UQAC

Département d'informatique et de mathématique
Université du Québec à Chicoutimi

TP : ADAPTATION ET QUALITÉ LOGICIELLE

Département d'Informatique et de Mathématique (DIM)
Adaptation et qualité logicielle (8INF228)
Automne 2023

PhD Sc. TI: K. Marcelin	Ce travail est en équipe de 2 ou 3 étudiants
Semestre : Automne. 2023	Date de distribution : 18 septembre 2023
Pondération : 15 points	Date de remise : 4 décembre 2023

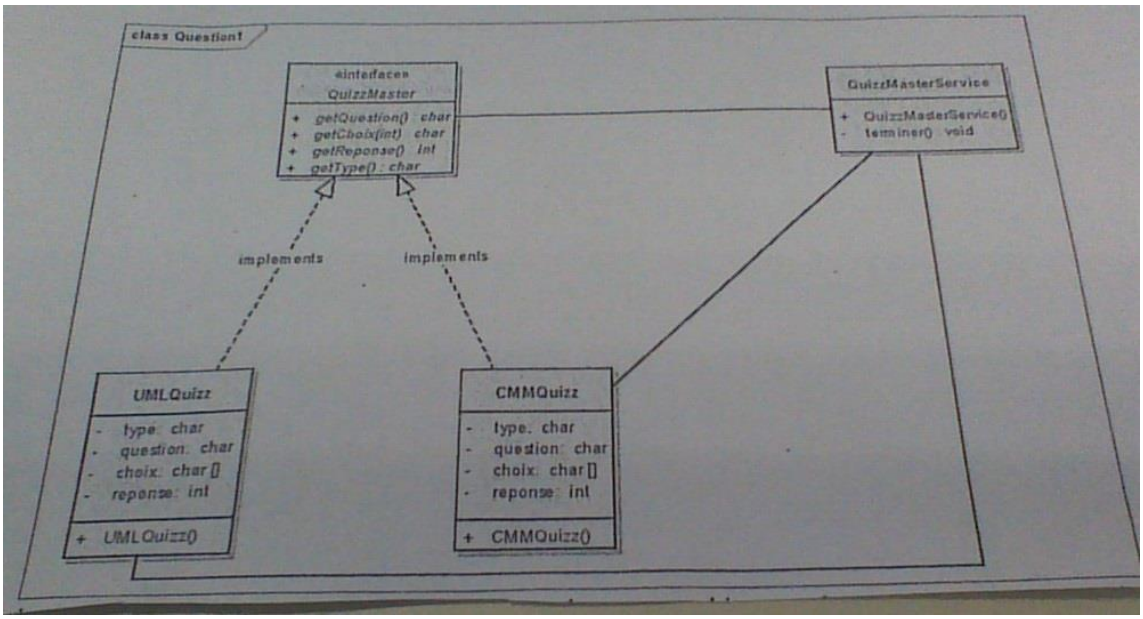
Le but du TP est de familiariser l'étudiant avec les concepts suivants :

- L'adaptation d'une application logicielle, Qualité Logicielle
- Le patron de conception et l'injection de dépendance
- La modification de l'architecture pour diminuer le couplage des applications
- La séparation des préoccupations : AOP, AspectJ

Note : Dans ce T, Vous devez choisir de faire la question1 ou la question2 mais pas les deux à la fois.

Question 1 : (15 points)

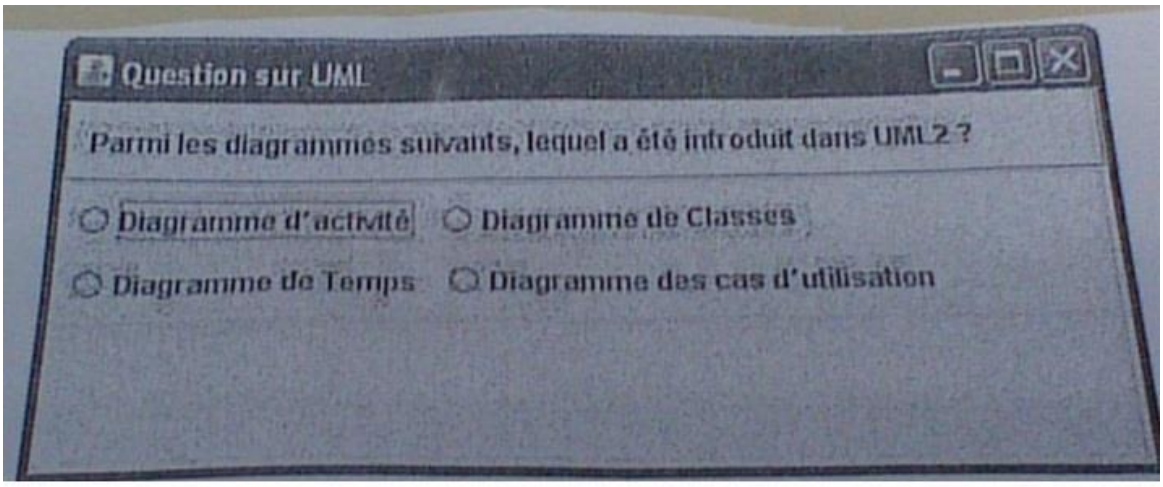
Bill Gate, un ingénieur logiciel, désire réaliser un logiciel de quiz pour un professeur en informatique de l'UQAC. Dans un premier temps, il développe une première version Java du logiciel en se basant sur le diagramme de classes suivant :



Cependant, Bill n'est pas satisfait car il trouve, que dans son architecture, le couplage entre les différents composants est fort, ce qui à la longue pourrait lui compliquer les opérations de maintenance du logiciel. Par exemple, pour créer une instance QuizzMaster dans la classe QuizzMasterService, il a fait :

```
private QuizzMaster quizMaster= new UMLQuizz(); // (ligne UML)
```

L'exécution de l'application donnerait alors comme résultat :



Pour afficher le questionnaire sur le modèle CMM, il faudrait modifier la classe QuizzMasterService de la manière suivante : (remplacer la ligne UML par la ligne CMM)

```
private QuizzMaster quizMaster= new CMMQuizz();// (ligne CMM)
```

Ayant entendu dire que vous étiez des experts Spring, Bill vous demande de modifier son application en utilisant le design pattern « injection de dépendance ».

Le chef de projet vous demande :

1) *Évaluer le modèle au principes SOLIDS ;*
2) *Évaluer le couplage entre les classes ;*
3) *Ajouter une ou des règles qui utilise l'approche OCL ;*
4) *Proposer deux nouvelles versions Spring de l'application de Bill, en injectant les dépendances à l'aide de constructeurs et l'autre à l'aide de setters. Vous trouverez le code source de l'application originale de Bill avec ce travail.*
5) *Réaliser les tests, Afficher les résultats*
6) *Imaginez que ce projet a une équipe de développement et une équipe de production, comment mettre en œuvre une approche devOps ? Expliquez.*

Question 2 : (15 points)

Dans le cadre du suivi d'un compte bancaire d'un client, le chef de projet informatique d'une banque, a initié un projet intitulé :'ProjetObserverDesignPattern'. Dans ce projet, le chef projet a exigé l'implémentation du Design Pattern Observer ou Publish Suscribe. La modélisation complexe d'un diagramme classe est interdit. Une modélisation simple pour implémenter Le patron Observateur est souhaité.

Le compte bancaire peut changer d'état en fonction des transactions bancaire du client. Les informations qui caratérisent le compte sont : numeroCompteClient, montantDebit, montantCredit . Ce compte peut prendre trois états selon certaines conditions :

E1 : Compte Debiteur, si le solde du compte est négatif
E2 : Compte Créditeur, si le solde du compte positif
E3 : Compte Etat nul, si le solde du compte est

L'objectif du projet est d'informer automatiquement le client et les autres observateurs autorisés les changements d'état du compte.

Les personnes autorisées à observer ou à recevoir ces alertes sur le compte sont :

Le client, le gestionnaire du client, la conjointe du client et le conseiller financier du client.

Le chef de projet vous demande donc de réaliser les travaux suivants :

1) Proposer un modèle qui utilise au moins 2 principes SOLIDS et Proposer des diagrammes qui utilisent une ou des règles (OCL)

2) Écrire les implémentations du sujet pour permettre aux observateurs du compte du client de recevoir les messages d'alertes ;

3) Écrire l'implémentation de l'observateur pour recevoir les messages ; Proposer les méthodes nécessaires dans les différentes.

4) Utiliser les injections de dépendances par constructeur avec Spring IoC pour créer les différents objets observateurs dans le programme principal

5) Réaliser les tests, Afficher les résultats

6) Imaginez que ce projet a une équipe de développement et une équipe de production, comment mettre en œuvre une approche devOps ? Expliquez.

Note : Vous pouvez créer une liste d'observateur. Vous pouvez appuyer sur le code de l'exemple vu en classe. Plateforme implémentation free cependant assurez-vous de disposer des bibliothèques et classes pour implémenter le DesignPattern Observer.

Livrable Question 1 ou Question 2

1) *Document Word (voir modèle de document uqac)*
   Diagramme, Plan deTests, Résultat, printscreen des écrans
2) *Code dans un éditeur de texte (note pad++)*
3) *Code source dans un dossier compressé*
4) *Démo en classe avec le prof ou une vidéo claire avec un plan de test*