IFT785 – Approches Orientées Objets

Devoirs – patrons de conception

Déroulement

- Devoir 1, tests, refactorisation et State DP : Q1
- Devoir 2, Abstract Factory DP : Q2.
- Devoir 3, Singelton DP, Observer DP et Adapter DP : Q3
- Devoir 4, State DP, Command DP, Composite DP: Q4 et Q5
- Devoir 5, Observer DP (vetoable) : Q6

Notation

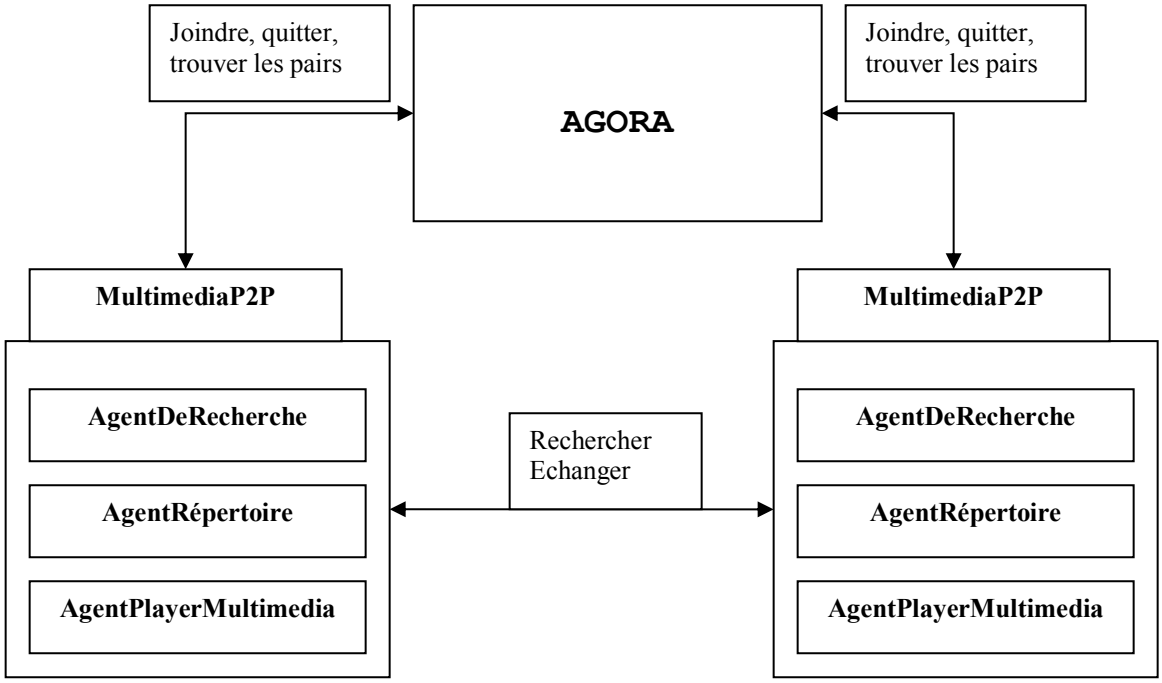
- 0 : pas de remise
- 1 : service minimal
- 2 : problème résolu
- 3 : impeccable (bonus)

PARTIE PRATIQUE (70 points)

Étude de cas : Du multimédia partagé…

Vous avez été engagé par XULIN pour faire un système d'échange P2P (peer-to-peer) de contenu multimédia. Ce système comporte les éléments suivants :

- Une place centrale `Agora` où les agents se connectent
- Pour chaque nœud participant, un agent MultimediaP2P qui sert de point d'entrée au nœud. Cet agent gère trois sous-agents :
  - `AgentDeRecherche`
  - `AgentPlayerMultimedia`
  - `AgentRepertoire`



Première étape : Ecouter du multimédia

Question 1 : Refactorisation et poly morphisme

Comme première étape, vous devez construire un agent capable d'utiliser le « player » approprié (vidéo, MP3, WAV, etc.) pour la plateforme où il s'exécute (Apple, Windows, etc.). Cet agent est capable de contrôler le nombre d'utilisation du contenu. Les contenus multimédias sont soit vendus ou gratuits (nombre illimité d'utilisation), soit loué pour un certain nombre d'utilisation. Une utilisation correspond soit à une écoute complète (i.e. la fonction « stop » du lecteur a été invoquée), soit à une écoute incomplète non reprise, i.e. l'utilisateur a appuyé sur la fonction « pause » et n'a jamais repris l'écoute). Ainsi la séquence PLAY, PAUSE, RESUME, PAUSE correspond à une utilisation.

Refactoriser le programme suivant de manière à faire disparaître, autant que faire se peut, les conditionnelles qu'il contient en les remplaçant par du polymorphisme.

Avant de commencer la refactorisation, écrire au moins 10 tests unitaires pour s'assurer que le comportement du programme sera préservé. Il se peut que à la suite des autres questions, vous ayez à modifier et adapter vos tests.

```
public class AgentPlayerMultiMedia {

    final static private int VIDEO = 0;
    final static private int MUSIQUE = 1;

    final static private int CREATED = 0;
    final static private int STARTED = 1;
    final static private int PAUSED = 2;
    final static private int RESUMED = 3;
    final static private int STOPPED = 4;

    private int etat_;                    private int type_;
    private int maximum_;                 private int joue_;
    private boolean location_;            private String titre_;
    private Object contents_;

    private MediaPlayer player_;

    public AgentPlayerMultiMedia(  boolean location, int nbFois,
                             int type, String titre,
                             Object contents) {
        location_ = location;              maximum_ = nbFois;
        titre_ = titre;                    contents_ = contents;
        joue_ = 0;                         etat_ = CREATED;
    }

    public void start() {
        if (etat_ != CREATED) { return;  // on commence à jouer le contenu
        }
        if (location_ && joue_ < maximum_) { return;
        }
```

```
        if (type_ == MUSIQUE) { player_ = new iTunes();
                                player_.play(this);
                                etat_ = STARTED;
        }
        if (type_ == VIDEO) {  player_ = new QuickTime();
                               player_.play(this);
                               etat_ = STARTED;
        }
    }

    public void pause() {
        //le contenu doit jouer pour que cette opération soit valide
        if (etat_ == PAUSED) { return;
        }
        if (etat_ == STOPPED_) { return;
        }
        if (type_ == MUSIQUE) { player_.pause(this);
                                if (location_) { joue_ = joue_ + 1; }
                                etat_ = PAUSED;
       }
        if (type_ == VIDEO) { player_.pause(this);
                              if (location_) { joue_ = joue_ + 1; }
                              etat_ = PAUSED;
        }
    }

    public void resume() {
        // cette opération n'est valide qu'après une pause
        if (etat_ != PAUSED) { return;
        }
        if (location_) { { joue_ = joue_ - 1; }
        }
        if (type_ == MUSIQUE) {  player_.play(this);
                                 etat_ = RESUMED;
        }
        if (type_ == VIDEO) {    player_.play(this);
        }
    }

    public void stop() {
        //le contenu doit jouer pour que cette opération soit valide
        if (etat_ == PAUSED) { return;
        }
        if (etat_ == STOPPED) { return;
        }
        if (type_ == MUSIQUE) {  player_.close(this);
                                  if (location_) { joue_ = joue_ + 1; }
        }
        if (type_ == VIDEO) {   player_.close(this);
                                if (location_) { joue_ = joue_ + 1; }
        }
    }
}
```

Question 2 : Multi-plateforme

Actuellement le système ne fonctionne que pour MAC OS X (les players utilisés sont itunes et Quicktime). Modifier ce programme pour le faire fonctionner sur de nouveaux systèmes d'exploitation. Par exemple sous Windows, les lecteurs audio et vidéo seraient alors « Windows Media Player »

En Java, les propriétés du système permettent d'avoir de l'information sur le système d'exploitation utilisé.

```
System.getProperty("os.name")
http://java.sun.com/docs/books/tutorial/essential/system/properties.html
```

Remarque : Vous n'avez pas à utiliser ou exécuter les « players » réels.

Question 3 : Suivi de la lecture du document Multimedia

Vous disposez d'une classe `Afficheur` ci-dessous qui permet de suivre sur la console l'exécution des contenus multimédia.

- Complétez la classe `Afficheur` pour qu'elle implémente un Singleton.

- Modifiez le programme pour faire en sorte que l'agent **AgentPlayerMultimedia** informe l'afficheur de son changement d'état. Mis à part les modifications pour implémenter le singleton, vous NE devez PAS modifier la classe `Afficheur` pour implémenter cette fonctionnalité.

```
public class Afficheur  {
    static public Afficheur getAfficheur() {…}  //Singleton

    public void display(String titre, String etat)   {
      System.out.println(titre + " >>> " + etat);
    }
}
```

- Modifiez maintenant la classe `Afficheur` pour qu'elle s'affiche sur une autre sortie par exemple un fichier de Log.

Question 4 : Ajout d'un nouveau type de média

Un nouveau type de média « 3Dmov » a été développé. Il s'agit de vidéo 3D ayant comme particularité de permettre de modifier le point de vue du spectateur. Lorsque le film joue (état STARTED ou RESUMED, on peut passer en mode « GLOBAL_SCENE ». Lorsque l'on est en mode « GLOBAL_SCENE », la seule opération valide est « resume ». Le mode « GLOBAL_SCENE » n'est valide que pour ce type de média. Modifiez le programme en conséquence.

Deuxième étape : Recherches de fichiers multimédia

A l'étape suivante, vous devrez implémenter un agent `AgentRecherche` qui reçoit des requêtes et qui interroge les agents répertoires des sites participants au réseau. Pour le moment, il s'agit de construire la partie qui permettra d'effectuer les requêtes.

Question 5 : Spécification d'une requête

Une requête simple correspond à un prédicat spécifiant l'auteur, le titre ou le statut (gratuit, location, vente). Une requête complexe correspond à des conjonctions / des disjonctions de prédicats. Par exemple, pour avoir toutes les chansons gratuites de Eminem, la requête serait

- mm.getAuteur() == « Eminem » AND mm.isGratuit()
- mm étant ici une instance de `Multimedia` (voir plus bas)

Implémenter un système permettant de construire et exécuter des requêtes simples et complexes. On devra pouvoir ajouter ultérieurement (mais vous n'avez pas à le faire pour l'examen) des nouveaux prédicats, par exemple « catégorie » et de nouveaux opérateurs logiques, par exemple NOT.

Remarque :

- une requête complexe peut être représenté par un arbre.
- Vous disposez des interfaces suivantes

```
public interface Multimedia {
    String getTitre();
    String getAuteur();
    boolean isGratuit();
    boolean isLocation();
    boolean isAchat();
    String getProprietaire();
}

public interface Predicat {
    Multimedia searchTitre(String unTitre);
    Vector searchAuteur(String unTitre);
    Vector searchGratuit();
    Vector searchLocation();
    Vector searchAchat();
    Vector searchProprietaire(Vector owners);

}
```

Troisième étape : Réseau d'échange de fichiers musicaux

Question 6 : Réseau d'échanges P2P

Dans cette section, vous devez implémenter un réseau d'échanges P2P de fichiers musicaux. Chaque participant `MultimediaP2P` possédera trois agents internes: `AgentDeRecherche`, `AgentRepertoire`, `AgentPlayerMultimedia`.

Un registre central unique des agents participants **`Agora`** maintiendra à jour la liste des participants. Le registre central assume les fonctions suivantes :

- fournir la liste de tous les participants sur demande
- informer les participants de la connexion d'un nouveau participant
- informer les participants de la déconnexion d'un participant

Les fonctionnalités d'un participant **`MultimediaP2P`** sont

- A sa création,
  - il crée ses trois agents internes
  - il se connecter au réseau d'échange
- Pendant sa vie active,
  - il donne accès à l'agent interne approprié pour faire des recherches, répondre aux requêtes, jouer des contenus multimédias.
  - imposer un veto sur l'échange de certains documents multimédias, par exemple si le demandeur fait partie d'une liste noire.
- A sa destruction,
  - il se déconnecte du réseau d'échange
  - il arrête ses agents internes

Les fonctionnalités d'un agent **`AgentRepertoire`** sont

- Répondre aux requêtes des autres participants
- Gérer la liste des documents multimédia
- Échanger des fichiers multimédia.
  - Un document qui est en train de jouer ne peut pas être échangé
  - Lorsque le fichier est gratuit, il en transmet une copie au demandeur.
  - Lorsque le fichier est loué ou acheté, il transmet le fichier au demandeur et efface le fichier de son répertoire
- Supprimer un document loué dont le nombre d'écoute permise a été utilisé.

Les fonctionnalités d'un agent **`AgentDeRecherche`** sont

- effectuer des recherches de documents multimédia chez les autres participants
- obtenir les fichiers multimédia rendus par la recherche,
  - Si le fichier existe sur plusieurs sites, il ne faut en obtenir qu'un seul exemplaire.