Analyse et conception

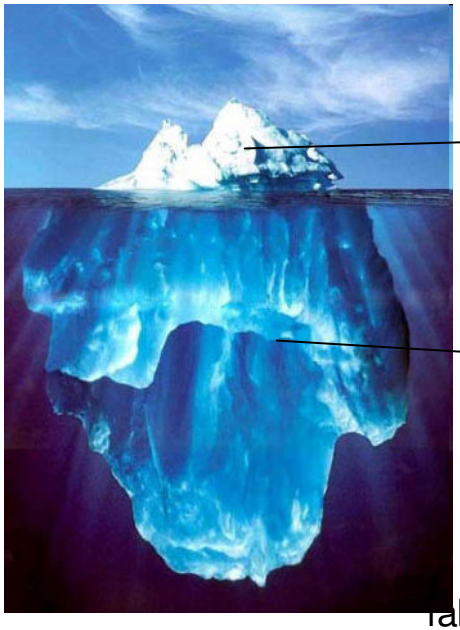
Cours de 1e année ingénieur

fabien.romeo@fromeo.fr
http://www.fromeo.fr

Diagramme de classes : Interface

Encapsulation

- [Parnas72] information hiding
  - The *secrets* of a module: Design decisions that can be changed without affecting any other module
  - The assembly of a system made of modules is usually best done if the interfaces are simple and well-understood by all involved



Interface (définitions)

- imagée : c'est la partie émergée de l'iceberg.
- d'un objet (exécution) : un ensemble de messages compris par un objet.
- d'une classe (conception) : l'interface d'une classe fournit sa vue externe : on augmente l'abstraction en cachant la structure et les secrets du comportement de la classe.
- UML : *an interface is a kind of classifier that represents a declaration of a set of coherent public features and obligations. An interface specifies a contract; any instance of a classifier that realizes the interface must fulfill that contract.*

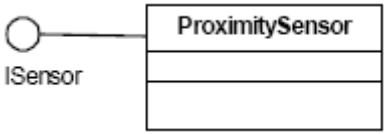
Notion d'interface

- Les interfaces déclarent des opérations (et des *propriétés**) publiques offertes par une classe
- Déclaration => elles ne sont pas instanciables !
- ≈ classes abstraites pour lesquelles **toutes** les opérations sont abstraites
- Une classe qui réalise une interface doit présenter les méthodes publiques qui se conforment à la spécification de l'interface
- Une classe peut posséder plusieurs interfaces (regroupement de services liés)

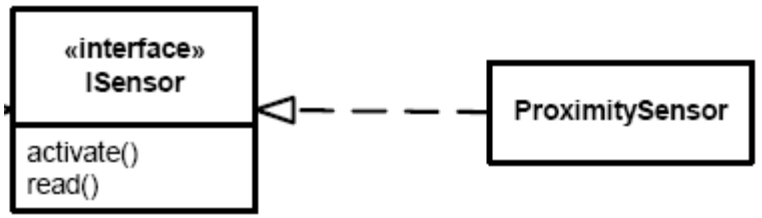
* A property declared on an Interface does not necessarily imply that there will be such a property on a classifier realizing that Interface (e.g., it may be realized by equivalent get and set operations).

Interface fournie

- Expose un ensemble de services mis à disposition des clients d'une classe
- Permet d'établir un faible couplage entre le client et la classe qui implémente réellement les services fournis
- Composition propre des classes du système
- Favorise la substituabilité des classes pour l'implémentation des différents services (maintenance, test, adaptabilité, …)



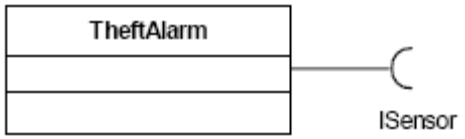
Lollipop notation



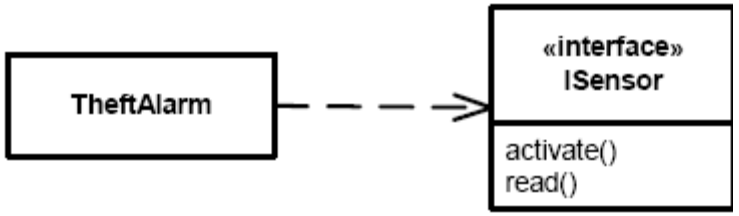
realization

Interface requise

- Duale à la notion d'interface fournie
- Déclaration explicite des dépendances fonctionnelles requises par une classe pour fonctionner
- Permet de contenir la cohésion d'une classe pour en favoriser la réutilisation



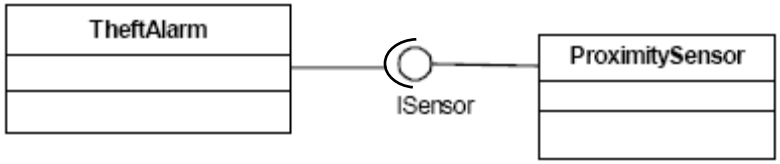
Socket notation

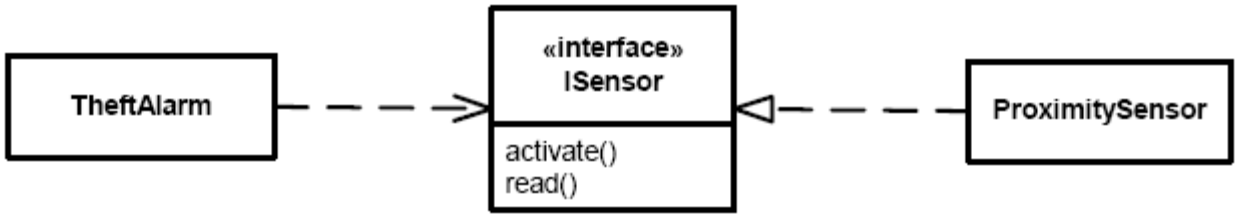


usage

Assemblage par interface

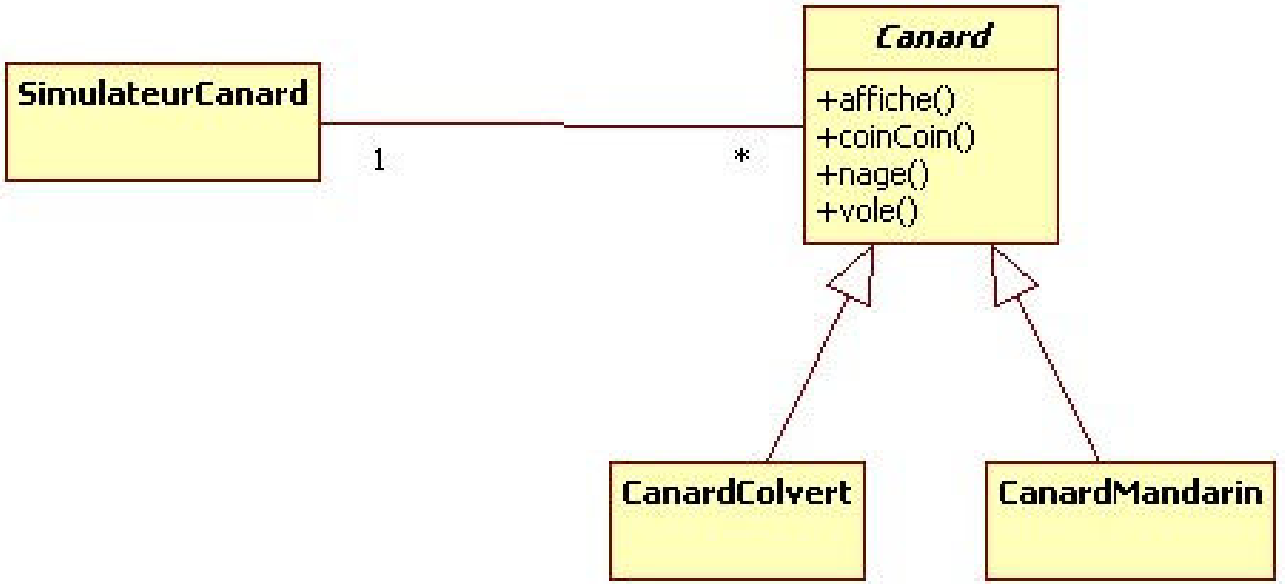
Connector notation



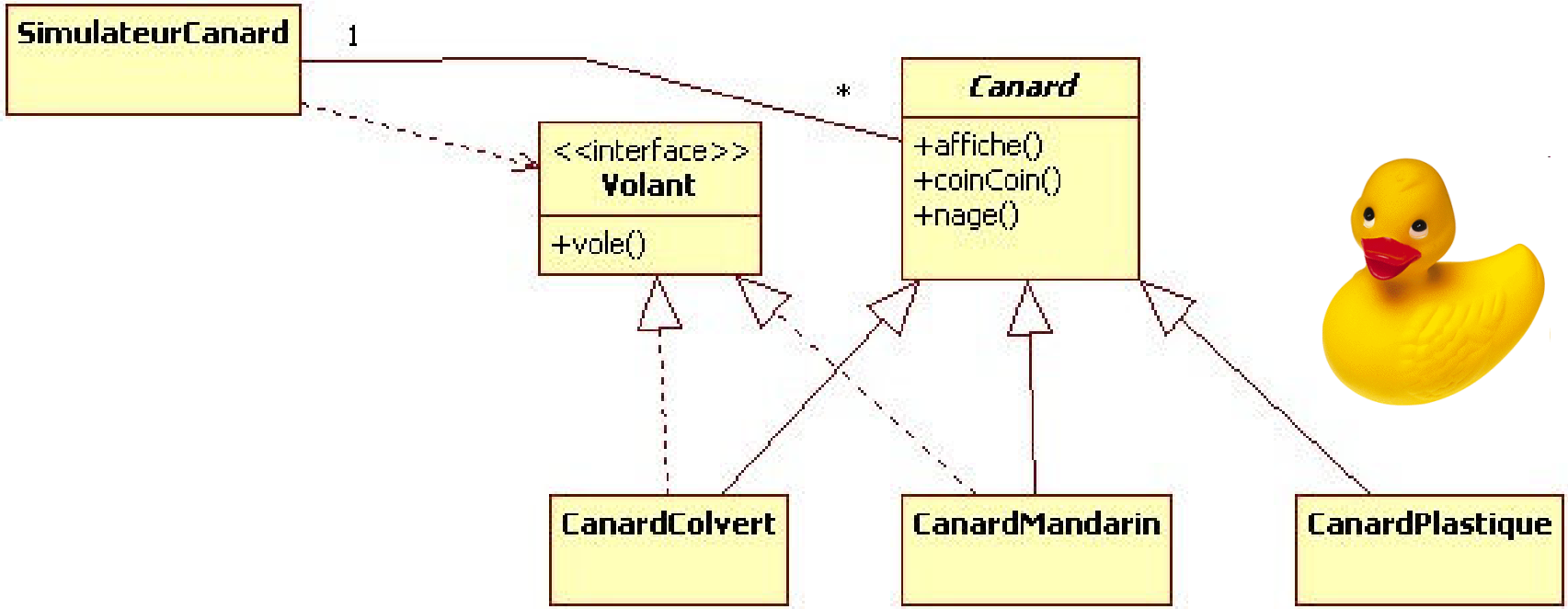


Explicit notation

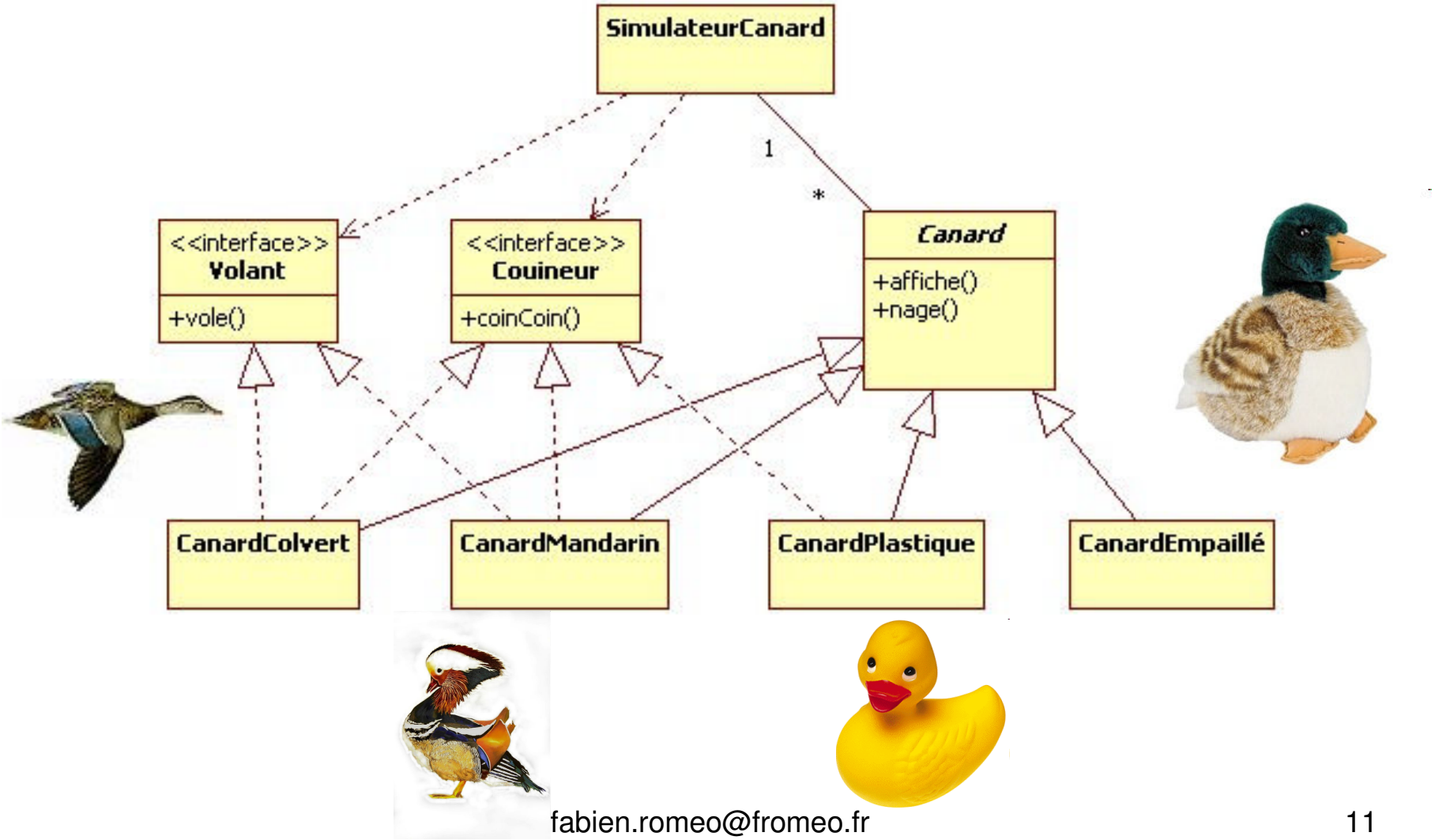
Simulateur de canards : héritage



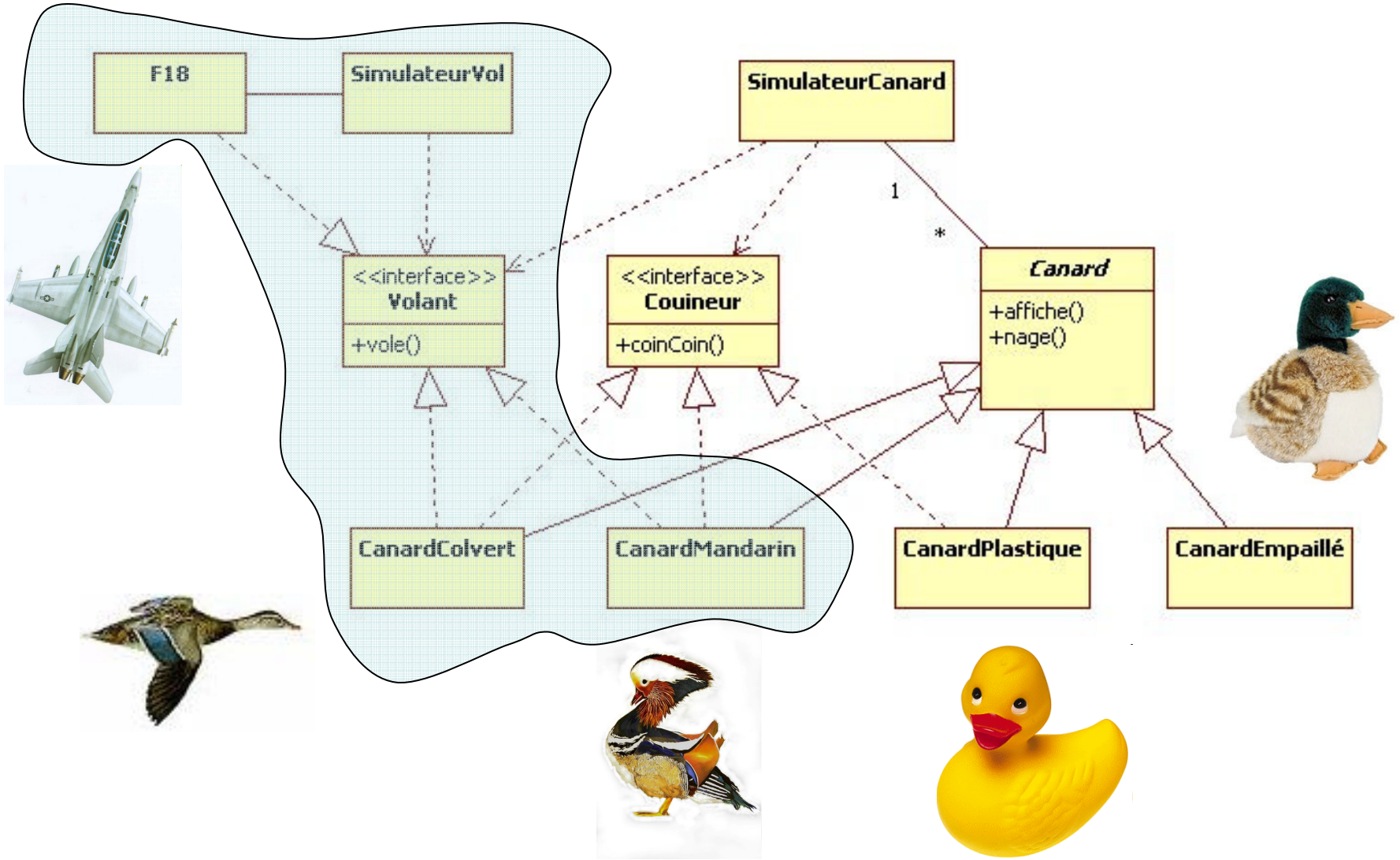
Simulateur de canards : une interface



Simulateur de canards : deux interfaces

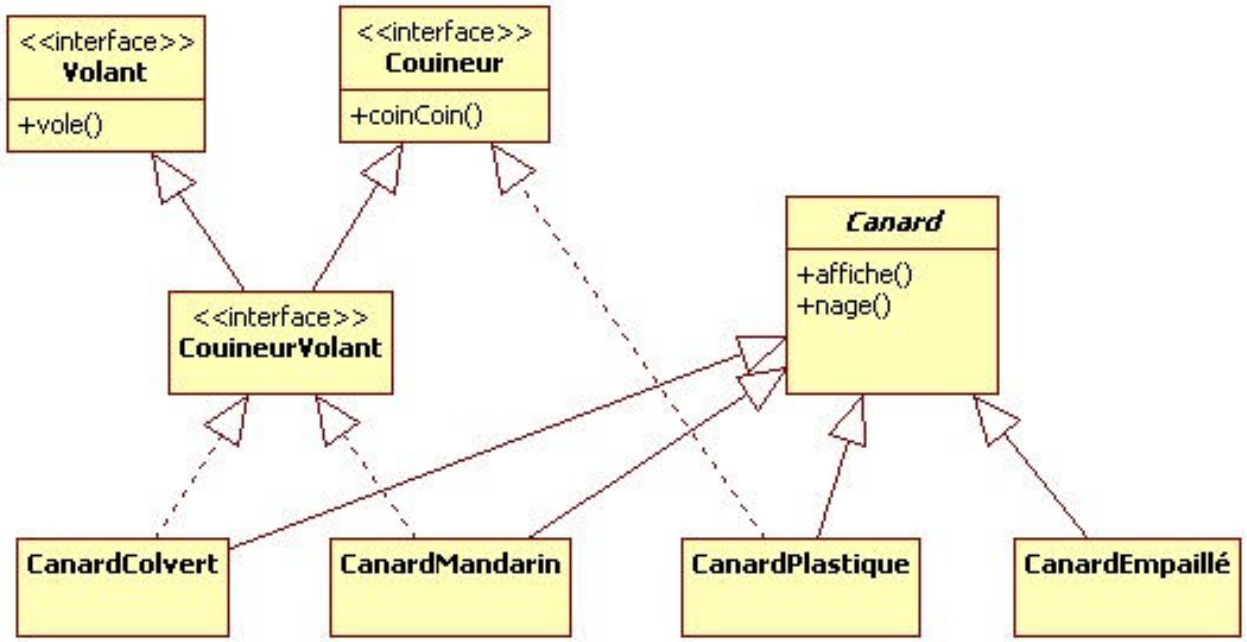


Substitution de classes réalisant une interface



Héritage d'interfaces

- Différent du lien « realize »
- Même notion que l'héritage de classes

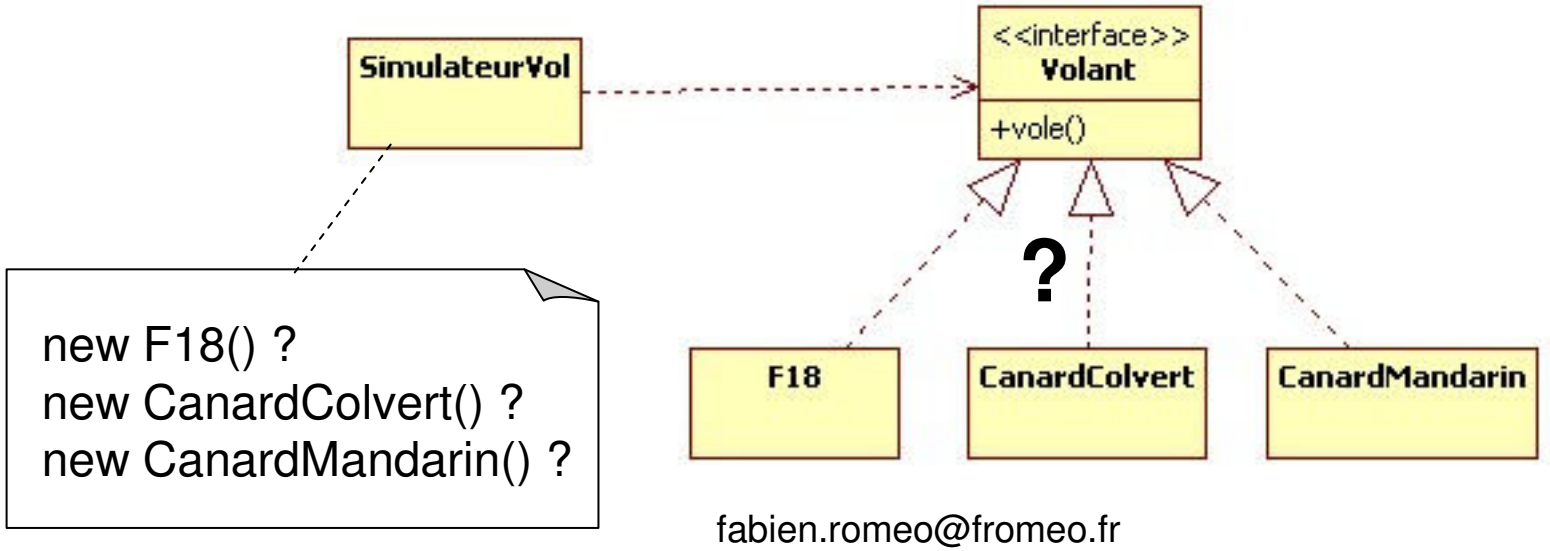


Principe de conception OO

- [GoF95] *Programming to an Interface, not an Implementation*
- Deux avantages
  - les classes clientes n'ont pas besoin de connaître le type spécifique des objets qu'elles manipulent du moment qu'ils se conforment aux interface requises
  - les classes clientes ne reposent pas sur les classes implémentant ces objets mais seulement sur la définition de leurs interfaces

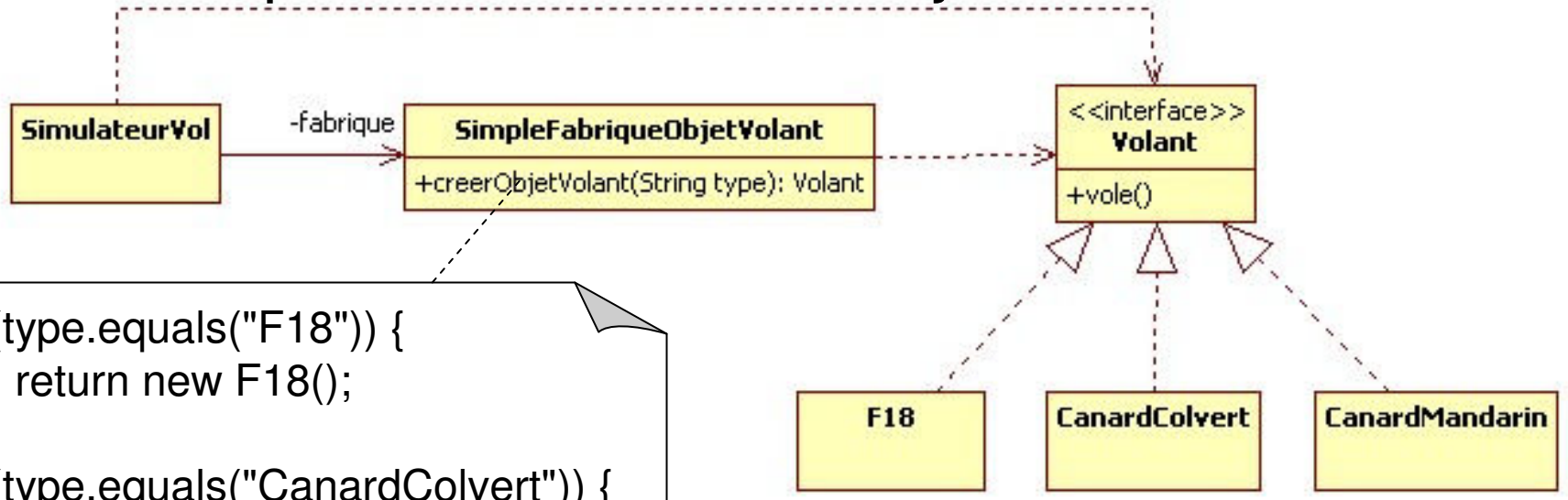
Interface « volant », des objets non identifiés ?

- Le simulateur de vol est lié à l'interface « volant » afin de pouvoir modifier/ajouter/supprimer facilement les objets que le simulateur peut faire voler.
- Cependant il lui faut instancier des classes d'objets volants (i.e. faire des new()) : on perd alors l'intérêt de l'interface puisque la création des objets volants se fait à l'intérieur du simulateur.



Notion de fabrique

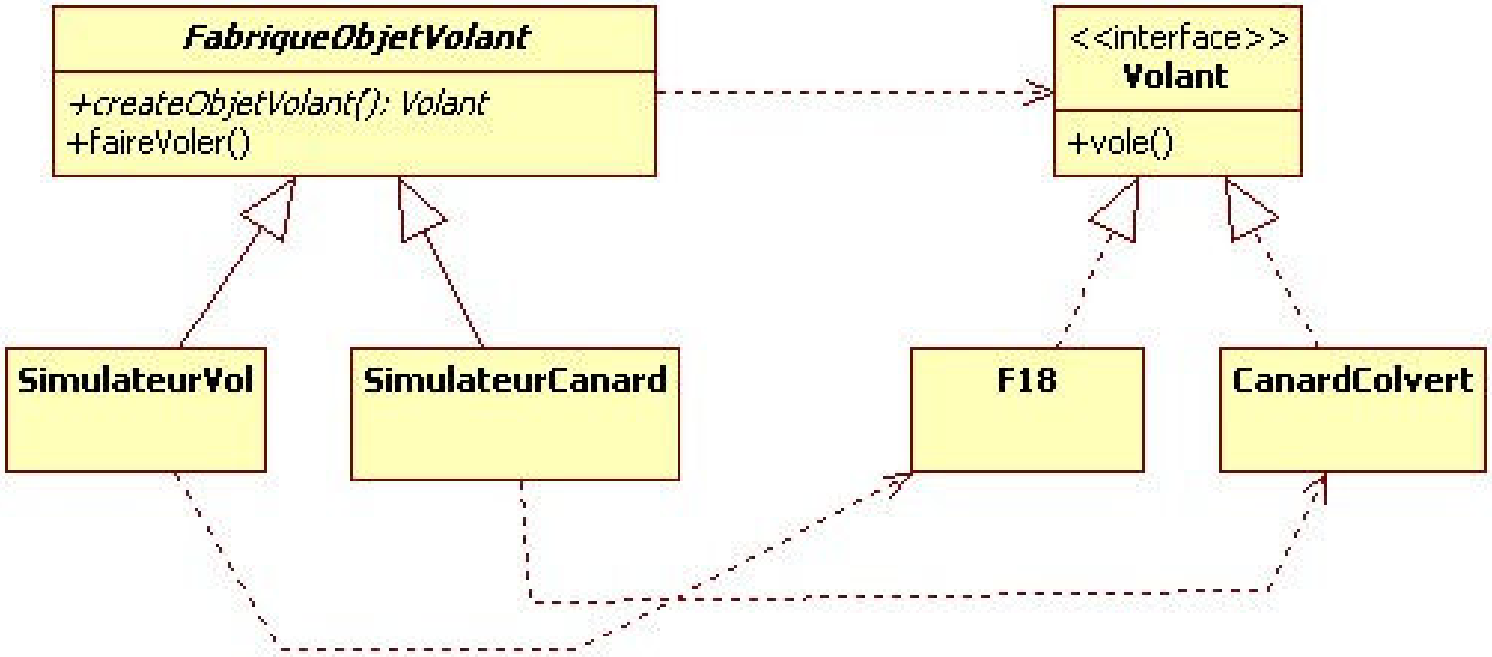
- On délègue la gestion de la création d'objets à une fabrique.
- Une fabrique est une classe qui permet d'encapsuler la création d'objets.



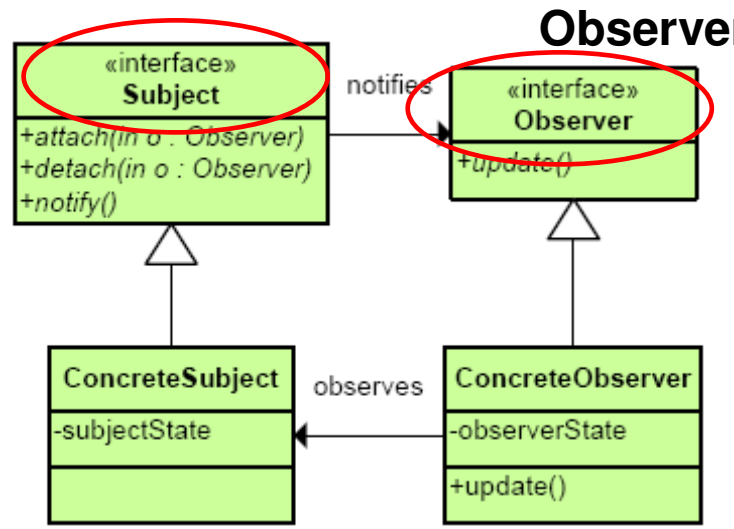

```
if(type.equals("F18")) {
    return new F18();
}
if(type.equals("CanardColvert")) {
    return new CanardColvert();
}
// …
```

Design Pattern Fabrique

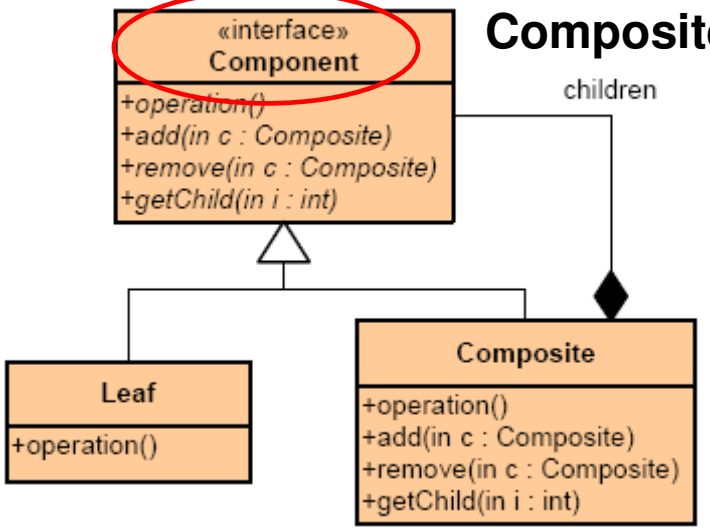
- Le pattern fabrique fournit un framework pour la création d'objets abstraits (interface)
- La création des objets concrets est déléguée à des fabriques concrètes



Interfaces dans les design patterns

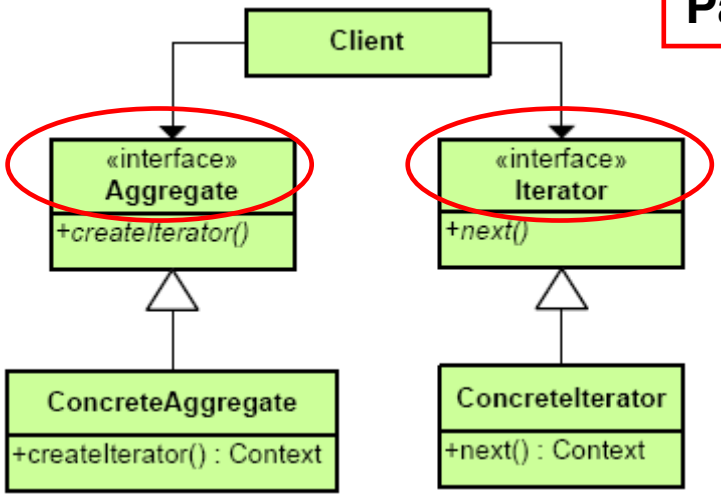


Observer

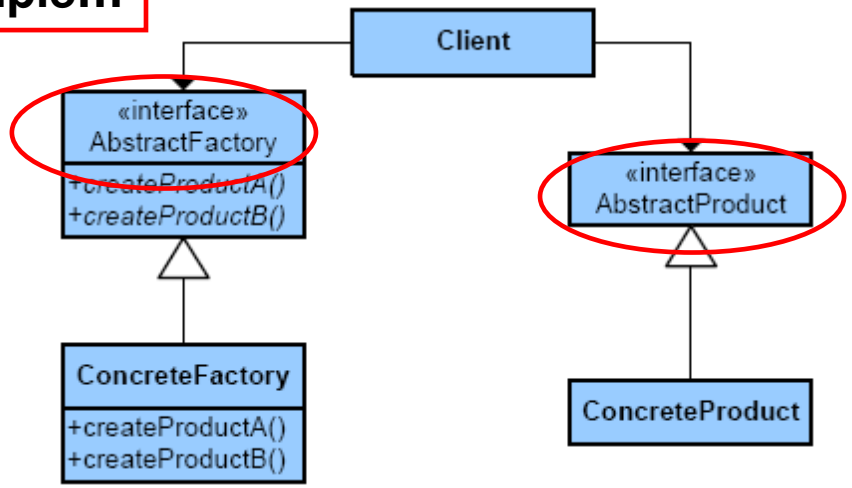


Composite

Par exemple…

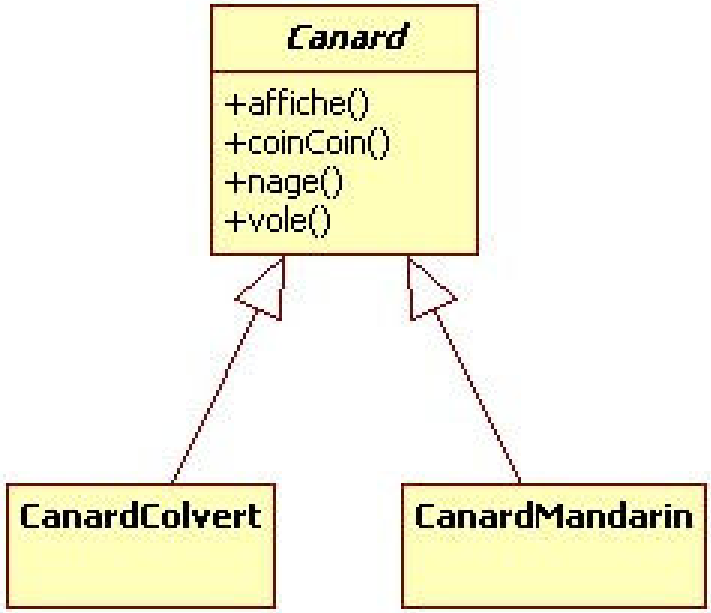


Iterator

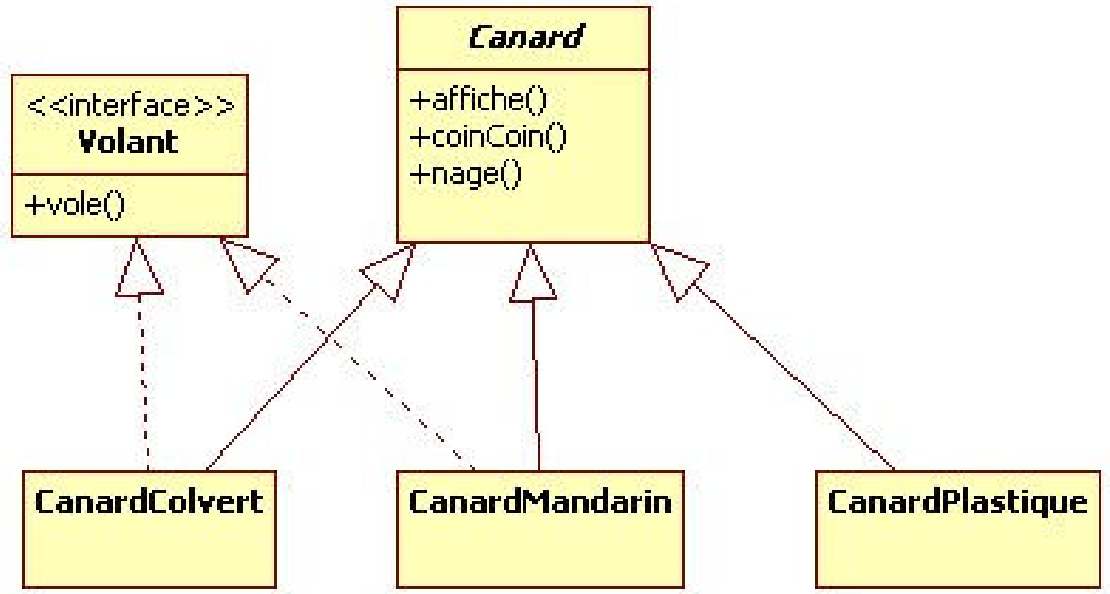


Abstract Factory

Simulateur de canards : héritage



Simulateur de canards : une interface



Simulateur de canards : deux interfaces