**Explication des designs patterns**

# Strategy :

Définir une famille d'algorithmes, encapsuler chacun d'eux et les rendre interchangeables. La stratégie permet à l'algorithme de varier indépendamment des clients qui l'utilisent.

Capturez l'abstraction dans une interface, enterrez les détails d'implémentation dans des classes dérivées.

# Singleton :

Assurez-vous qu'une classe n'a qu'une seule instance et fournissez un point d'accès global à celle-ci.

Initialisation "juste à temps" ou "initialisation à la première utilisation" encapsulée.

# Observer :

Définissez une dépendance d'un à plusieurs objets de sorte que lorsqu'un objet change d'état, toutes ses personnes à charge soient notifiées et mises à jour automatiquement.

Encapsuler les composants centraux (ou communs ou moteurs) dans une abstraction de sujet, et les composants variables (ou optionnels ou interface utilisateur) dans une hiérarchie d'observateur.

# **Builder :**

Séparer la construction d'un objet complexe de sa représentation afin que le même processus de construction puisse créer des représentations différentes.

Analyser une représentation complexe, créer une cible parmi plusieurs.

# Composite :

Composez des objets en arboresc ences pour représenter des hiérarchies de pièces entières. Le composite permet aux clients de traiter les objets individuels et les compositions d'objets de façon uniforme.

Composition récursive

"Les répertoires contiennent des entrées, chacune pouvant être un répertoire."

1 à plusieurs "a" en haut de la hiérarchie "est une" hiérarchie

Modèle / Vue / Contrôleur

Dans le modèle nous allons devoir mettre les différentes classes qui vont être considérés comme la base de données de notre programme, les classes du modèle seront utiliser dans les classes du contrôleur qui lui fera le lien avec la vue.

La partie Vue, c’est une partie qui existe pour faire le lien entre l’homme et le programme, c’est plus communément appeler l’IHM pour interface homme machine. Pour cette partie nous avons décidés d’utiliser le moteur graphique MonoGame qui est un moteur graphique adapté pour le C# ce qui nous est demandé dans le projet.

Le Contrôleur est la partie qui fait le lien entre la partie modèle et la partie vue, il fait appel aux différentes classes du modèle pour l’intégrer dans la partie vue ce qui rend une application qui permet des interactions avec des données. Dans notre cas nous allons devoir faire le lien avec les recettes, ingrédients, les employés, etc … puis l’intégrer avec notre modèle graphique du restaurant.