2-Patron Composite :

1. L’intention du patron composite est essentiellement de manipuler des objets similaires (repeatTransform, invertTransform) par l’intermédiaire d’une interface(AbsTransform), qui ont des méthodes communes et aux implémentations adaptées à chaque objet.

b)



Feuilles : invertTransform, repeatTransform

Responsabilités :

invertTransform : inverser le segment audio

repeatTransform : dupliquer le segment audio un nombre de fois donné.

Composite : compositeTransform

Responsabilitees :

Inverser le segment audio

Dupliquer le segment audio un nombre de fois donné.

Ajouter et supprimer des enfants.

Cloner la transformation composite et ses commandes enfant.

Retourne le debut et la fin du conteneur.

Component : absTransform

Responsabilitees :

Definie l’interface uniforme qui :   
 effectue la transformation

Ajoute et supprime un enfant  
 Retourne le debut et la fin du conteneur.

1. L’attribut de type transformContainer du compositeTransform est responsable de la creation de l’arbre des composantes.

3-Patron Composite :

1. L’intention du parton proxy est de créer un conteneur pour un objet avec lequel le programme doit interagir. Dans ce cas les objets en question sont des fichiers « audio ». MemAudioFile.cpp implémente une façon de stocker les données nécessaires dans la mémoire sans avoir besoin de relire les fichiers binaires à chaque fois que les données en question doivent être utilisés.

Un des avantages les plus importants est aussi la facilitation d’interaction et manipulation des données grâce au patron proxy. MemAudioFile.cpp implémente justement un itérateur pour parcourir les « chunks ».

Tout ceci non-seulement rends les interactions moins couteuses mais aussi plus simples à faire.

1. 