Feuille de TP 2 – Conception et Programmation de Composants MAY

L'objectif de cette deuxième feuille de TP est de continuer à se familiariser avec le concept de composant et avec la plateforme MAY. Le code d'une application de vente en ligne est fourni et devra être « refactorisé » pour accroître sa modularité, puis réutilisé pour obtenir une implémentation en composants MAY/SpeADL.

Étape 1 – Préliminaires

1. Considérez le projet Java « ComFlex_TP2_Obj » dans Eclipse où doit être importé le code des 12 classes liées à l'application de vente en ligne, à partir du « listing Estore Java » fourni sur Moodle.
2. Répartissez ces différentes classes dans les paquetages Java « exceptions », « impl » (implémentation des services), « datatypes » (structures de données manipulées par les services — choix proposé ici : Order, Cart, ItemInStock).
3. Le code obtenu est-il facilement maintenable/réutilisable/extensible ? Si non, quel est à votre avis le principal obstacle à ces critères ?

Étape 2 – *Design Pattern Bridge* et injection de dépendances

[cf. cours / *design pattern* **pont**, et cours Java EE / **injection de dépendances**]

1. Faites ressortir du code obtenu à l'étape 1 les principales **interfaces** et définissez-les dans le package « interfaces », puis utilisez le **patron de conception pont** pour réduire le couplage entre les classes en utilisant les interfaces identifiées (ajout de clauses **implements**, changement de type des attributs…). Vous pouvez utiliser la commande de menu d'Eclipse Refactor > Extract Interface…
2. Pour réduire encore le couplage entre les classes, remplacez dans chaque classe le code de création des instances concrètes des autres classes par une méthode d'**injection de dépendances** « public void init(...) ».
3. Déplacez le code de la méthode « public static void main(...) » de la classe Client dans une classe Main où sera faite l'instanciation des objets concrets, l'injection de dépendances et l'exécution de la méthode « public void run() » de la classe Client.

Étape 3 – Composants MAY/SpeADL

1. Considérez le 2e projet Java « ComFlex_TP2_MAY » ayant « ComFlex_TP2_Obj » comme dépendance (clic droit > Propriétés > Java Build Path > Projects > Add…). Idéalement l'implémentation en composants de ce 2e projet ne devra contenir aucun « code métier » et réutiliser les classes du 1er projet telles quelles. Le cas échéant, cela validera l'objectif visé par le « refactoring objet » de l'étape précédente.
2. Créez un fichier SpeADL décrivant l'architecture du projet en termes de composants. Note : l'interface du port fourni par le Client pourra être « Runnable ».
3. Réalisez l'implantation de chaque composant MAY en réutilisant sans modification les classes du projet « ComFlex_TP2_Obj ». Les injections de dép. seront faites à l'initialisation des composants dans une méthode @Override protected void start().
4. Créez une classe Main avec une méthode main où sera effectuée l'instanciation de l'assemblage global et l'exécution de la méthode « public void run() » de son port fourni.