

LOG2410: Conception à base de patrons I

Travail pratique # 4

Hiver 2019

Équipier 1: Mariam Sarwat (1901777)

Équipier 2 : Yasmina Abou-Nakkoul (1897266)

Date de remise :

26 mars 2019

**2 - Patron Composite**

1) Identifiez les points suivants :

a) L’intention du patron Composite.

b) La structure des classes réelles qui participent au patron ainsi que leurs rôles. Faite un diagramme de classes avec Enterprise Architect pour l’instance du patron composite. Ajouter des notes en UML pour indiquer les rôles, et exportez le tout en pdf).

2) Identifiez toutes les abstractions présentent dans la conception du TP4, et pour chacune, identifiez les responsabilités spécifiques qui lui ont été assignées.

3) Dans l’implémentation actuelle du système PolyVersion, quel objet ou classe est responsable de la création de l’arbre des composantes.

**3 – Patron Proxy**

1) Identifiez les points suivants :

a) L’intention du patron Proxy.

b) La structure des classes réelles qui participent au patron ainsi que leurs rôles. Faite un diagramme de classes avec Enterprise Architect pour l’instance du patron proxy. Ajouter des notes en UML pour indiquer les rôles, et exportez le tout en pdf.

**4 – Conteneurs et Patron Iterator**

2) Identifiez les points suivants :

a) L’intention du patron Iterator.

b) La classe de conteneur de la STL utilisée pour stocker les enfants dans la classe Composite et les classes des Iterators utilisés dans la conception qui vous a été fournie.

3) Expliquez le rôle de l’attribut statique m\_empty\_transforms défini dans la classe AbsTransform. Expliquez pourquoi, selon vous, cet attribut est déclaré comme un attribut statique et privé.

4) Quelles seraient les conséquences sur l’ensemble du code si vous décidiez de changer la classe de conteneur utilisée pour stocker les enfants dans la classe Composite? On vous demande de faire ce changement et d’indiquer toutes les modifications qui doivent être faites à l’ensemble du code suite au changement. Reliez la liste des changements à effectuer à la notion d’encapsulation mise de l’avant par la programmation orientée-objet. À votre avis, la conception proposée dans le TP4 respecte-t-elle le principe d’encapsulation ?

5) Les classes dérivées TransformIterator et TransformIterator\_const surchargent les opérateur « \* » et « -> ». Cette décision de conception a des avantages et des inconvénients. Identifiez un avantage et un inconvénient de cette décision.