# Design patterns à utiliser

◌ Decorator : Le pattern Décorateur (Decorator) attache dynamiquement des responsabilités supplémentaires à un objet. Il fournit une alternative souple à l’héritage, pour étendre des fonctionnalités.

● Factory : Elle permet d'instancier des objets dont le type est dérivé d'un type abstrait.

◌ Abstract factory : Plusieurs fabriques peuvent être regroupées en une fabrique abstraite permettant d'instancier des objets dérivant de plusieurs types abstraits différents.

◌ Builder : créer des objets complexes à partir d’autres objets

◌ Bridge : découpler l'interface d'une classe et son implémentation.

● Composite : Permet de concevoir une structure d’arbre

● Iterator : Parcourir une liste sans accéder

◌ Template : Fixer clairement des comportements standards qui devraient être partagés par toutes les sous-classes, même lorsque le détail des sous-opérations diffère. Factoriser du code qui serait redondant s'il se trouvait répété dans chaque sous-classe.

● Adapter : Convertit l’interface d’une classe en une autre interface

● Visitor : ajouter une nouvelle méthode virtuelle à un ensemble de classe qui ne le permet pas.

◌ Strategy : des algorithmes peuvent être sélectionnés à la volée au cours du temps d'exécution selon certaines conditions

● Façade : cacher une interface complexe et difficile à comprendre

◌ Memento : permet de restaurer un état précèdent d’un objet sans violer le principe d’encapsulation.