Réponses aux questions

# 1) a)

Composer des objets dans des structures d’arbre pour représenter des hiérarchies composants/composés et permettre au client de manipuler uniformément les objets simples et les objets au sein de leurs compositions.

## b)

# 2)

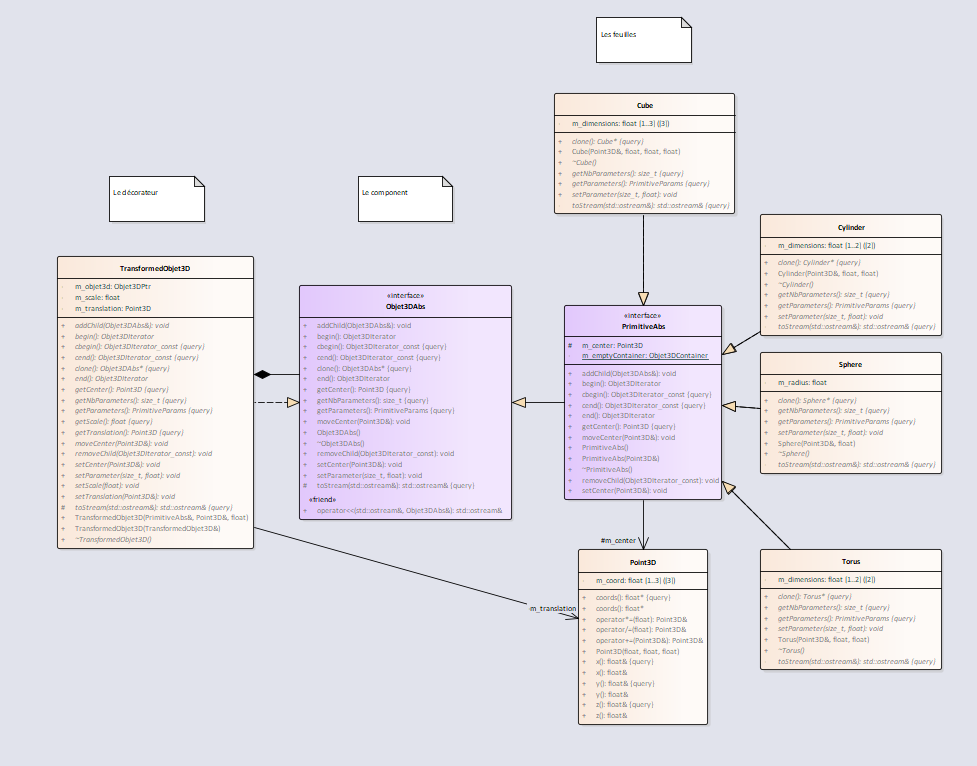
Objet3Dabs est la première abstraction, c’est une interface qui définit les fonctions qui seront réimplémentées dans les objets. C’est à partir de cette interface que les différents types de primitives sont utilisées, c’est donc l’interface uniforme. PrimitiveAbs est la deuxième abstraction qui permet de manipuler les différentes feuilles (les types de primitives), c’est un point commun pour chaque type de primitive qui eux redéfinissent seulement les fonctions qui ont une implémentation unique.

# Patron décorateur

## 1)a)

Donner des responsabilités additionnelles à un objet dynamiquement. C’est une alternative à la dérivation de classe qui permet d’étendre les fonctionnalités d’une classe.

## b)



# 2)

Les classes primitives ont les mêmes responsabilités qu’elles avaient auparavant en plus d’avoir les fonctions rajoutées par le décorateur. Dans notre cas, les fonctionnalités ont été étendues de façon à pouvoir translater et mettre à l’échelle les primitives. On a également modifié l’implémentation du getCenter et getParameters.

# 3)

Dans notre cas nous voulons modifier des primitives de base une à la fois, le décorateur étant lié à PrimitiveAbs permet cela, mais nous pourrions également l’appliquer sur tous les objets 3D, il faudrait modifier l’implémentation pour pouvoir modifier chaque primitive d’un objet 3D.

# Conteneur

## 1)a)

Fournir une méthode d’accès séquentielle aux éléments d’un conteneur d’objets sans exposer sa structure interne.

## b)

Nous utilisons le conteneur vector et des itérateurs sur des objets3D qui ont été implémentés comme Objet3DIterator et Objet3DIterator\_const.

# 2)

Le m\_emptyContainer nous permet de retourner le début et la fin des deux conteneurs implémentés, car il contient les primitives de base. Il est déclaré privé par la classe PrimitiveAbs, car il n’a pas d’autre utilité particulière qui permettrait aux différentes primitives de l’utiliser à bon escient. Il est aussi déclaré statique, car il est utilisé de la même façon pour les différentes primitives, donc une copie suffit.

# 3)

# 4)

Grace à ces surcharges le référence vers une primitive à partir d’un itérateur est possible et sa modification par des fonctions aussi. L’utilisations des objets 3D est donc simplifiée. DÉSAVANTGE??