IFT 2255 –Génie logiciel

UdeM/DIRO, Automne 2014

Travail pratique #3

1. Introduction

Analyse du problème de système de gestion de fichier (Diagrammes de classes, patrons de conception, localisation des concepts, et analyse d'impacts).

2. Conditions de réalisation

Groupe de deux ou trois personnes. Travail à remettre au plus tard le mardi 9 décembre au début de la séance de démonstration. Aucun retard ne sera accepté.

La solution (les diagrammes, l'implémentation en Java, les réponses, etc.) doit être imprimée et donnée directement au démonstrateur. **De plus,** les fichiers .java de votre solution doivent être compressés dans un fichier [VosNoms]_IFT2255_TP3.zip et envoyés par courriel au démonstrateur.

3. Travail à réaliser

En vous basant sur la spécification donnée dans la section 5:

1. Définir le diagramme de classe qui correspond à la spécification 5.a.
2. Modifiez ce diagramme de classe de façon à ce qu'il réponde aux exigences de la spécification 5.b (cette solution doit être évolutive)
3. Proposez ensuite une conception permettant à une classe client de disposer de la taille d'un répertoire sans avoir à modifier les classes existantes (i.e., les classes Element, Dossier, et Fichier).
4. Proposez une implémentation de votre diagramme de classe qui réponde aux questions 2 et 3, en utilisant le langage Java (le code doit absolument correspondre **exactement** au diagramme de classe proposé). Incluez également dans votre implémentation une classe Main.java qui contient la méthode **main** et des exemples qui illustrent votre solution.
5. Pour chacune des requêtes de changement dans la section 6 :
    - Identifiez les concepts pertinents
    - Précisez l'impact (les classes, les méthodes, et/ou les attributs qui ont été impactés par le changement)
    - Précisez si l'impact de changement est localisé ou pas et donner une explication précise (l'impact sur la fonctionnalité et la portée de l'impact).
    - Modifiez votre implémentation pour répondre au changement et commenter tous les entités qui ont été changées.

NB : Pour chaque diagramme, commentez vos choix et les décisions importantes que vous avez prises lors de l'élaboration du diagramme.

4. Barème

1. Diagramme de classe 20 %
2. Diagramme de classe modifié (évolutif) 20%
3. Diagramme de classe résolvant le problème de la taille 15 %
4. Implémentation 15%
5. Concepts, Changement et Impact 20%
6. Qualité du document d'analyse (structure, français, etc.) 10%

5. Spécification

a. Dans un système de gestion de fichier, un répertoire est composé de fichiers ou de répertoires aussi. Il est clair que le navigateur de fichiers doit toujours rester cohérent avec le contenu réel du répertoire vers lequel il est pointé, i.e. toute modification dans le répertoire (suppression, création de fichiers, etc.) doit être immédiatement reportée au navigateur afin que le navigateur mise à jour les informations affichées sur les répertoires actuellement ouverts.
Comme précisions :

- Les classes principales du système sont les 4 classes suivantes : Element, Dossier, Fichier, et la classe Navigateur qui joue le rôle du client.
- Un navigateur peut gérer plusieurs dossiers ouverts en même temps.
- **Mais** un seul dossier ouvert peut être caractérisé par le navigateur comme étant *active* (i.e., activé par l'utilisateur en le sélectionnant). Donc, si un dossier ouvert est sélectionné, il doit immédiatement notifier le navigateur afin que le navigateur remette à jour le dossier actif actuellement.
- **Il est important de noter** qu'un Navigateur donné ne peut avoir qu'une instance unique dans le système.
- Toutefois, le système doit permettre aux différents Navigateurs de s'intégrer au système sans avoir besoin d'introduire de changements majeurs.

b. Ce système de gestion de fichier évolue avec le temps. Ainsi sur chaque dossier ou fichier, nous pouvons ajouter un nombre quelconque de caractéristiques (exemple : intelligent, évolutif). Une première idée consiste à mettre en place une classe abstraite Element par exemple ayant les attributs (Nom et date de création, path, date de modification) et les accesseurs en lecture/écriture correspondants. Puis, créer pour chaque combinaison d'éléments et de caractéristiques une classe (DossierIntelligent, DossierEvolutif, FichierIntélligent, FichierEvolutif). Bien sûr cette solution n'est pas évolutive. Si on ajoute une dizaine de caractéristiques nous allons obtenir une centaine de classes (nombre de combinaisons).

NB : Intelligent : permet par exemple à un dossier de s'auto évaluer chaque période de temps, et de proposer des améliorations (certaine organisation) à l'utilisateur s'il y en a.

6. Changement, concepts, impacts

Première requête : le système permet la création des raccourcis et les répertoires peuvent contenir des raccourcis, où un raccourci est un élément qui peut pointer sur un dossier ou fichier (mais pas sur un autre raccourci!).

Sachez qu'un raccourci doit connaitre son élément (fichier ou dossier) mais l'inverse est faux (i.e., un fichier/dossier *ne doit pas* connaitre les raccourcis qui pointent vers lui).

Deuxième requête : supprimer un élément (un dossier, ou un fichier) entraine la suppression de tous les raccourcis qui y pointent.

Notez que la gestion des raccourcis dans le système est représentée par la classe GestionnaireRaccourcis qui connait tous les raccourcis dans le système.