IUT Lyon 1
l'excellence technologique

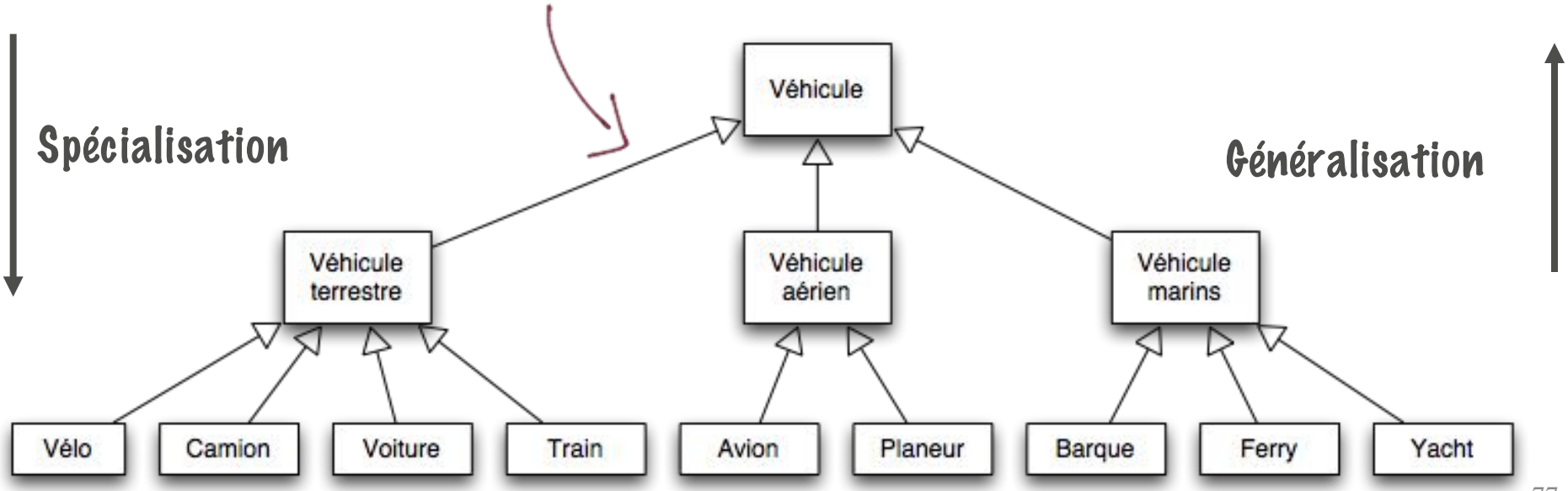
CHAPITRE IV
Héritage & Polymorphisme

Département GEII –POO

Héritage & Polymorphisme

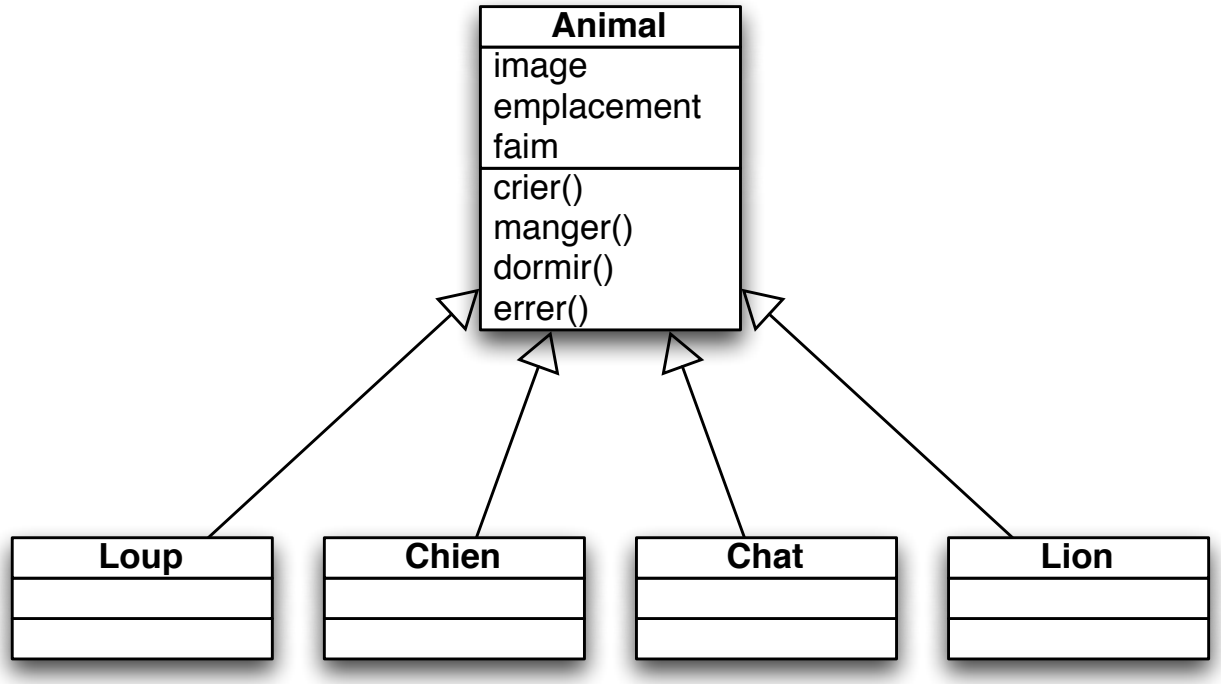
- Héritage
  - Les classes sont organisées hiérarchiquement
  - Une classe hérite d'une super-classe (ou classe mère)
    - Relation "EST UN"
    - Exemples
      - Un Avion EST UN Véhicule aérien
      - Un Véhicule aérien EST UN Véhicule

Représentation UML de l'héritage



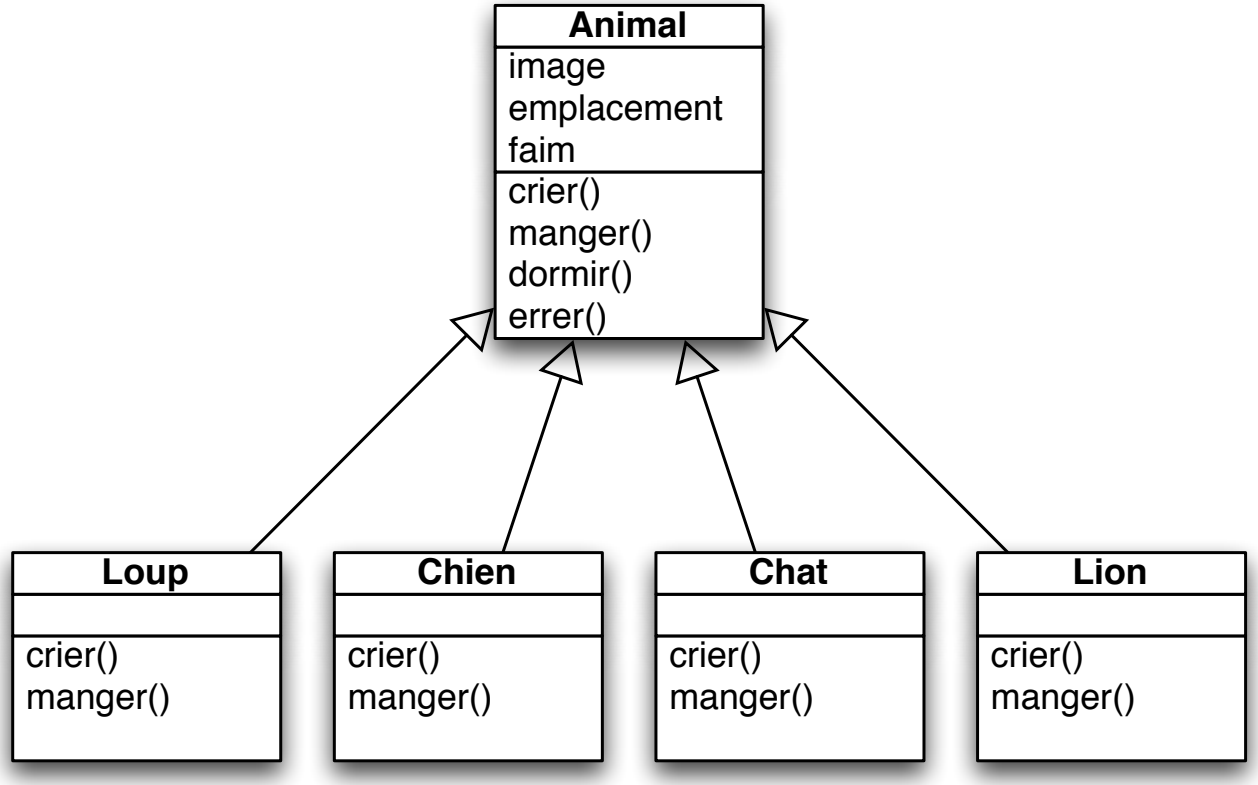
Héritage & Polymorphisme

- Héritage
  - Permet de regrouper dans une classe tout ce qui est commun à plusieurs sous-classes
  - Cela permet d'éviter de dupliquer du code
  - Les sous-classes héritent des champs et des méthodes de leur super-classe



Héritage & Polymorphisme

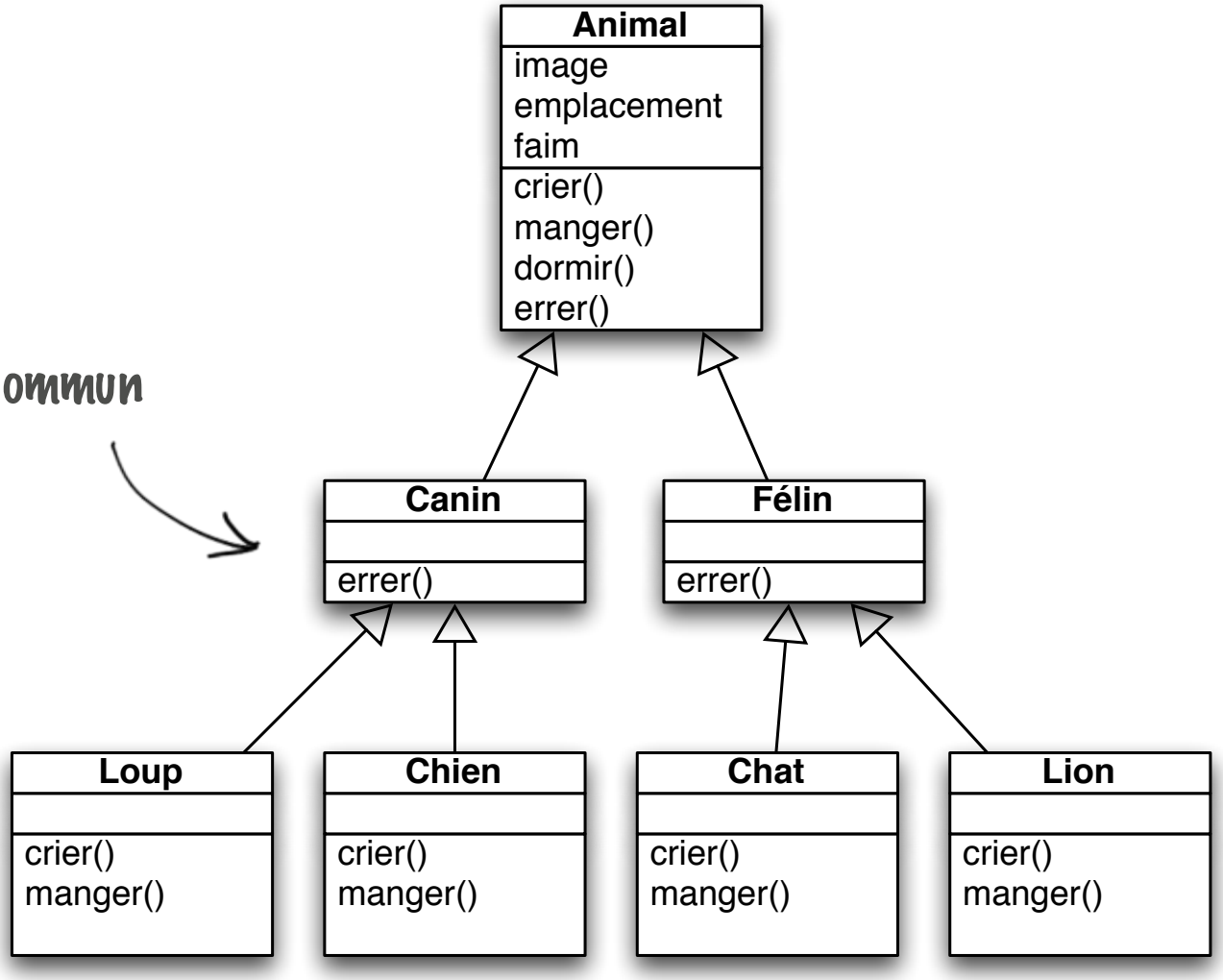
- Héritage
  - Spécialisation
    - Les sous-classes peuvent redéfinir (*override*) les méthodes de leur super-classe
    - Chaque animal possède sa propre façon de crier et de manger



Héritage & Polymorphisme

- Héritage

Factorisation du code commun



Héritage & Polymorphisme

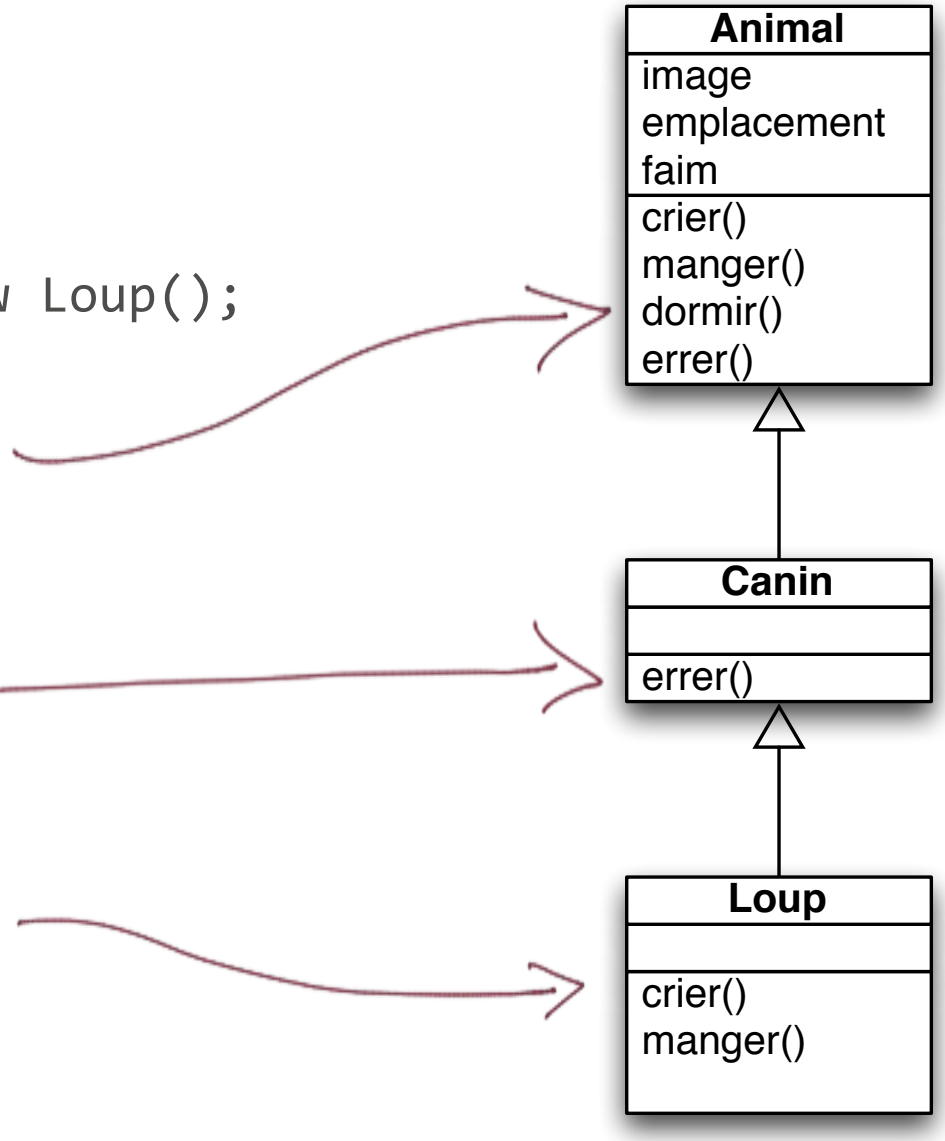
- Héritage
  - Quelle méthode est appelée ?

```
Loup loup = new Loup();

loup.dormir();

loup.errer();

loup.crier();
```

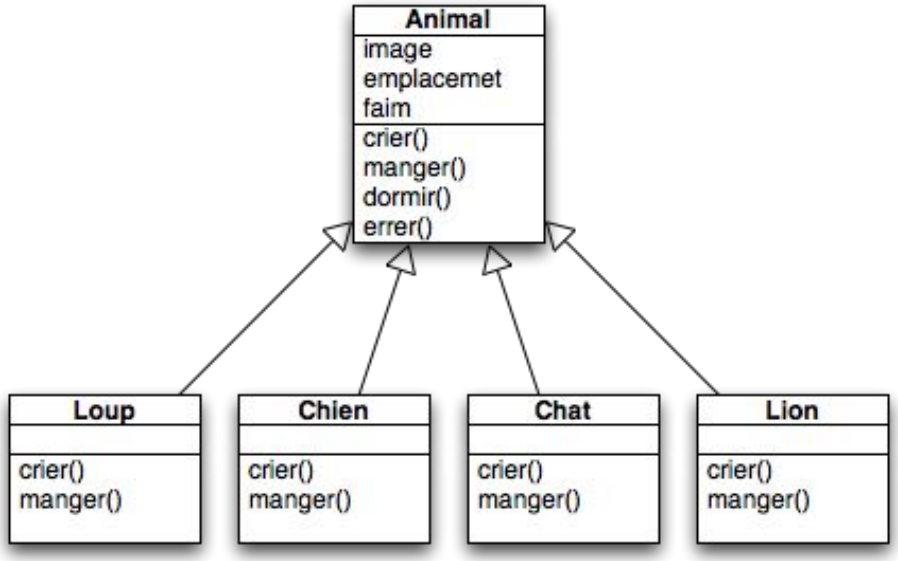


Héritage & Polymorphisme

- Polymorphisme
  - Possibilité de manipuler des objets en utilisant une variable référence d'un super-type

```
Animal [] animaux = new Animal[4];
animaux[0] = new Chien();
animaux[1] = new Loup();
animaux[2] = new Chat();
animaux[3] = new Lion();

for(Animal animal : animaux){
  animal.crier();
  if(animal.faim > SEUIL)
    animal.manger();
}
```



Toutes les méthodes et variables définies dans la classe Animal sont accessibles depuis la variable animal

Héritage & Polymorphisme

- Polymorphisme
  - On peut avoir des arguments et des types de retour polymorphes

```
class Veto {
  public void piquer(Animal a){
    //Faire des choses horribles à l'animal
    a.crier();
  }

  public Animal mettreBas(Animal a){
    Animal nouveauNe = a.mettreBas();
    return nouveauNe;
  }
}
```