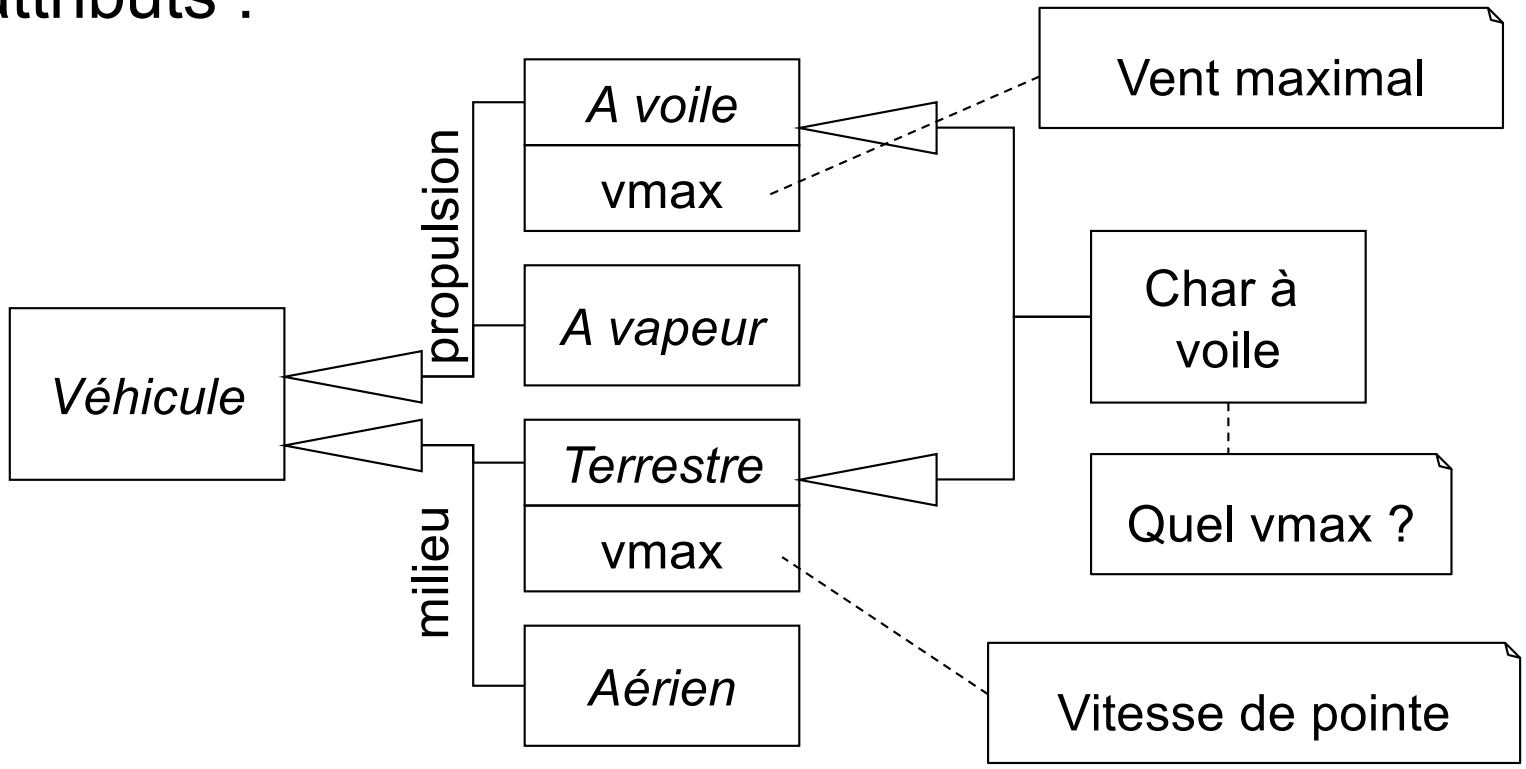
Critères de spécialisation, Généralisation multiple

- UML permet donc la spécialisation d'une classe générale selon plusieurs critères simultanément, ainsi que la généralisation multiple :



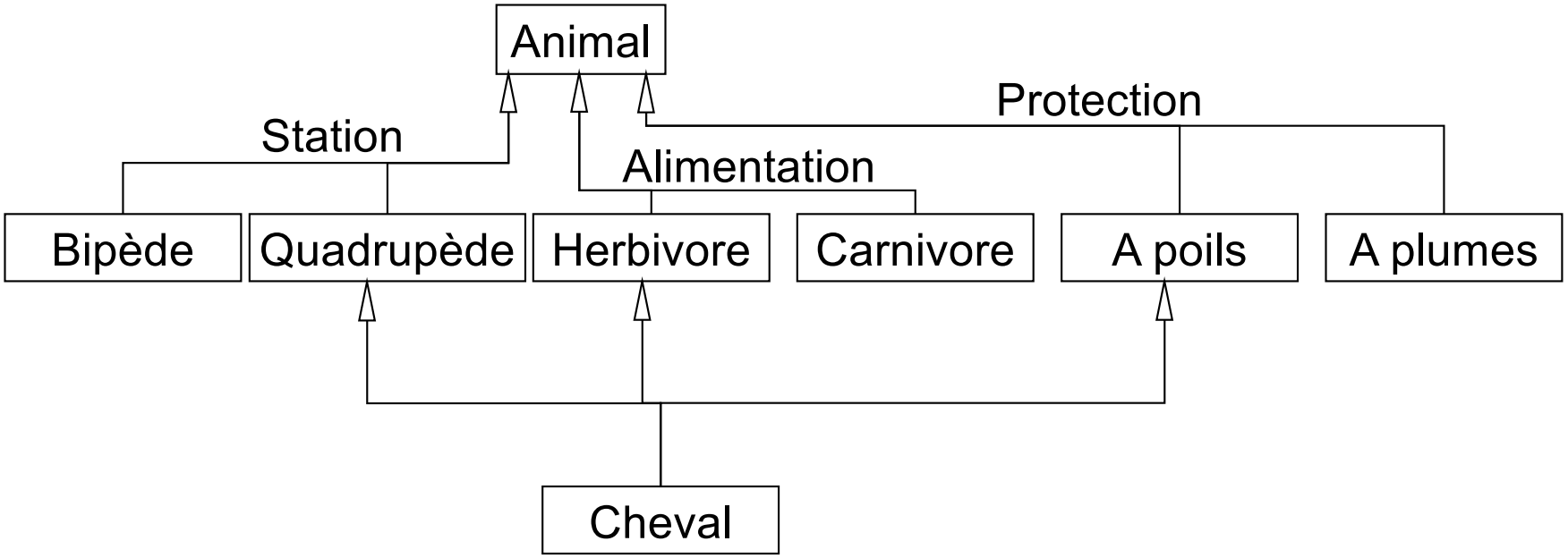
Critères de spécialisation, Généralisation multiple

- La généralisation multiple est à manipuler avec précautions car peut conduire à des collisions de noms d'attributs :



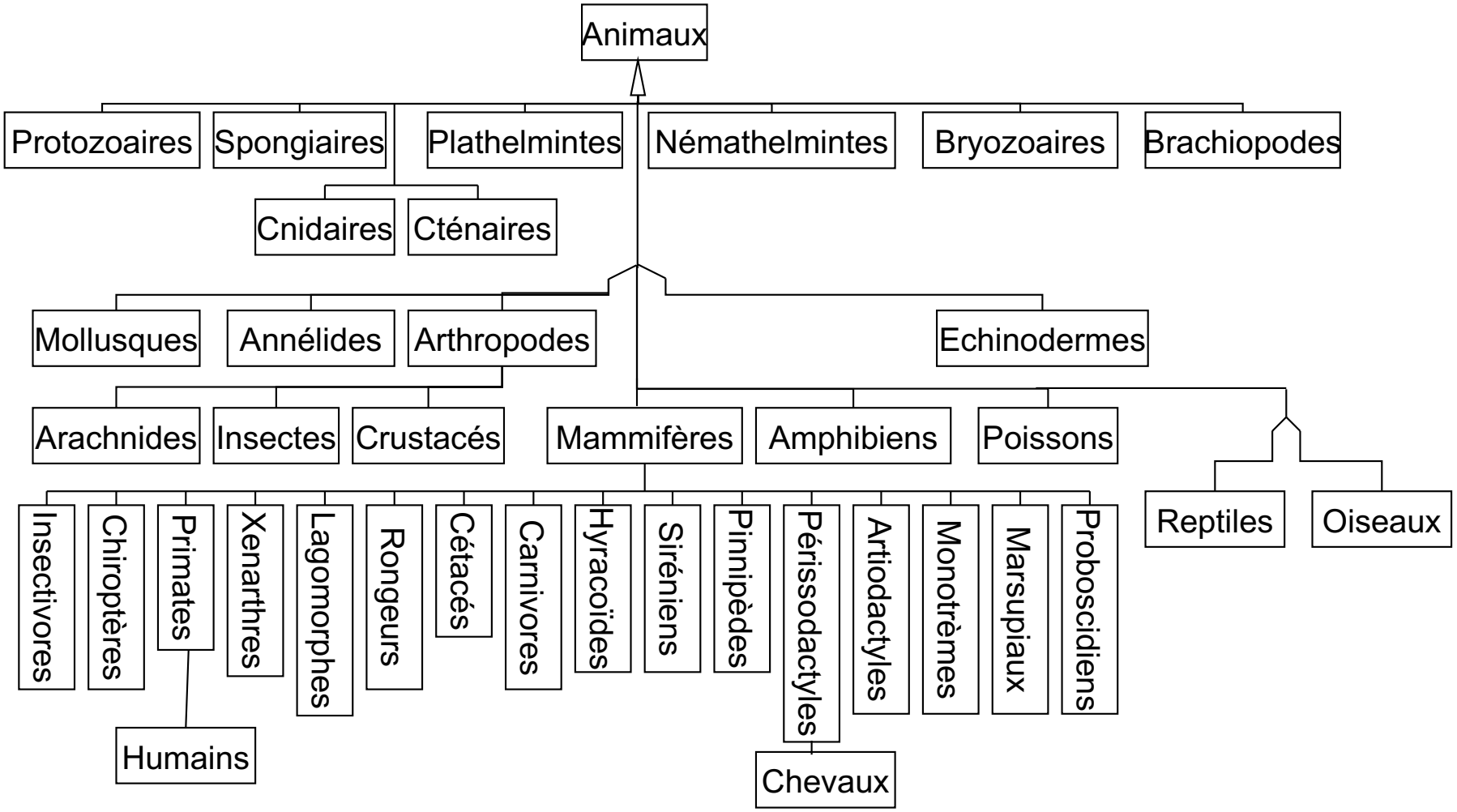
Critères de spécialisation, Généralisation multiple

- La généralisation multiple correspond donc à une spécialisation selon plusieurs critères simultanément, fréquemment utilisée en analyse objet car permettant de cerner rapidement les caractéristiques essentielles des classes spécialisées :



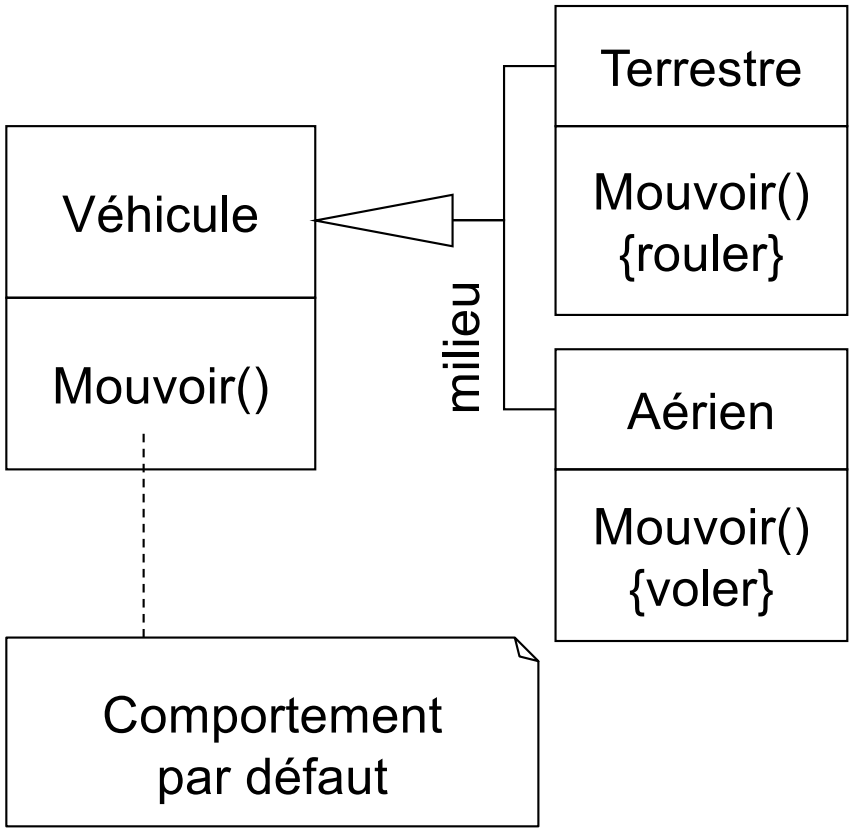
Critères de spécialisation, Généralisation multiple

- Extrait de la classification zoologique réelle (mono-critère)

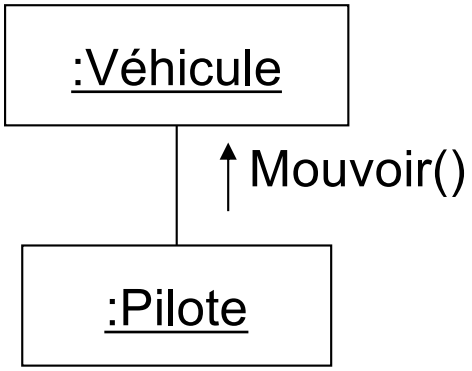


Surcharge des opérations, polymorphisme

- Les opérations peuvent être redéfinies dans les sous-classes. Elles sont alors dites surchargées

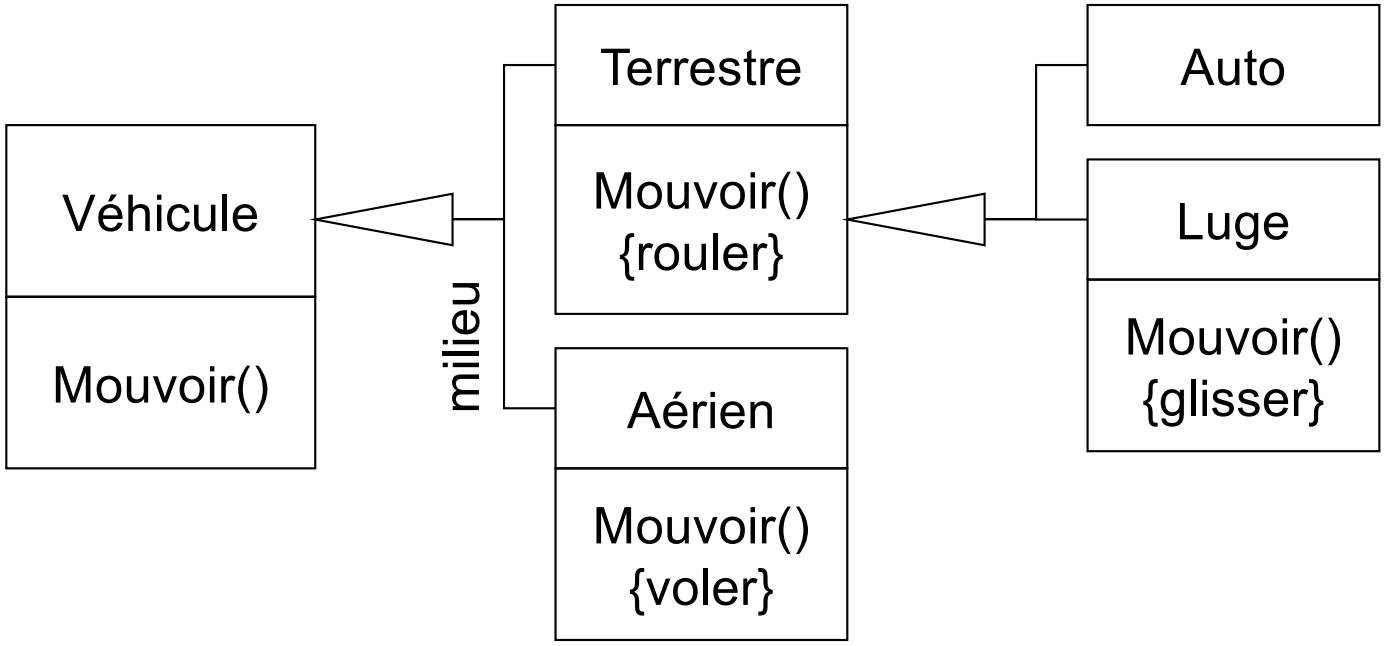


Permet de s'adresser à un ou plusieurs objets sans en connaître les détails (polymorphisme = comportement différent en réponse à un même message)



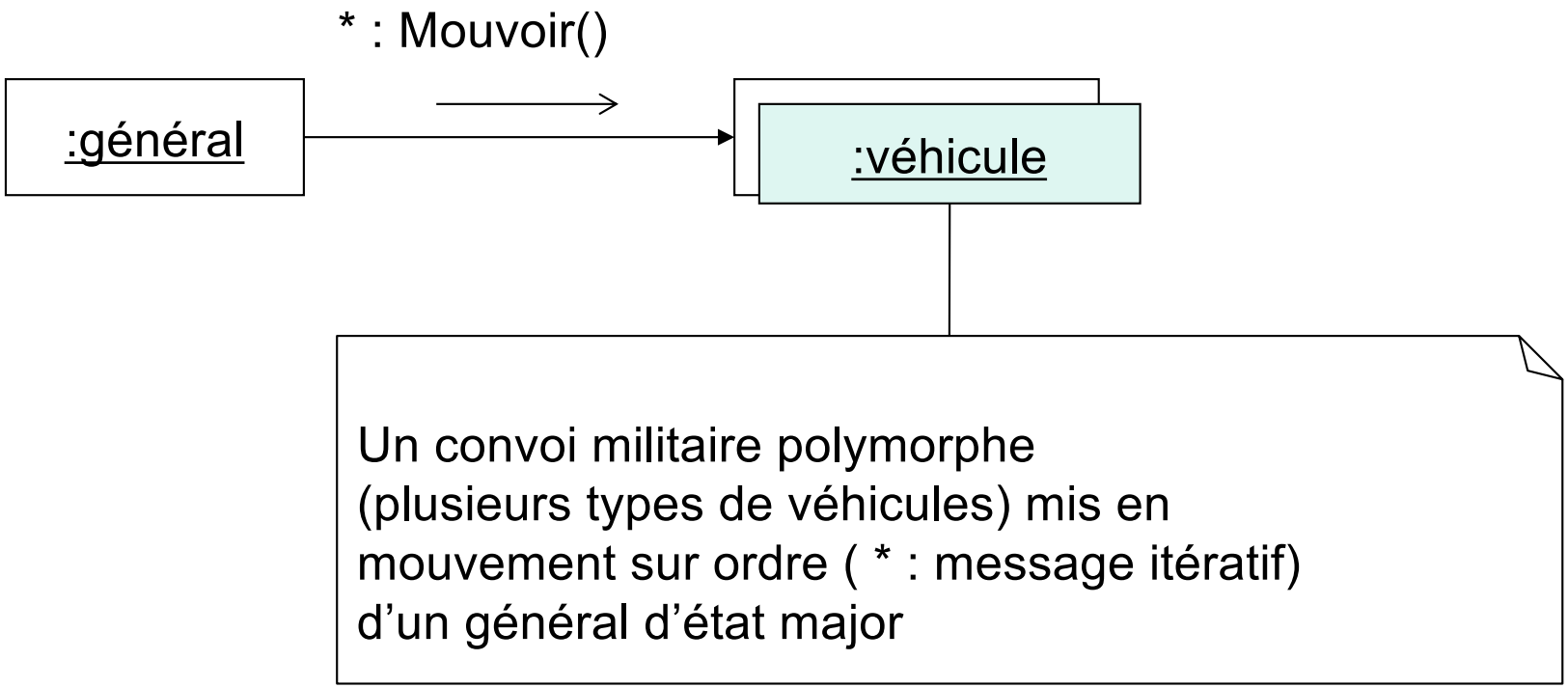
Polymorphisme

- Le polymorphisme favorise l'évolutivité du logiciel par simple ajout de nouvelles sous-classes avec leurs opérations éventuellement spécifiques :



Polymorphisme

- Il est alors possible de s'adresser à une collection d 'objets sans en connaître précisément la sous-classe.



Contraintes sur les généralisations

- Différentes formes de contraintes peuvent être formulées :

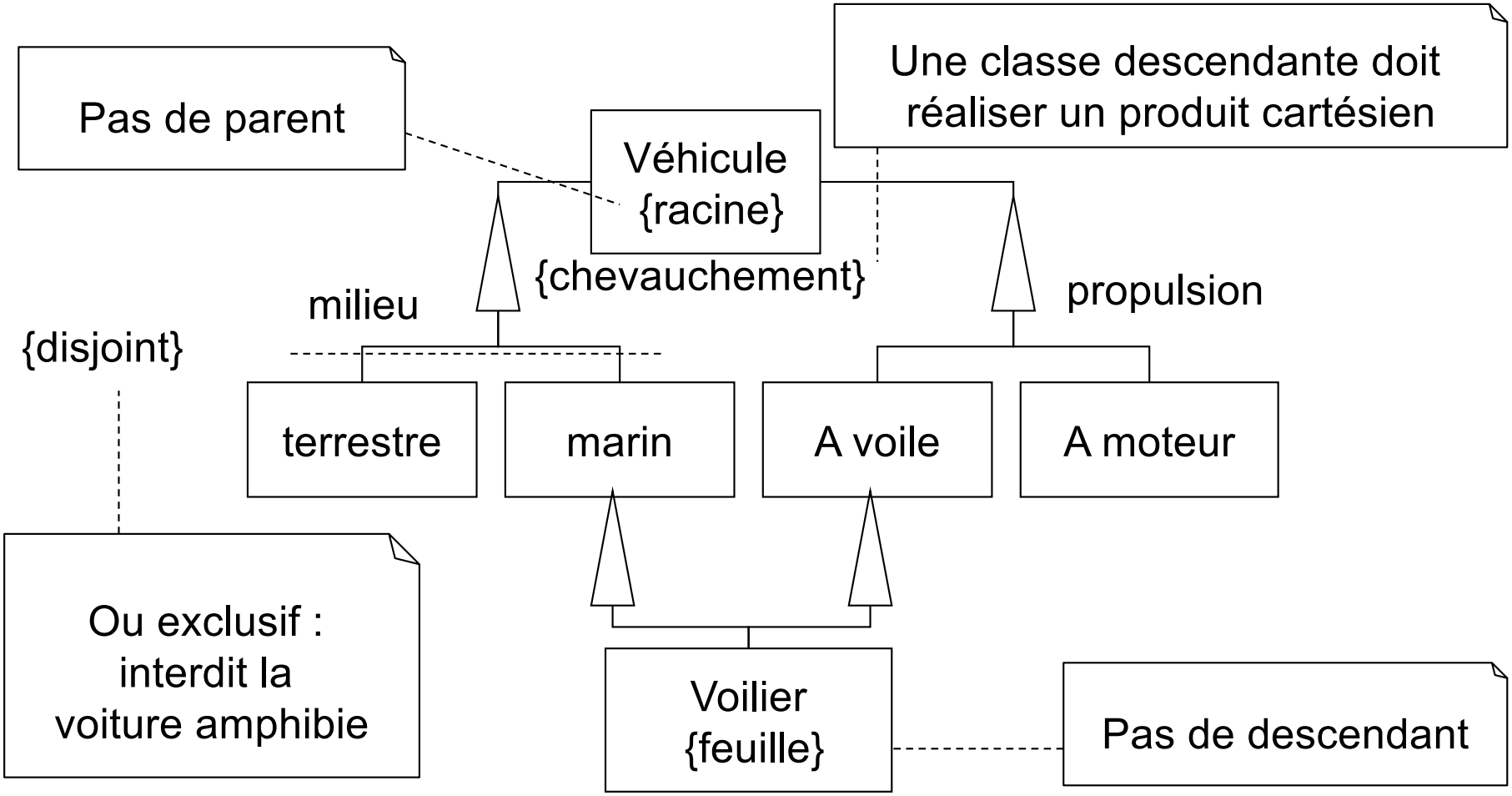
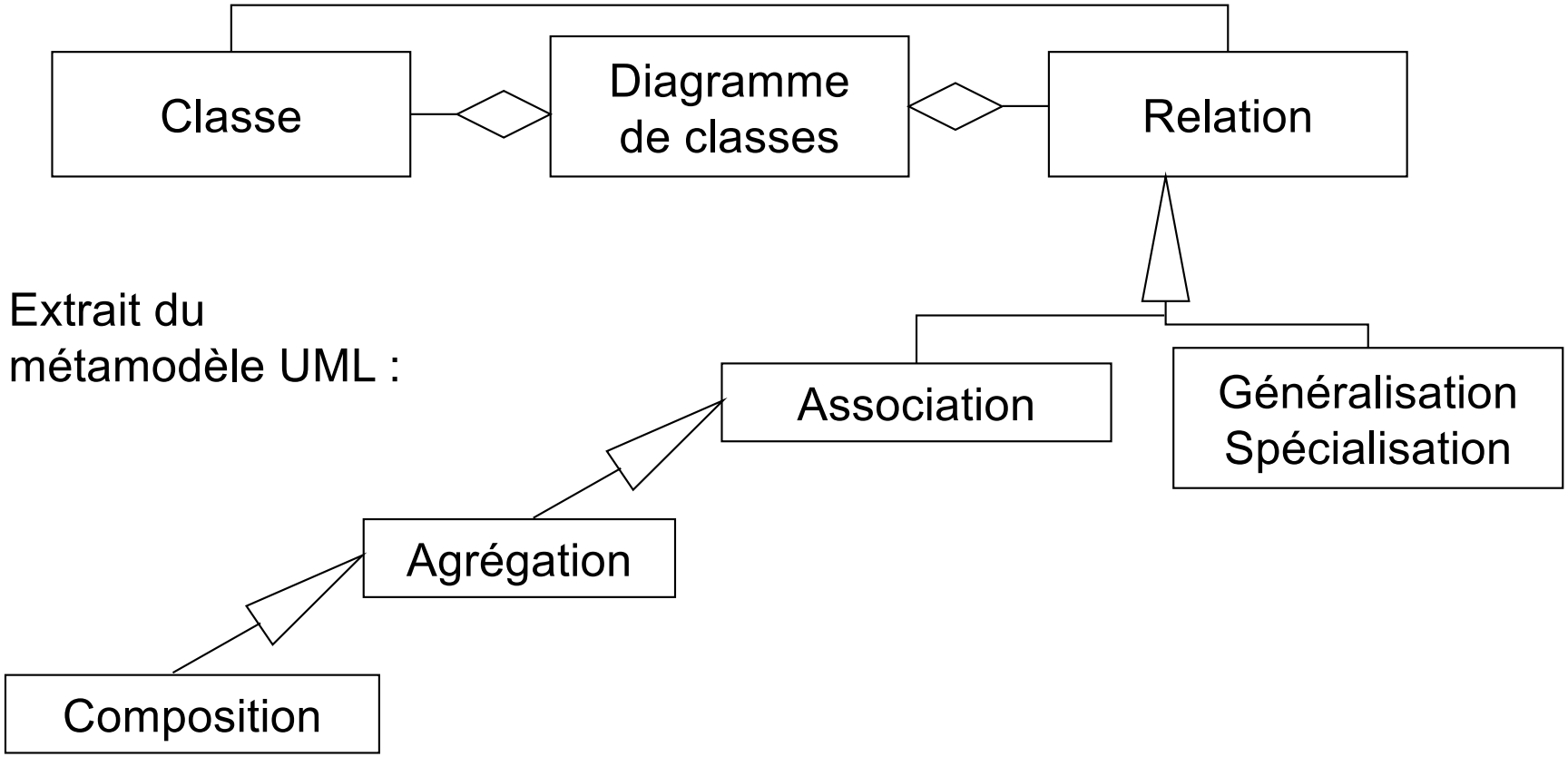


Diagramme de classes : conclusion

- Le plus important type diagramme UML. Représente le modèle statique de l'application.



Adaptations SysML

- Sur ce sujet, strictement aucune (généralisation / spécialisation : « héritage », utilisé exactement dans le même sens qu'en UML, pour signifier « est une sorte de »)
- Moins en évidence en revanche l'utilisation de l'héritage pour faire du polymorphisme, typique de l'héritage/surcharge en langage objet.