Question 1 (10 points) Faites 2 diagrammes de classes du logiciel, un pour chaque patron. Indiquez les attributs, les méthodes et les relations entre les classes. Vous pouvez utiliser un autre logiciel qu'Enterprise Architect pour les faire. Vous devez inclure vos 2 diagrammes dans votre fichier tp4.pdf. (Voir la section 5- Remise)

Voir fin du fichier

Question 2 (10 points) Identifiez les avantages et les désavantages des patrons composite et décorateur.

Patron composite :

Avantages :
D'abord, le patron composite permet de traiter les composants uniformément, que ça soit un objet composite de plusieurs autres objets, ou une feuille. De plus, le patron est très extensible puisqu'on peut facilement ajouter une sous-classe, comme une feuille, à notre arborescence, et celle-ci fonctionnera comme les autres composants de l'arborescence.

Désavantages :
En créant plusieurs objets composites, cela nécessite un coût élevé en mémoire.

Patron décorateur :

Avantages :
D'abord, la possibilité d'ajouter ou retirer des décorateurs librement rend le code beaucoup plus flexible que l'héritage statique. Ainsi, on peut avoir plusieurs types de décorateurs qui modifient chacun une classe décorée de façon différente. De plus, le patron décorateur permet de diviser une classe de base en plusieurs classes avec des responsabilités ciblées. Cela évite d'avoir une énorme classe de base comprenant toutes les fonctionnalités réunies.

Désavantages :
En premier, la division d'une classe de base en plusieurs décorateurs risque de nécessiter un coût élevé en mémoire, car on crée plusieurs objets plutôt qu'un seul. Aussi, l'identité de l'objet auquel on a ajouté un décorateur est modifiée. Donc, lorsqu'on tente d'accéder à l'objet original, cet accès est moins direct.

Question 3 (10 points) Supposons que l'on veuille, dans le cadre du patron composite, obtenir toutes les parcelles feuilles d'un arbre complexe de parcelles. Comment feriez-vous? Sans en faire l'implémentation, que devriez-vous ajouter dans les classes pour y parvenir?

CompositeParcelle a une méthode obtenirParcellesFeuilles, qui retourne toutes les parcelles feuilles de l'arborescence. Pour ce faire, on crée une liste parcellesFeuilles dans la méthode, puis on vérifie chaque AbsParcelle dans les m_parcellesEnfants. Si AbsParcelle est composite, alors on appelle récursivement la méthode obtenirParcellesFeuilles et ajoute la valeur de retour à la liste parcellesFeuilles. Si AbsParcelle n'est pas composite, alors elle est ajoutée à parcellesFeuilles. Finalement, la méthode retourne la liste parcellesFeuilles.

Question 4 (10 points) Au lieu d'utiliser le patron décorateur, un collègue vous propose de simplement faire plusieurs classes pour les différents types de parcelles, qui héritent directement d'une classe abstraite. On aurait alors par exemple ParcelleJardin, ParcellePelouse et ParcellePotager, qui hériteraient directement de AbsParcelleTerrain. Que permet le patron décorateur, qui n'est pas possible avec l'idée de votre collègue?

Le patron décorateur permet d'ajouter des responsabilités multiples plus facilement que l'héritage. Par exemple, si on veut que notre parcelle ait les responsabilités de ParcelleJardin, ParcellePelouse et ParcellePotager, il faudrait créer une classe ParcelleJardinPelouseEtPotager, qui hérite de AbsParcelleTerrain. Cela est long et fastidieux, car il faudrait faire une classe pour chaque combinaison. Par contre, avec le patron décorateur, on peut simplement décorer la parcelle par JardinDecorateur, puis par PotagerDecorateur et finalement par PelouseDecorateur pour lui assigner les responsabilités des trois décorateurs (comme c'est fait dans le fichier testDecorateur.cpp).

Starter Class Diagram

Package in package 'Model'

Starter Class Diagram
Version 1.0 Phase 1.0 Proposed
wiyoua created on 2020-11-04. Last modified 2018-02-07

Patron composite diagram

Class diagram in package 'Starter Class Diagram'

Patron composite
Version 1.0
wiyoua created on 2020-11-04. Last modified 2020-11-04

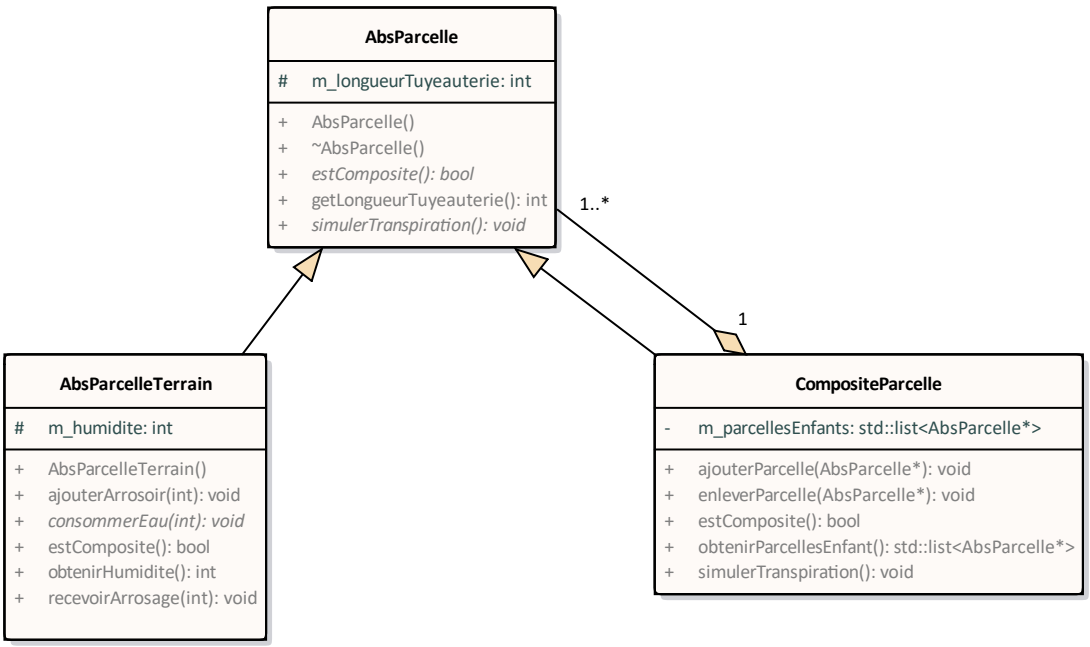

Figure 1: Patron composite

Starter Class Diagram

Package in package 'Model'

Starter Class Diagram
Version 1.0 Phase 1.0 Proposed
wiyoua created on 2020-11-04. Last modified 2018-02-07

Patron décorateur diagram

Class diagram in package 'Starter Class Diagram'

Patron décorateur
Version 1.0
wiyoua created on 2020-11-04. Last modified 2020-11-04

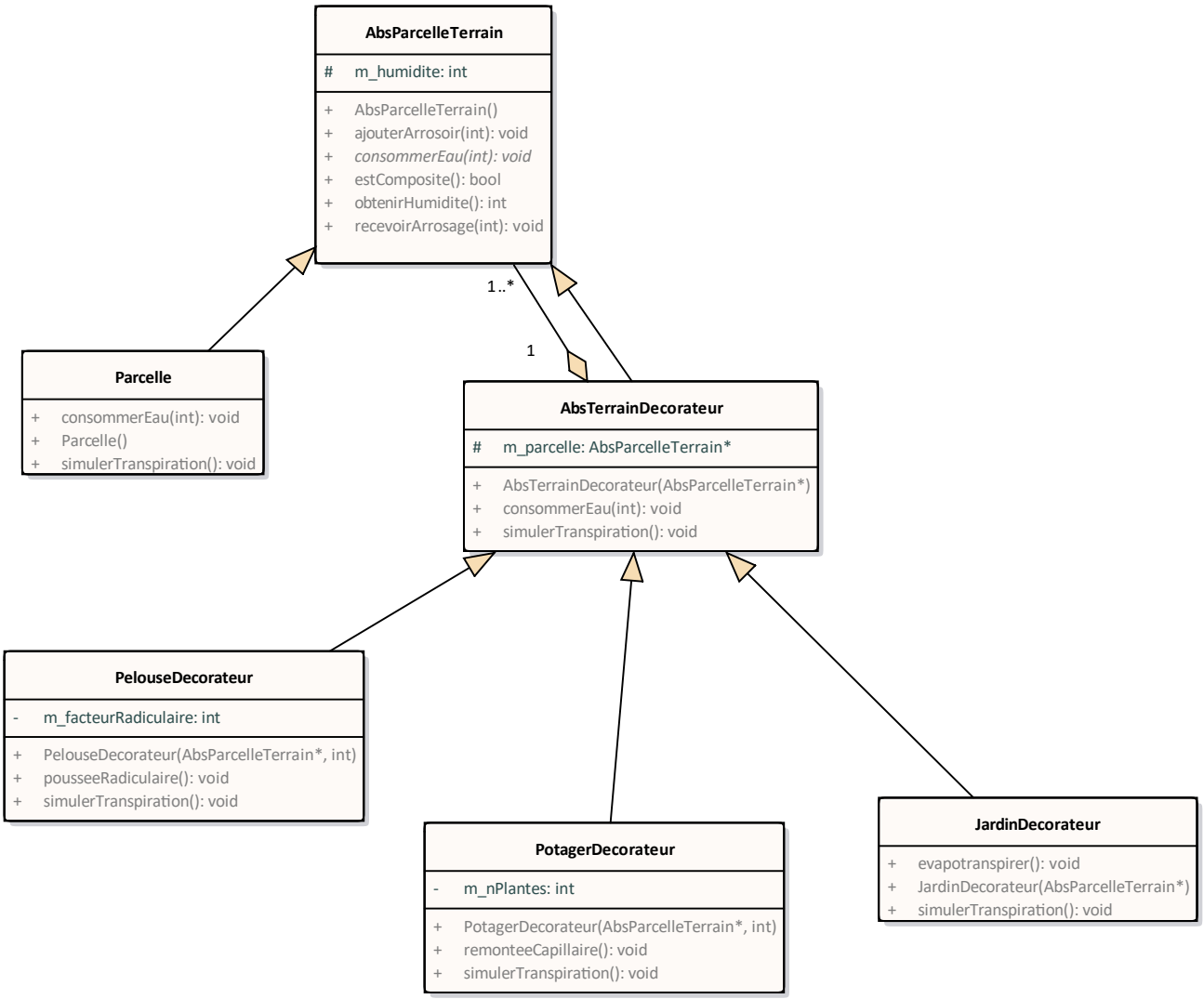

Figure 1: Patron décorateur