Université Claude Bernard Lyon 1 4A,
Département Informatique - Polytech Lyon

ISI3

Examen

Durée : 2 h — Seul document autorisé: la fiche *Récapitulatif des Patrons de conception* distribuée en cours.

---

1 Conception de jeux de tests

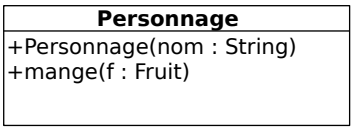
Question 1 *Proposez un partitionnement en classes d'équivalences valides (analyse partitionnelle) du domaine d'entrée d'une* ***fonction calculant la valeur absolue d'un entier****.*

Question 2 *Proposez un partitionnement en classes d'équivalences valides (analyse partitionnelle) du domaine d'entrée d'une* ***fonction calculant la distance entre deux points du plan*** *(*$f((x,y),(x',y')) = \sqrt{(x-x')^2 + (y-y')^2}$*).*

Question 3 *Proposez des* ***tests aux limites*** *pour une fonction qui incrémente un entier.*

2 Principes de conception orientée objet

On dispose de la classe `Personnage` suivante :



Dans le cadre d'une application permettant à des personnages de manger des fruits, les classes suivantes sont fournies :

```
public class ArbreFruitier {
  private String type;
  public ArbreFruitier(String type) {
      this.type = type;
   }

  public Fruit cueille() {
   if (this.type.equals("pommier") {
      return new Fruit("pomme");
```

```
    else if (this.type.equals("poirier") {
       return new Fruit("poire");
    }
  }

  public String toString() {
   return this.type;
  }
}

public class Fruit {
  private String type;
  public Fruit(String type) {
    this.type = type;
  }
  public String toString() {
    return this.type;
  }
}

public class Main {
  public static void main(String[] args) {
    ArbreFruitier pommier = new ArbreFruitier("pommier");
    ArbreFruitier poirier = new ArbreFruitier("poirier");
    Personnage maraudeur = new Personnage("Merry");
    maraudeur.mange(pommier.cueille());
    maraudeur.mange(poirier.cueille());
  }
}
```

Question 4 *Critiquez la conception illustrée par ce code. Quel principe de conception orientée objet n'est pas respecté ? Justifier.*

Question 5 *Corrigez les erreurs de conception que vous avez relevées en modifiant le code et en proposant un patron de conception. Vous donnerez le diagramme UML de votre solution ainsi que le détail de l'implémentation. Vous préciserez en quoi votre solution respecte cette fois-ci le principe de conception orienté objet non respecté par la proposition initiale.*

3 Robocup Soccer Simulation

Dans cette partie, nous allons explorer l'utilisation des *design pattern* dans le contexte d'une application similaire à celle de la *Robocup Soccer Simulation*. La *RoboCup Soccer*[1] est un tournoi international de robotique qui a lieu tous les ans. L'un des objectifs de ce tournoi est d'arriver à créer une équipe de football robotisée (cf. figure 1(c)) capable de battre l'équipe de football "humaine" championne du monde d'ici à 2050. Une des divisions de la *RoboCup Soccer* est la ligue *Simulation*[2] qui permet de simuler des parties de football entre deux équipes de 11 joueurs robots simulés (cf. figure 1(a) et (b)).

1. http ://www.robocup2014.org/
2. http ://wiki.robocup.org/wiki/Soccer_Simulation_League

(a) Simulation 2D

(b) Simulation 3D

(c) Plateforme Standard (robots Nao)

FIGURE 1 – Exemples de divisions dans la RoboCup Soccer

Dans l'application de simulation que l'on souhaite développer ici, le jeu de football est représenté par une classe concrète `JeuFoot` composée des classes abstraites suivantes : `Joueur`, `Ballon`, `Terrain`, `But`. La classe concrète `AffichagePartie` gère l'affichage graphique de la partie. Les différentes composantes du jeu de foot peuvent être créées comme des éléments 2D ou 3D. La figure 2 propose un diagramme de classes partiel pour cette application.

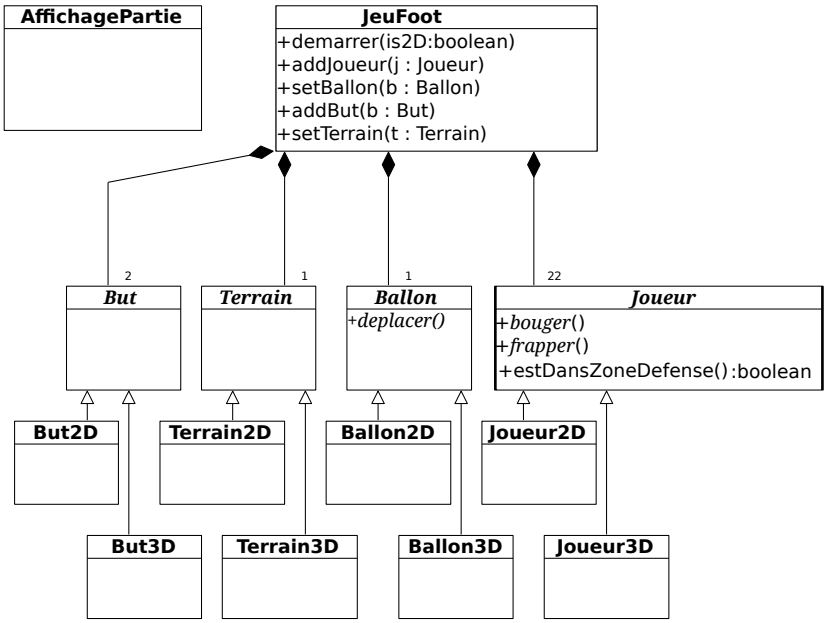


FIGURE 2 – Diagramme de classes (partiel) de l'application

Les questions suivantes sont indépendantes les unes des autres. Pour chaque question, vous devez proposer un diagramme de classe en complétant le diagramme de classe donné à la figure 2 et préciser le ou les patterns utilisé(s). Les classes de votre diagramme doivent présenter toutes les opérations nécessaires au fonctionnement demandé à chaque question. Le diagramme doit également montrer les associations, multiplicités et dépendances entre classes. Vous préciserez l'implémentation des méthodes nécessaires à la compréhension de votre conception.

La création d'un jeu 2D ou 3D dépend du choix de l'utilisateur lorsqu'il démarre une partie (méthode *demarrer(boolean is2D)* de la classe `JeuFoot`). Par conséquent, il est recommandé que la classe `JeuFoot` exploite le pattern **Abstract Factory** pour instancier les différentes composantes du jeu. La classe abstraite pour l'usine de création des composantes du jeu doit être unique dans l'application.

Question 6 *Proposer une implantation du pattern Abstract Factory en précisant le code des classes ajoutées ainsi que le code de la méthode* `demarrer(boolean is2D)`.

L'interface publique de la classe `Joueur` offre deux comportements de base (qui sont implantés par les classes concrètes `Joueur2D` et `Joueur3D`) : *bouger()* (pour déplacer le joueur) et *frapper()* (pour taper le ballon).

On s'intéresse dans un premier temps aux joueurs 2D. Le comportement d'un joueur 2D est spécifique selon sa position sur le terrain : s'il est dans la zone de défense de son équipe, il agit comme un défenseur ; sinon il agit comme un attaquant.

Question 7 *Proposer un pattern et détailler son implantation pour réaliser cette fonctionnalité.*

On s'intéresse maintenant aux joueurs 3D. On aimerait utiliser la classe (non-modifiable) `Joueur3DNao` (cf. figure 3) qui permet de simuler parfaitement un robot humanoïde. Avant de frapper, le joueur3DNao doit se positionner, lever une jambe, puis se stabiliser, puis avancer la jambe avant de la reposer.

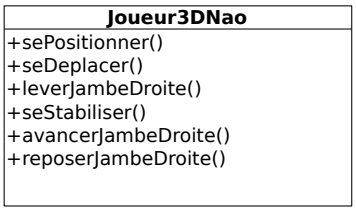


FIGURE 3 – Interface publique de la classe (non-modifiable) `Joueur3DNao`

Question 8 *Proposer un pattern permettant d'utiliser cette classe dans l'application et détailler l'implantation proposée.*

La classe `AffichagePartie` est responsable d'afficher régulièrement la position des joueurs et du ballon sur le terrain (ne vous souciez pas de la manière avec laquelle cet affichage s'effectue).

Question 9 *Proposer un pattern et détailler son implantation pour faire en sorte que l'affichage des joueurs et de la balle soit efficace.*