Solutions des exercices de révision pour examen intra

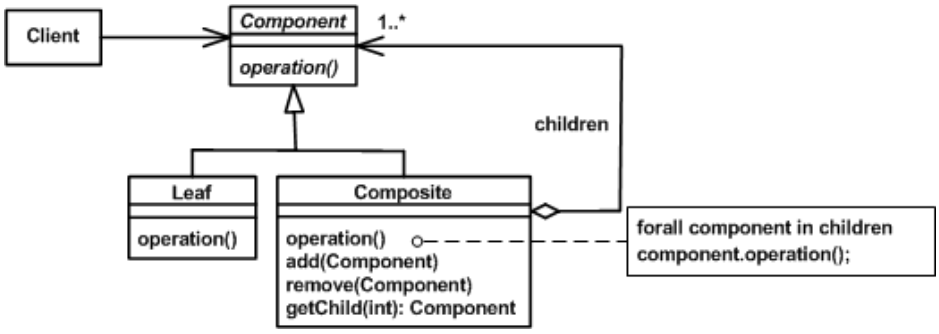
Question 1

A) Encapsulation : Cacher les détails d'implémentation d'une classe aux classes utilisant l'objet

B) Le patron Itérateur renforce l'encapsulation en fournissant un itérateur qui cache la structure de données utilisée pour implémenter une collection.

Question 2

A) Considérons le patron Composite dont la structure générique est la suivante



Polymorphisme : la classe Client appèle la méthode « operation » définie par la superclasse Component. La méthode exécutée sera celle de l'objet (Leaf ou Composite) affecté à l'attribut du type Component que le client a (montré avec l'association entre Client et Component).

B) Le polymorphisme aide à réduire le couplage statique : la classe Client connaît uniquement la superclasse Component. Ce n'est qu'à l'exécution que la classe Client reçoit une référence vers un Leaf ou un Composite.

Question 3

A) Patron à appliquer : Observater.

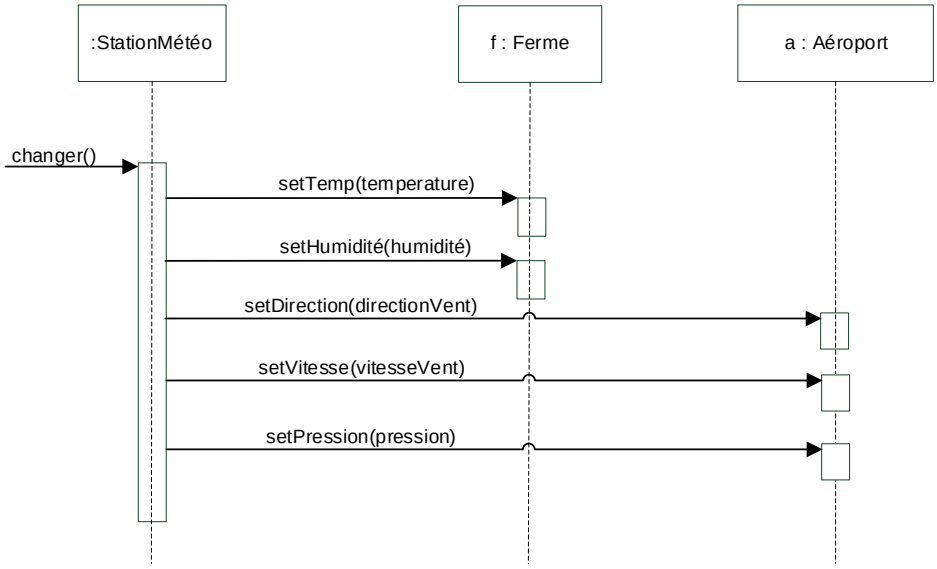
Avantage dans ce contexte : Le patron Observateur permet de découpler l'application des courtiers. L'application n'a pas besoin de connaître les différents processus que les courtiers déclenchent. Elle doit juste les notifier des variations des côtes:

B) Patron à appliquer : Méthode template.

Avantage dans ce contexte : Le patron nous permet d'imposer les mêmes étapes à tous les types de commande et d'éviter la duplication du code commun aux commandes.

Question 4

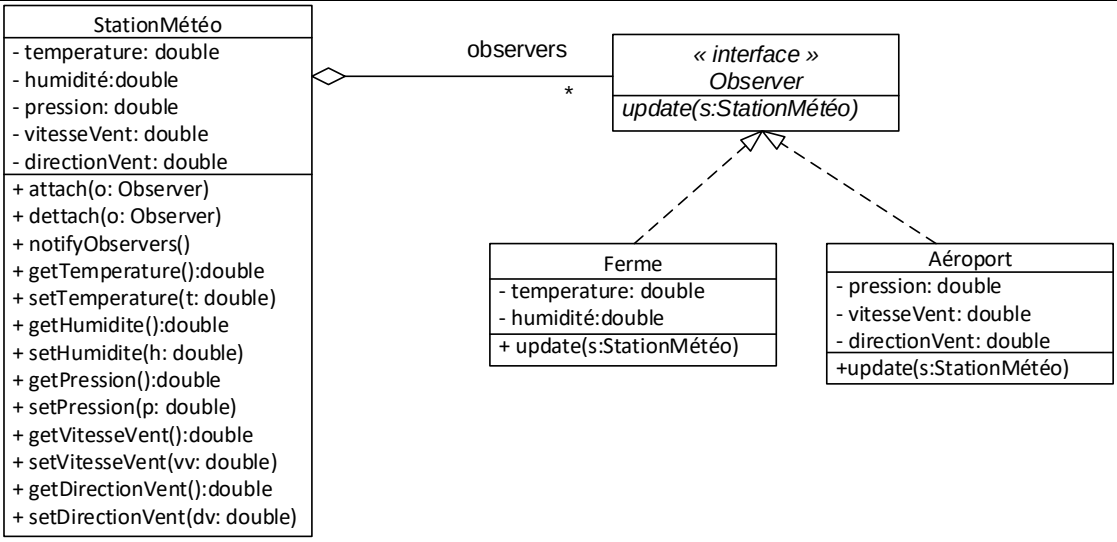
1) En se basant sur le diagramme de classes fournie dans l'énoncé, voilà le diagramme de séquences de la méthode changer() :



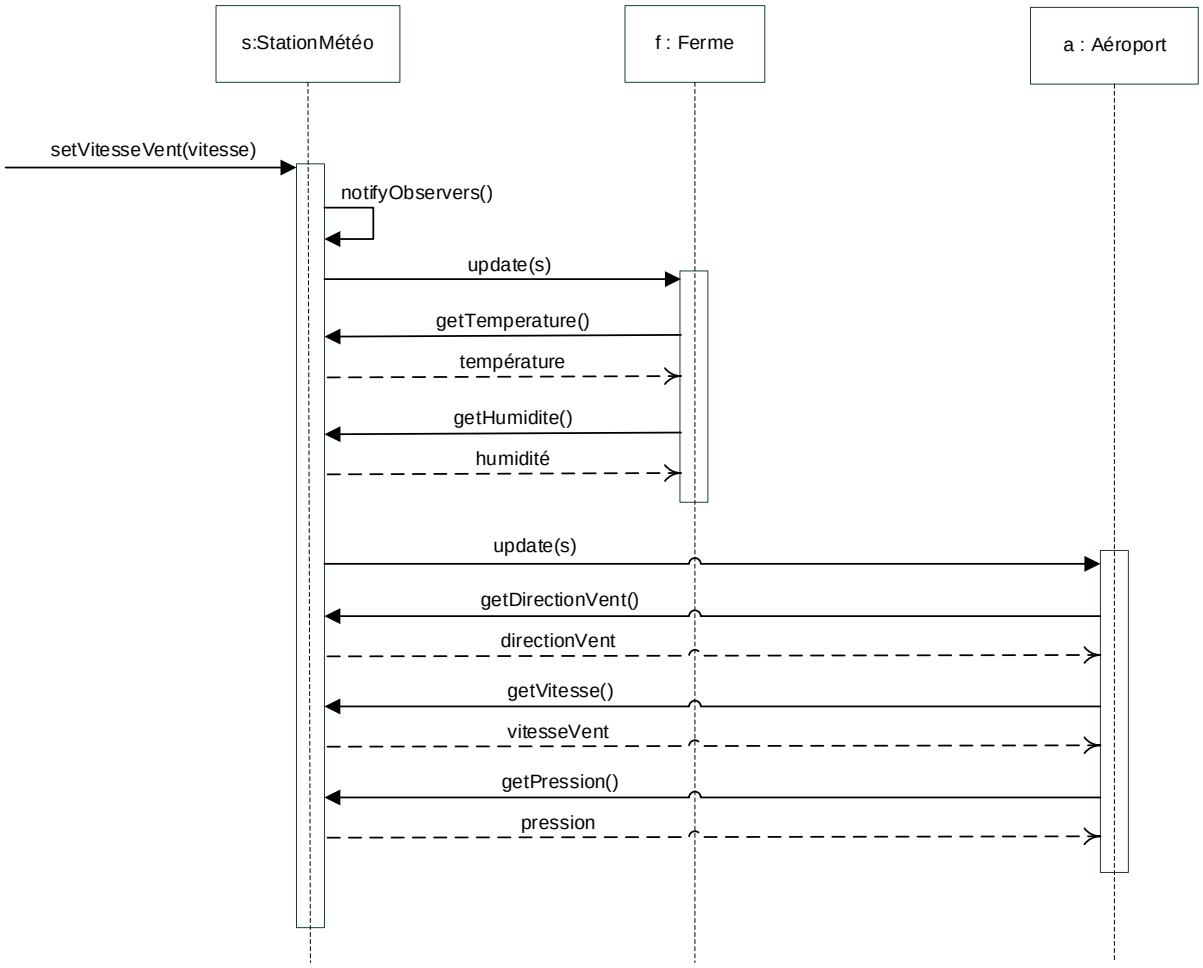
2) Problèmes de cette conception : la classe StationMétéo connaît toutes les classes qui ont besoin de ses informations et elle connaît les informations dont chacune de ces classes a besoin. Cela crée beaucoup de couplage. Aussi, la classe StationMétéo doit être modifiée à chaque fois que les besoins en information d'une de ces classes change.

3) Patron à appliquer pour résoudre le problème : Observateur

   Diagramme de classes avec le patron Observateur :



4) Diagramme de séquence de la méthode setVitesseVent(vitesse : double)



Question 5

Diagramme de séquence de l'appel à la méthode chercherContour() de la classe DetectionContour :