

Diagramme de Classe

**Etude d'un système de
réservation de vols**

Plan

- 1) pas a pas vers un Diagramme de classe**
- 2) Généralisation et Réutilisation**
- 3) Paquetage**

- Cette étude de cas concerne un système simplifié de réservation de vols pour une agence de voyages.
- Les interviews des experts métier ont permis de résumer leur connaissance du domaine sous la forme des phrases suivantes

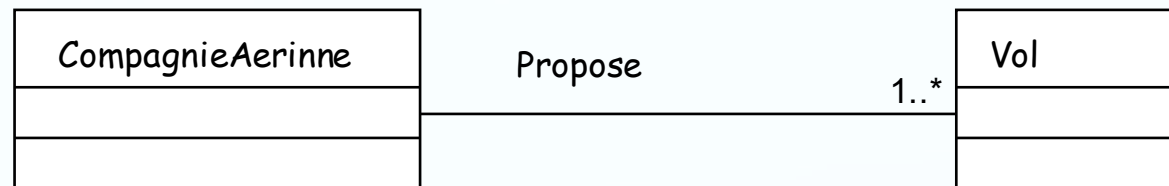
- 1) Des compagnies aériennes proposent différents vols.**
- 2) Un vol est ouvert à la réservation et fermé sur ordre de la compagnie.**
- 3) Un client peut réserver un ou plusieurs vols, pour des passagers différents.
- 4) Une réservation concerne un seul vol, et un seul passager.
- 5) Une réservation peut être annulée ou confirmée.
- 6) Un vol a un aéroport de départ et un aéroport d'arrivée.
- 7) Un vol a un jour et une heure de départ et un jour et une heure d'arrivée.**
- 8) Un vol peut comporter des escales dans des aéroports
- 9) Une escale a une heure d'arrivée et une heure de départ.
- 10) Chaque aéroport dessert une ou plusieurs villes

Nous allons entreprendre progressivement la réalisation d'un Diagramme de classe

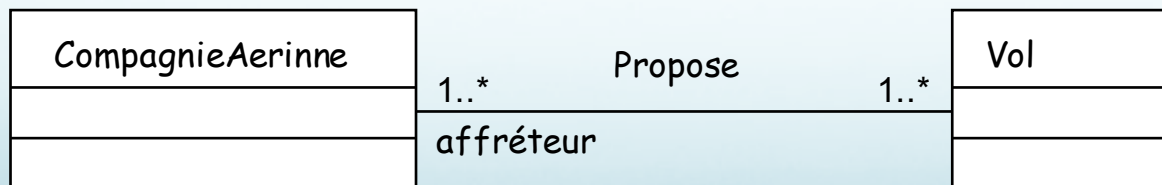
➤ Modélisation de la phrase :

1) Des compagnies aériennes proposent différents vols.

CompagnieAerienne et *Vols* sont 2 objets métiers : 2 classes

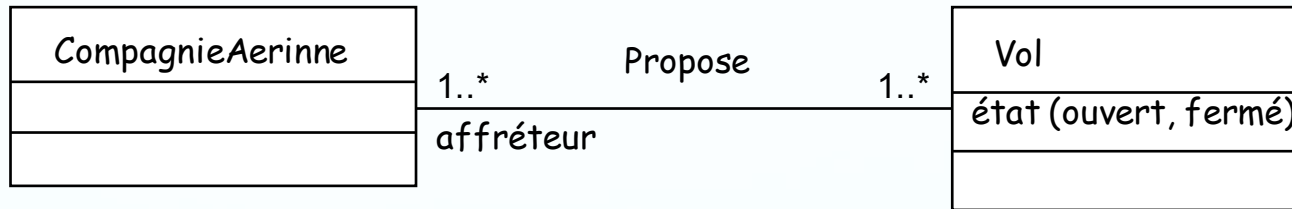


- Un vol est réalisé par une seule compagnie mais partagé par plusieurs affréteurs

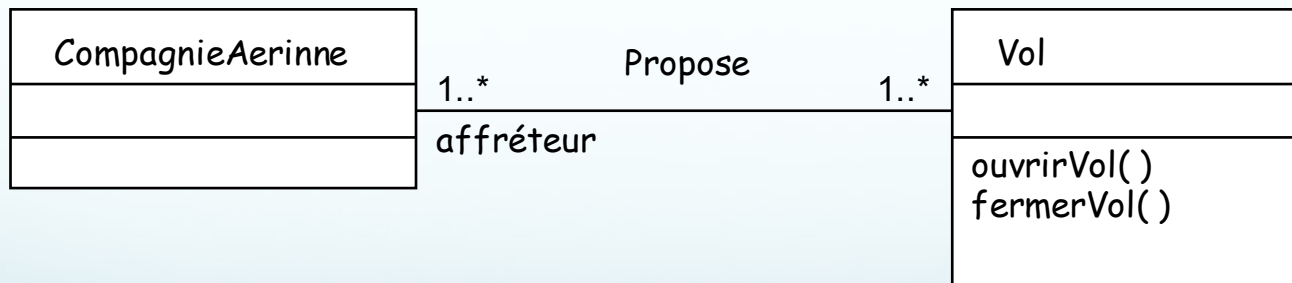


➤ Modélisation de la phrase :

2) Un vol est ouvert à la réservation et fermé sur ordre de la compagnie.



- Tout objet peut avoir un état (diagramme d'états).
- Dans un diagramme de classes tout concept dynamique est modélisé en opération.
- Il faut représenter la 2° phrase par 2 opérations : *ouvrirVol()* et *fermerVol()*
- Dans quelle classe ? Responsabilité d'une classe

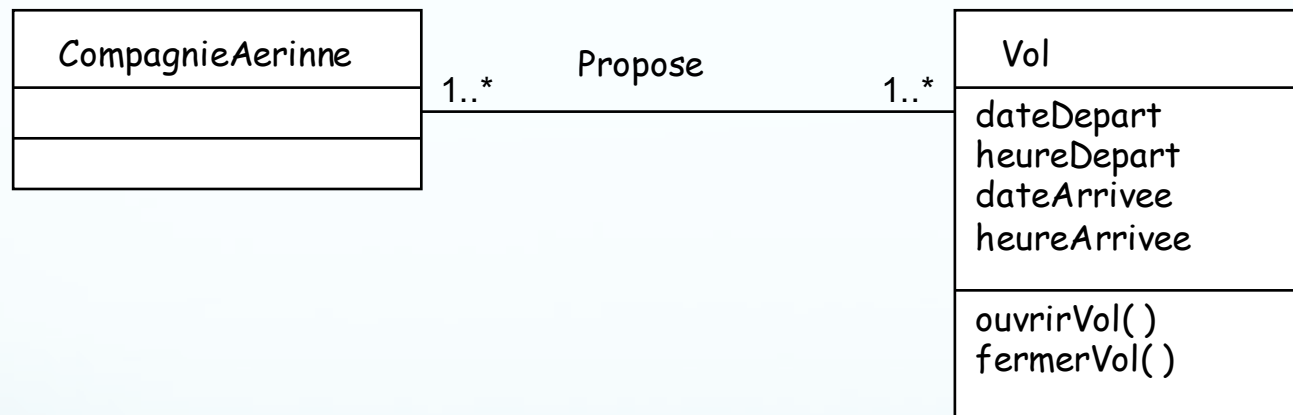


- Les opérations sont déclarées dans l'objet dans lequel elles doivent s'exécuter
- Les autres pourront déclencher ces opérations par envoi de messages
- La classe CompagnieAerienne a une association avec la classe vol.

➤ Modélisation des phrases :

7° Un vol a un jour et une heure de départ et un jour et une heure d'arrivée.

➤ Les dates et les heures de départ et d'arrivée ne représentent que des valeurs : attributs.



➤ Pour savoir si un élément doit être représenté en attribut ou en objet :

➤ S'il n'y a que sa valeur qui est intéressante : c'est plutôt un attribut.

➤ Si plusieurs questions peuvent concerner l'élément, alors il faut le représenter en objet.

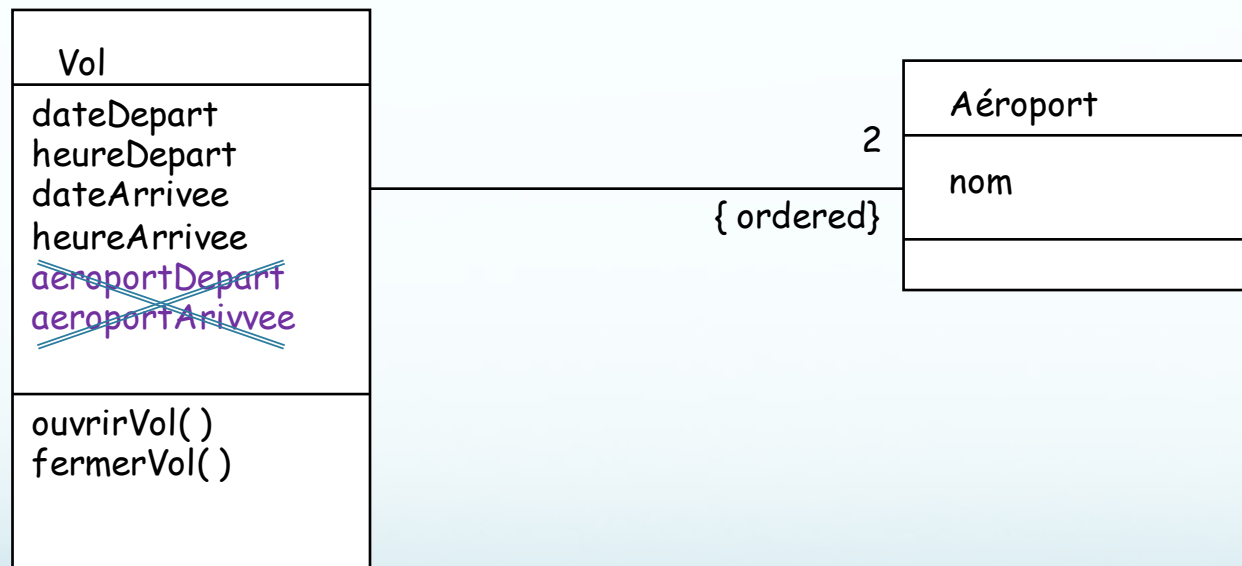
➤ Modélisation des phrases :

6) Un vol a un aéroport de départ et un aéroport d'arrivée.

➤ Par quoi peut-on représenter l'élément "Aéroport" ?

3 réponses sont envisageables :

1. Soit avec une classe et une association de multiplicité 2

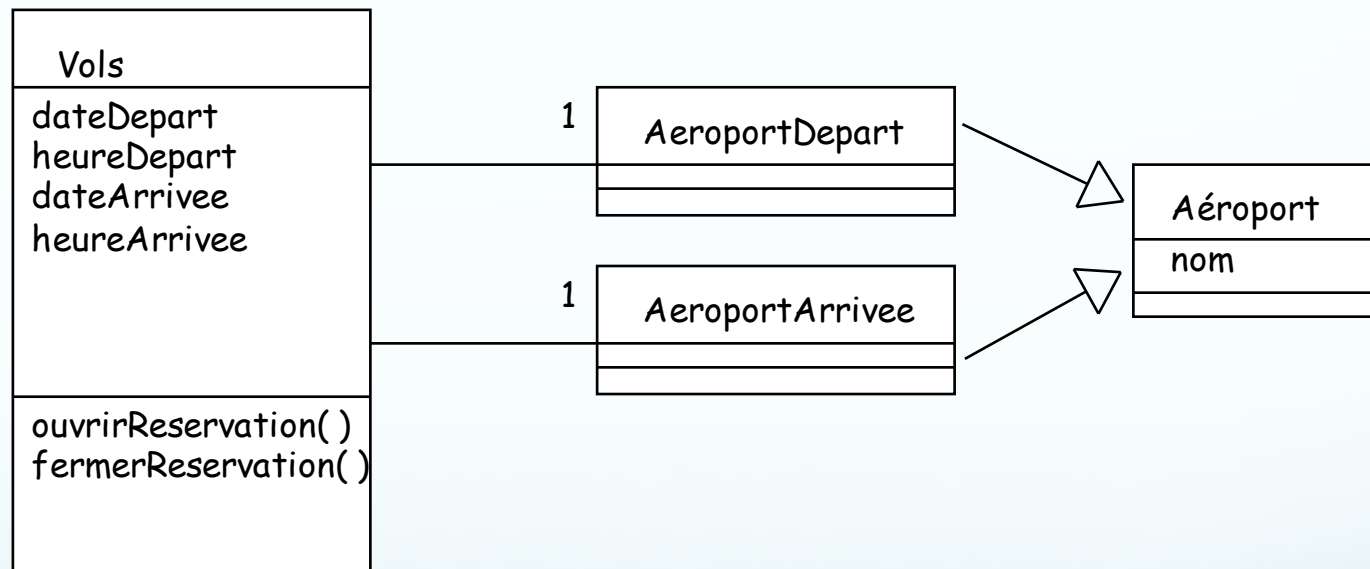


👉 Modélisation peu parlante.

➤ Modélisation des phrases :

6) Un vol a un aéroport de départ et un aéroport d'arrivée.

2. Soit avec 2 classes

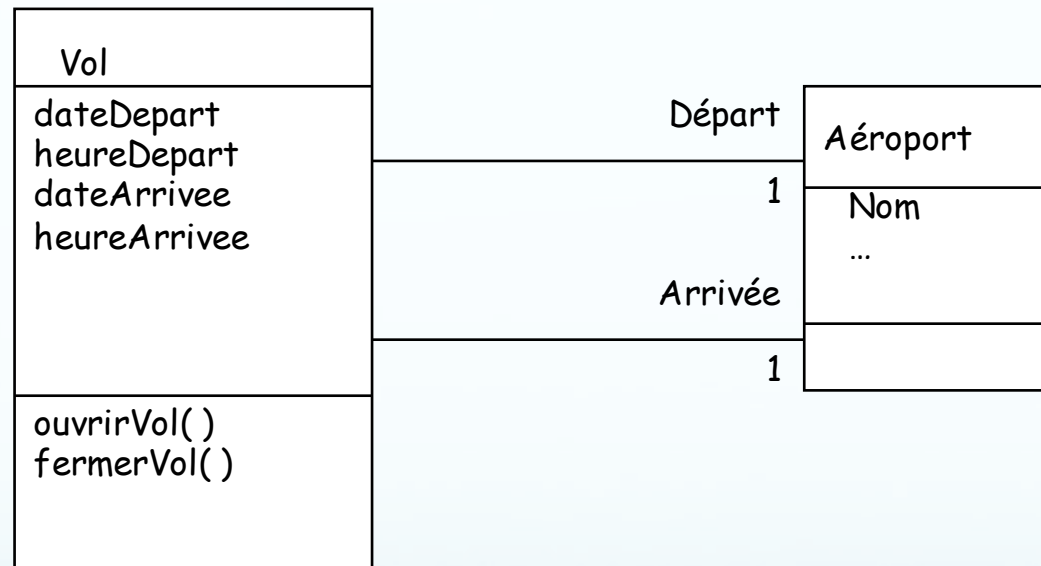


👉 Modélisation incorrecte. Tout aéroport peut être de départ et d'arrivée.

➤ Modélisation des phrases :

6) Un vol a un aéroport de départ et un aéroport d'arrivée.

3. Soit avec 2 associations

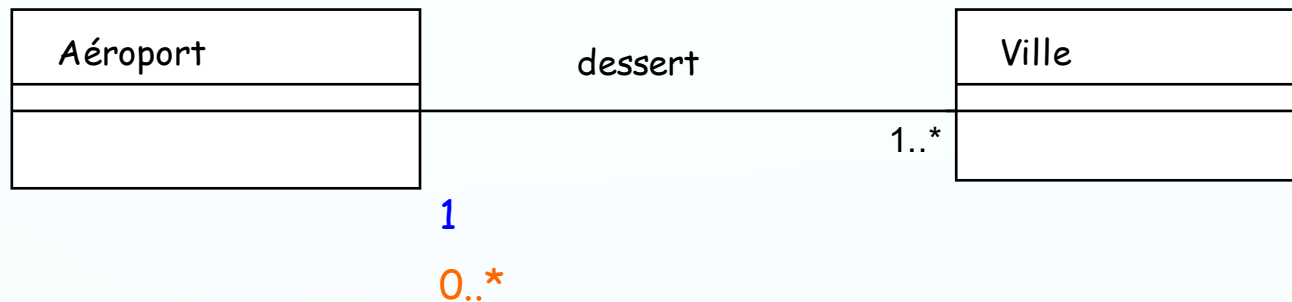


👉 Le rôle de chaque association précise son sens.

➤ Modélisation des phrases :

10) Chaque aéroport dessert une ou plusieurs villes

➤ On ne peut pas savoir la multiplicité de "Aéroport"



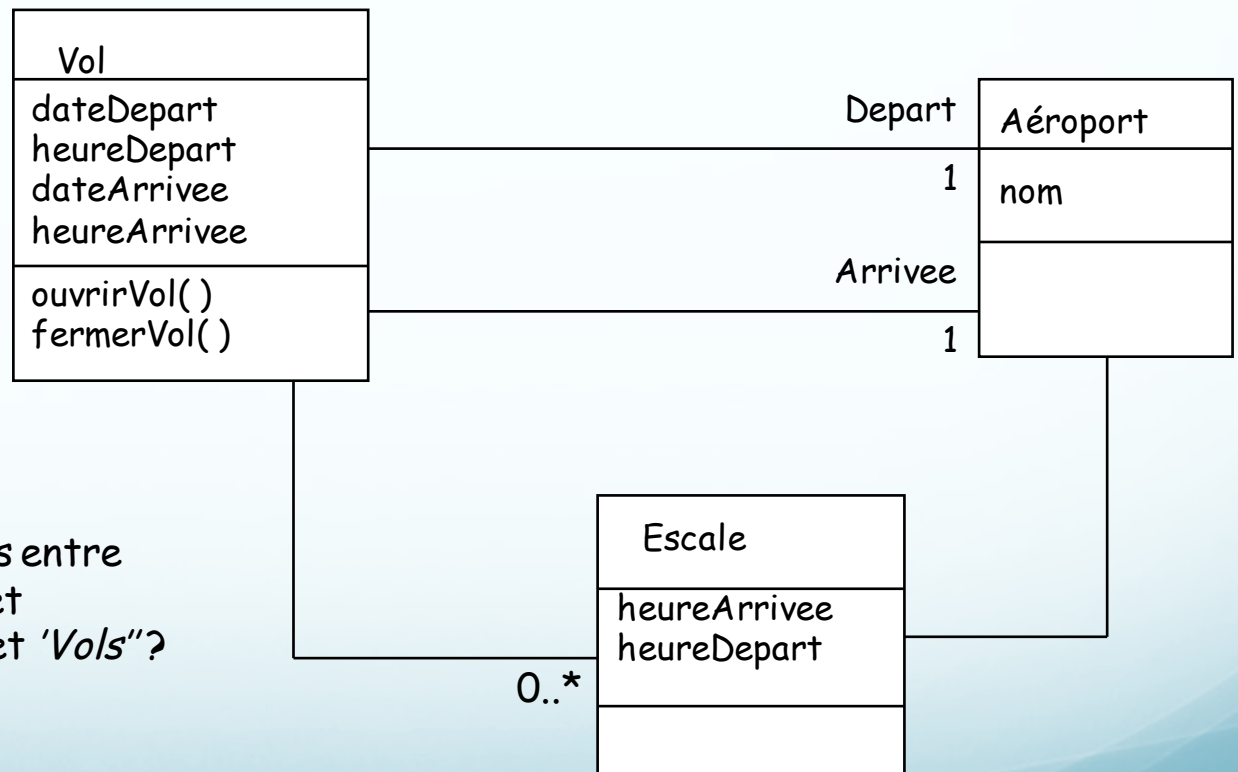
- Si on considère que desservir une ville signifie l'aéroport le plus proche, il n'y en a qu'un :
la multiplicité est de 1
- Si on considère que desservir une ville signifie les aéroports dans un rayon de x km :
la multiplicité est de 0..*

➤ Modélisation des phrases :

8) Un vol peut comporter des escales dans des aéroports

9) Une escale a une heure d'arrivée et une heure de départ.

➤ Une escale a les propriétés heure d'arrivée et heure de départ, c'est donc un objet.



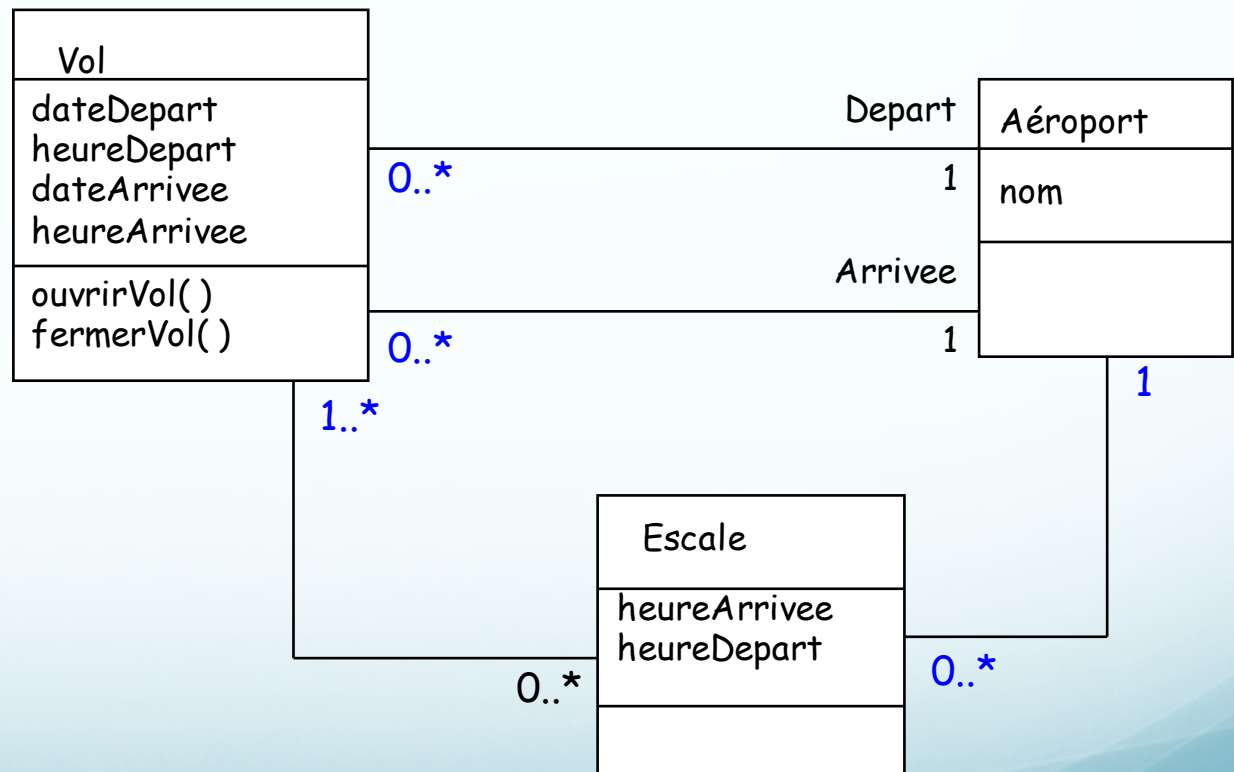
➤ Quelles sont alors les multiplicités entre "Vols" et "Escale", entre "Escale" et "Aéroport" et entre "Aéroport" et "Vols"?

➤ Modélisation des phrases :

8) Un vol peut comporter des escales dans des aéroports

9) Une escale a une heure d'arrivée et une heure de départ.

➤ Une escale a les propriétés heure d'arrivée et heure de départ, c'est donc un objet.



➤ Modélisation des phrases :

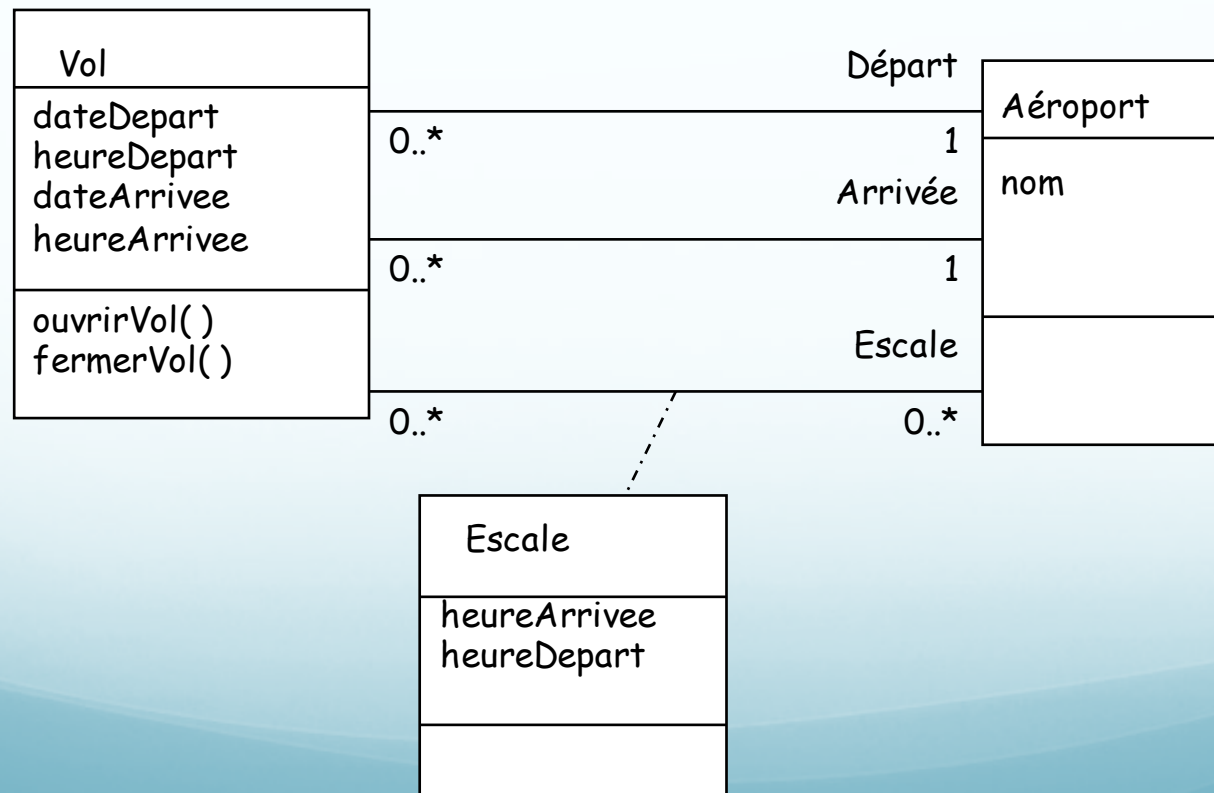
8) Un vol peut comporter des escales dans des aéroports

9) Une escale a une heure d'arrivée et une heure de départ.

➤ "Escale" a peu d'informations propres. Elle n'est qu'une partie de "Vol".

➤ On peut la représenter comme une spécialisation de "Aéroport". Mais elle n'est pas totalement un aéroport.

➤ La meilleure solution serait de la modéliser comme une classe d'association entre et "Vols" et "Aéroport".



➤ Modélisation des phrases :

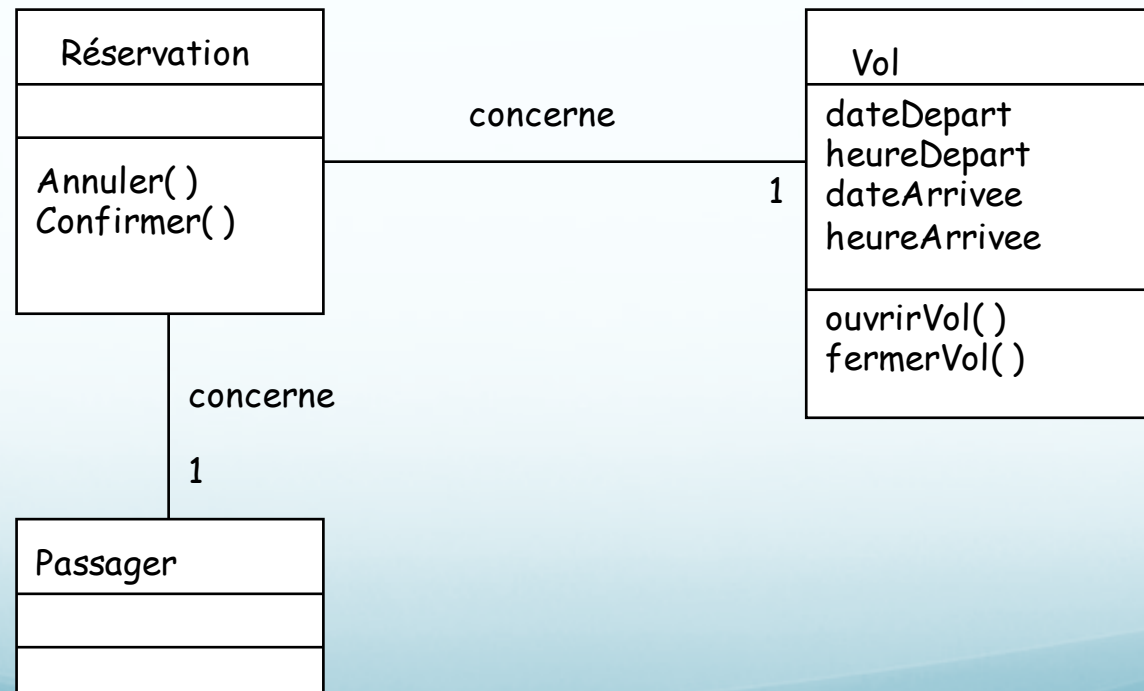
4) Une réservation concerne un seul vol, et un seul passager.

5) Une réservation peut être annulée ou confirmée.

➤ La réservation et le passager sont 2 concepts métier : 2 classes d'objets

➤ Une réservation concerne un seul vol et un seul passager: donc 2 associations entre "Vol" et "Réservation" et entre "Réservation" et "Passager".

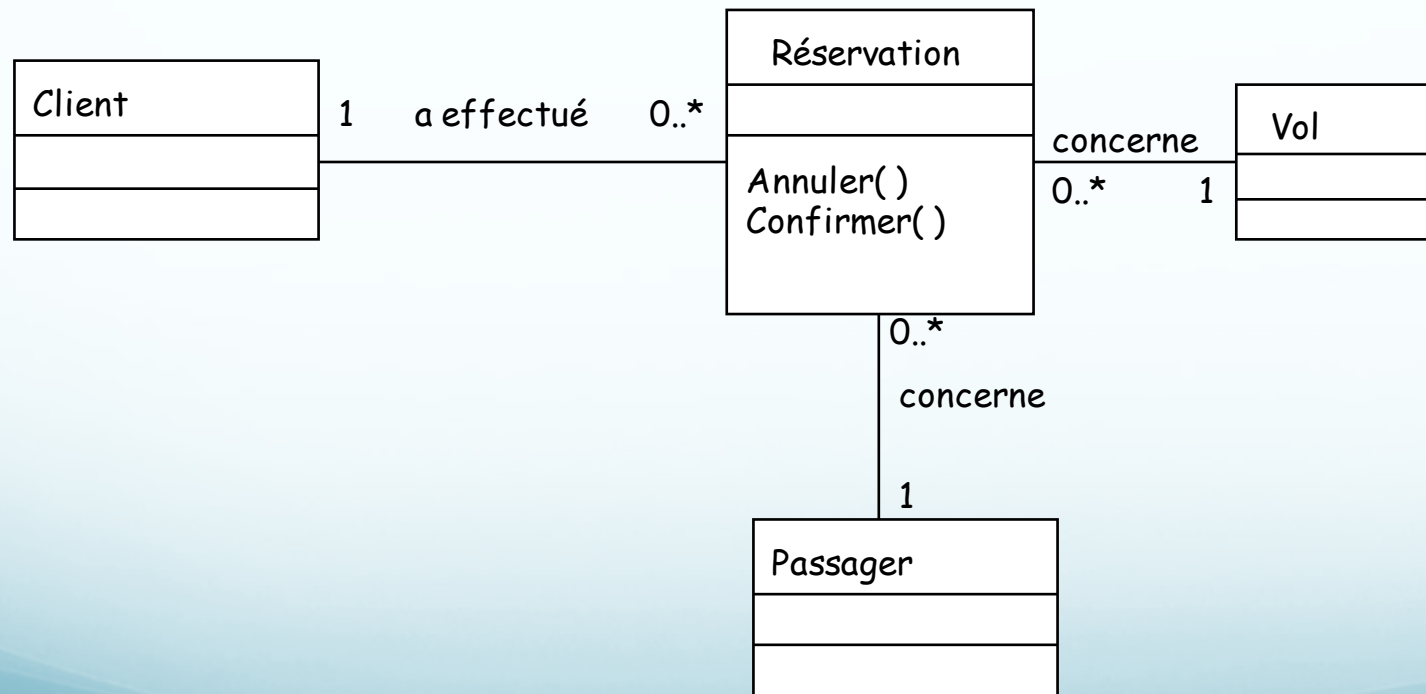
➤ La 5° phrase se traduit par l'ajout de 2 opérations *annuler()* et *confirmer()* dans "Reservation".



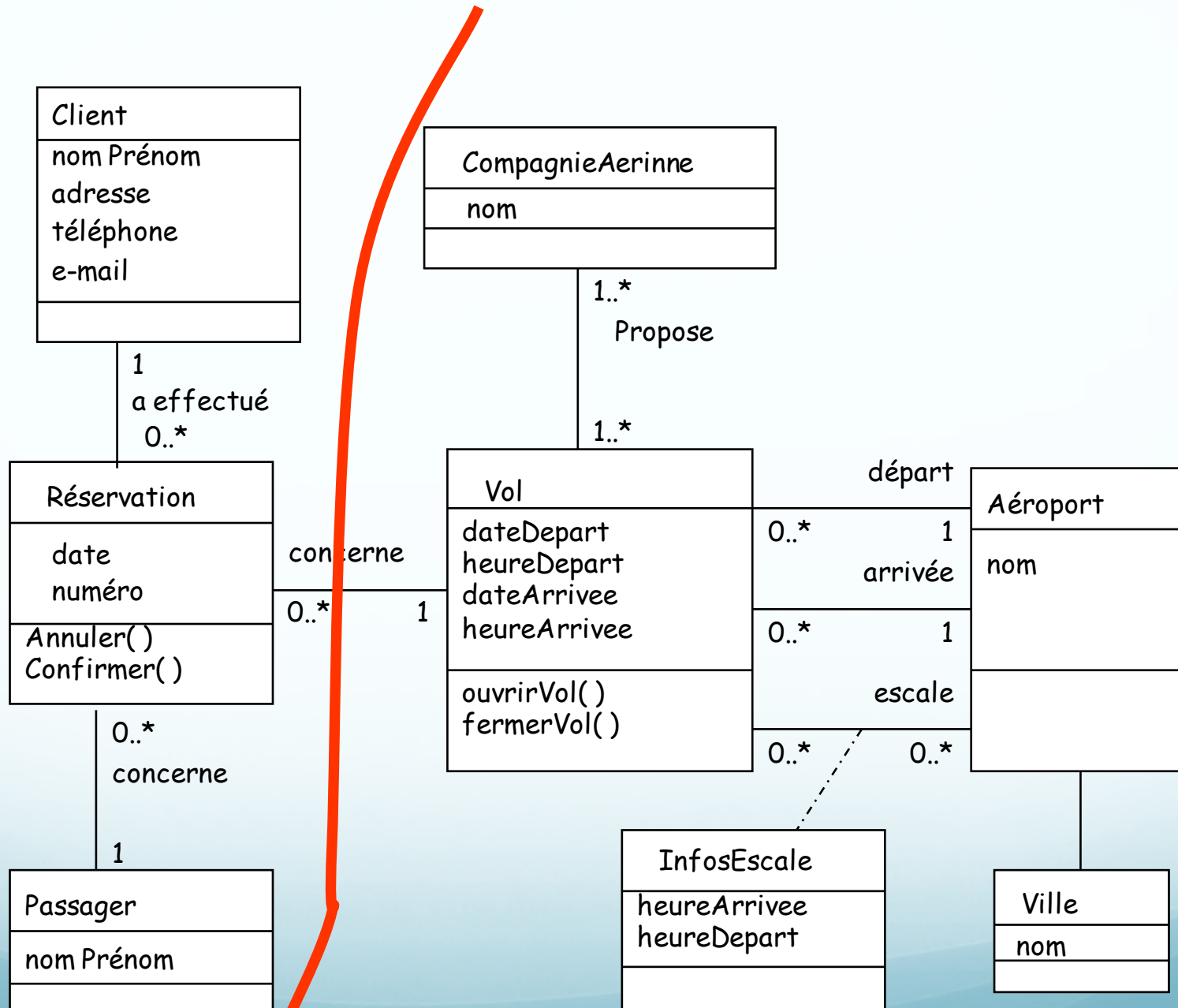
➤ Modélisation des phrases :

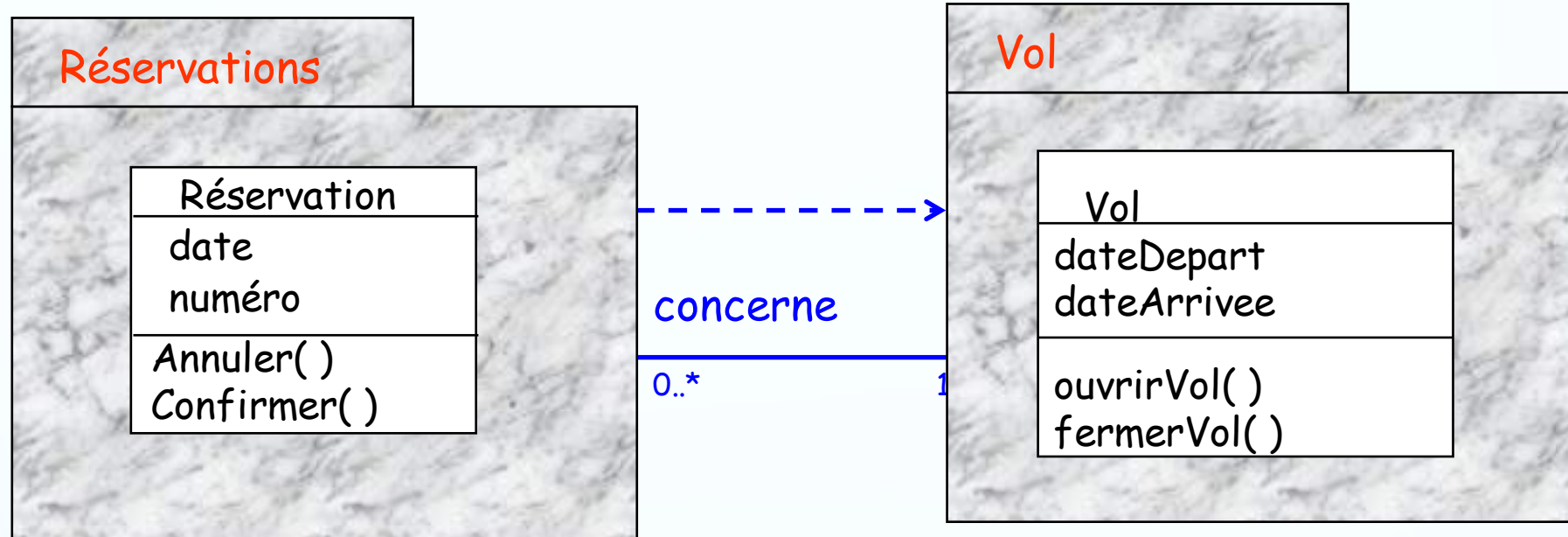
3) Un client peut réserver un ou plusieurs vols, pour des passagers différents.

➤ Il faut discerner un client d'un passager



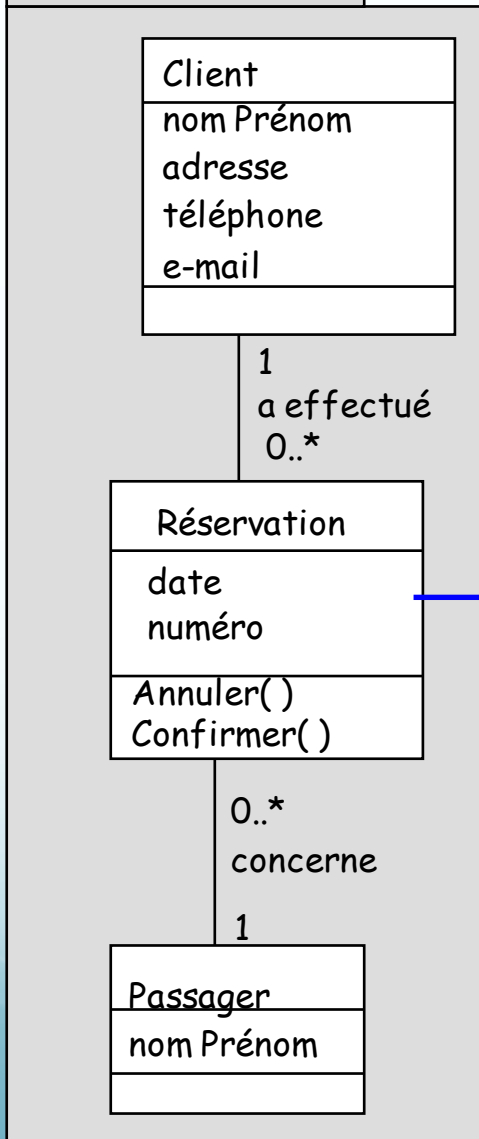
➤ Le diagramme des classes peut être réorganisé en packages



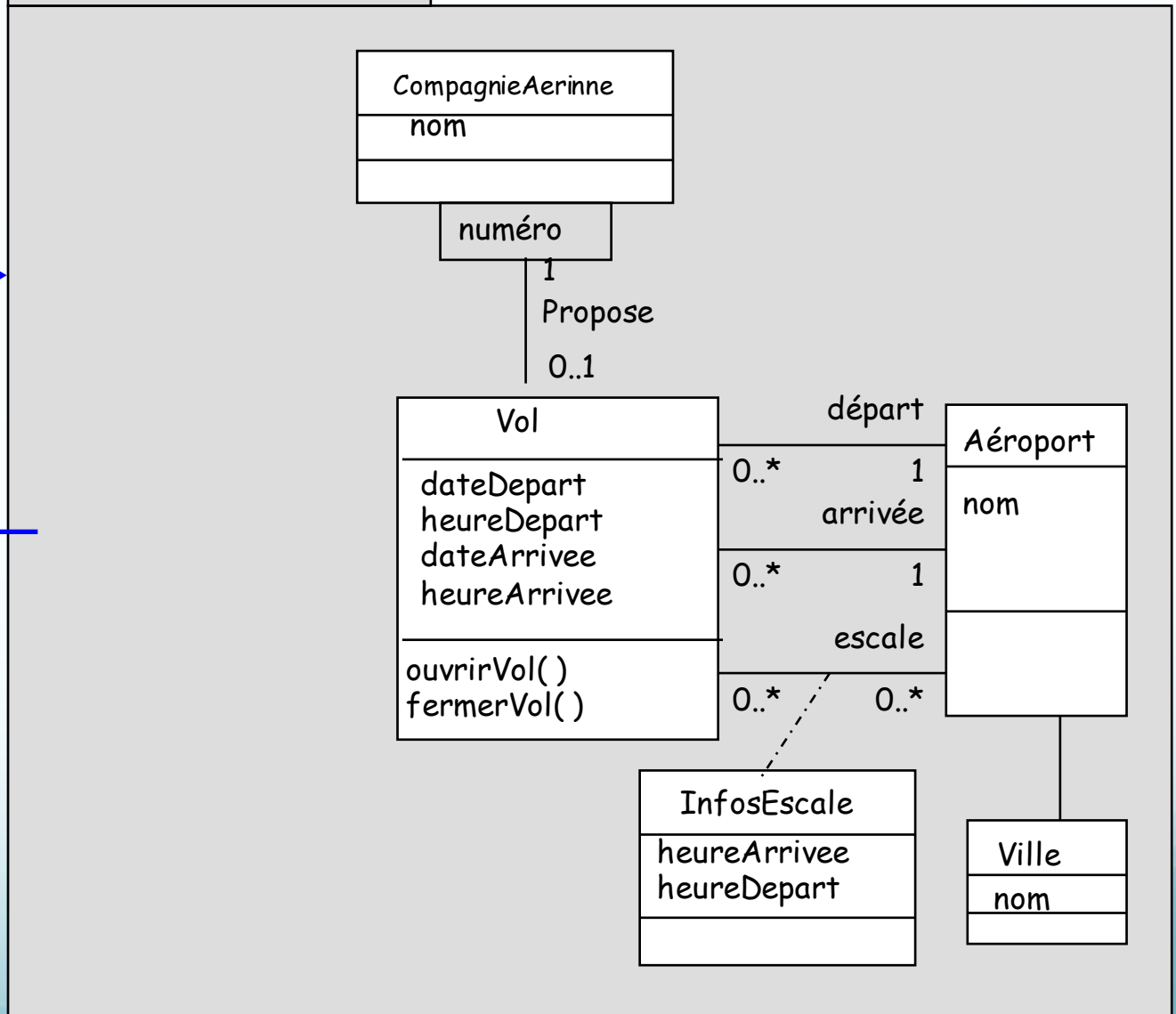


- Réduire la dépendance mutuelle afin d'augmenter la modularité et l'évolutivité d'une application

Réservations

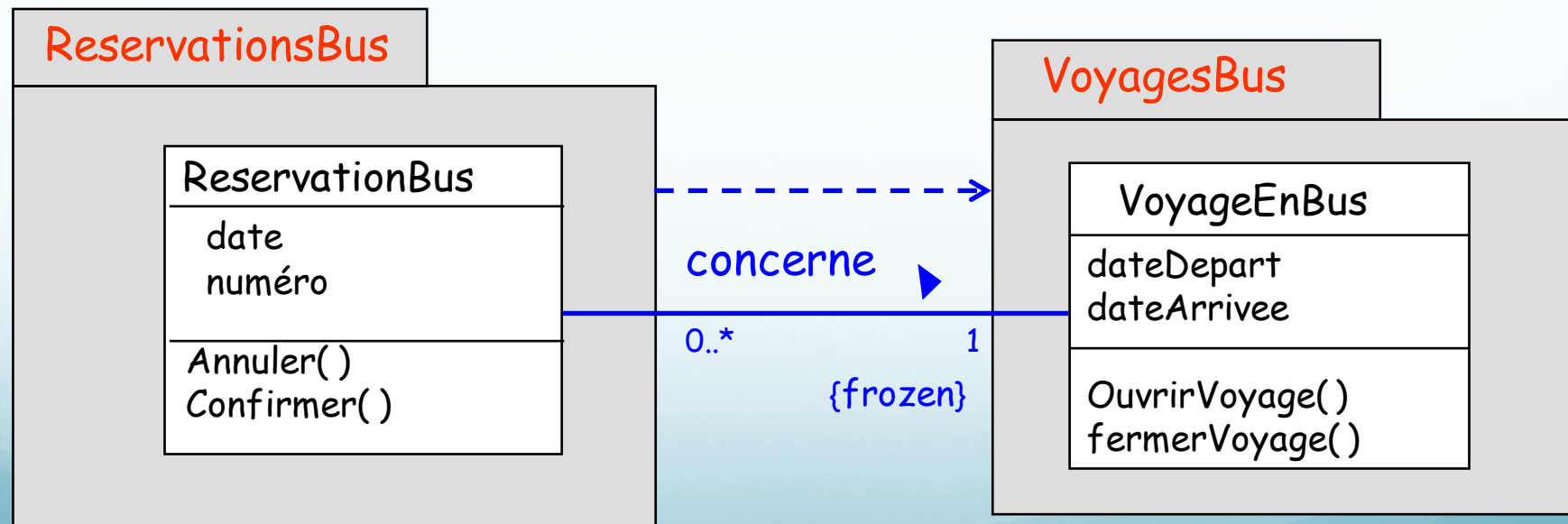


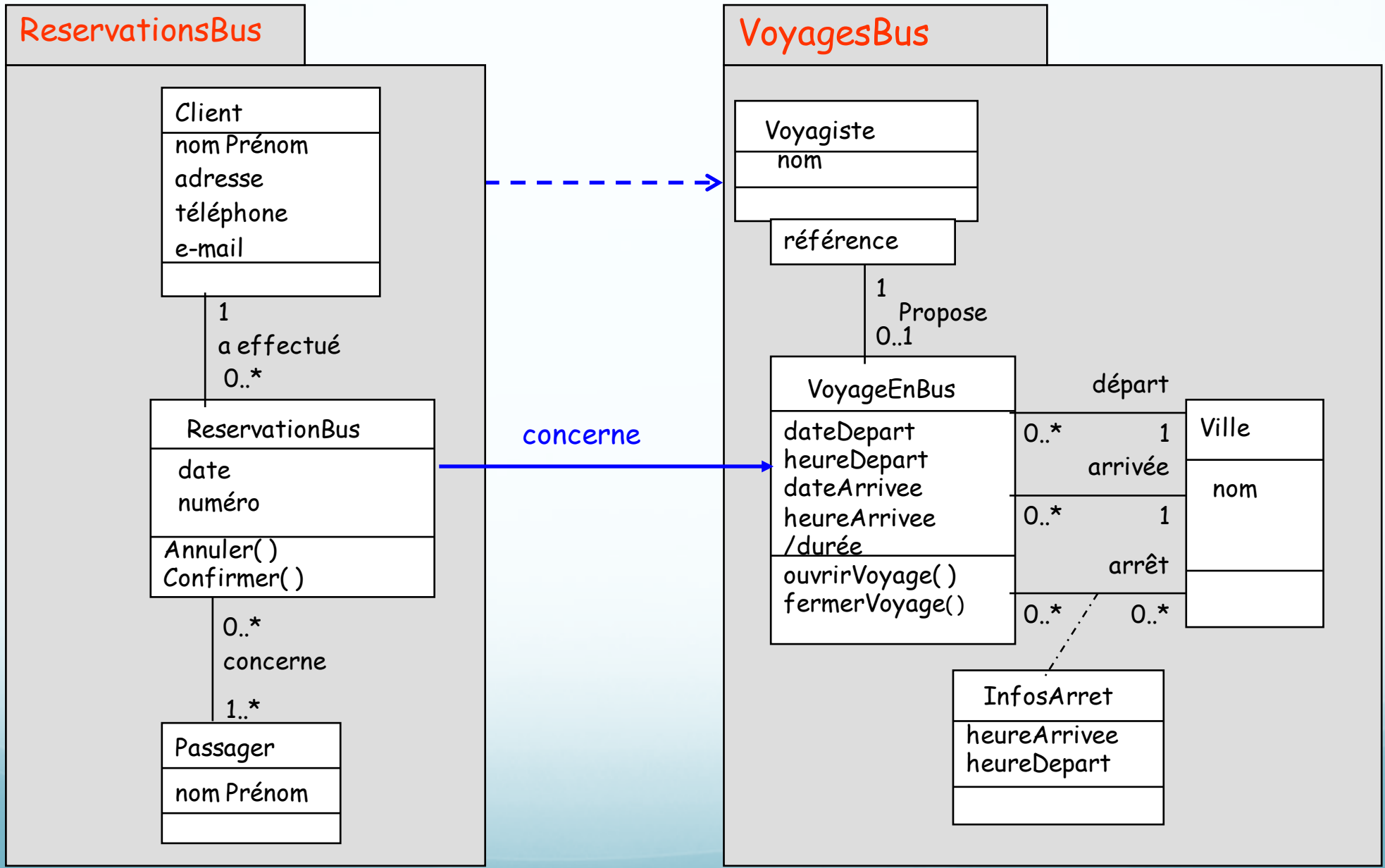
Vol



Généralisation et réutilisation

- On veut élargir ce domaine aux voyages par bus que des transporteurs assurent.
- Un voyage en bus à une ville de départ et un ville d'arrivée avec des dates et des heures associées.
- Un trajet peut comporter des arrêts dans des villes intermédiaires.
- Un client peut réserver un ou plusieurs voyages pour un ou plusieurs passagers

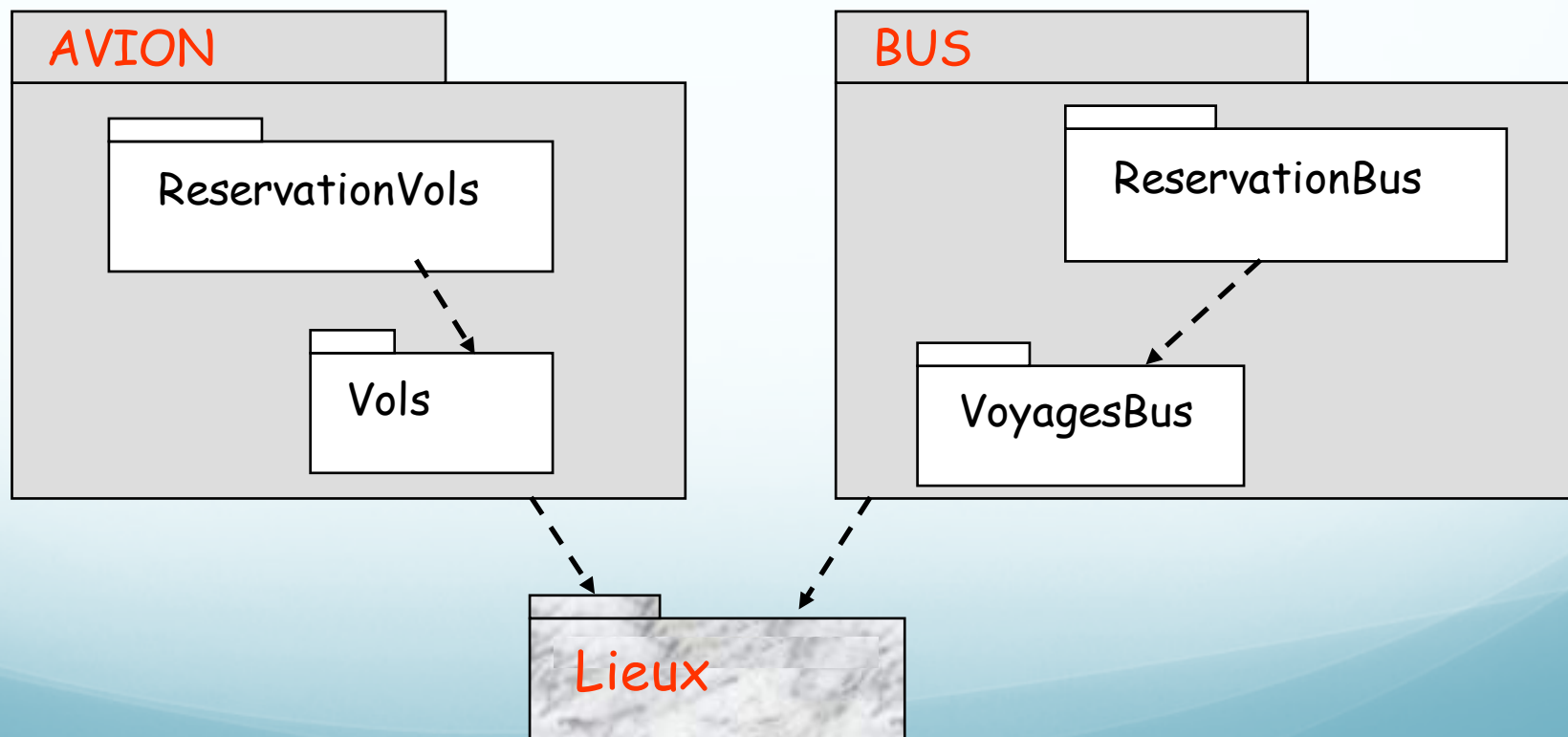




Fusion des 2 modèles

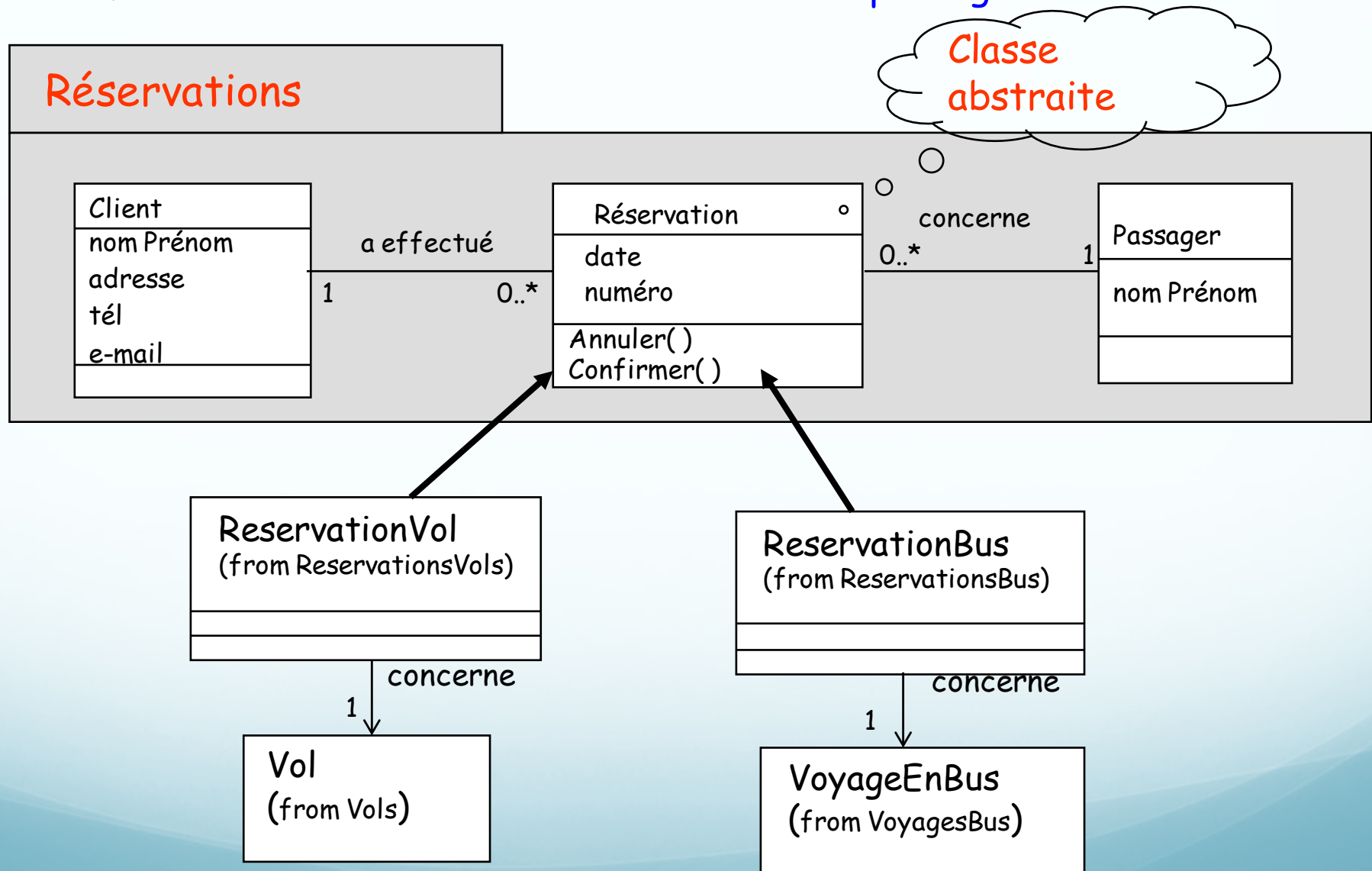
1. Il faut isoler les classes communes dans des packages
2. Il faut factoriser les propriétés communes

1



2

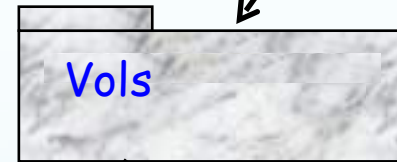
Il faut isoler les classes communes dans des packages



Package généralisé



Packages spécialisés



Package réutilisable

