Rapport Test Driven Development
Reprise de projet

Albert Emile 14022
Anizet Thomas 14164
Lekens Amaury 14027
Selleslagh Tom 14164
Wéry Benoit 14256

08/12/2017

1 État de développement

- **Classe *Sensor***
  Un capteur donne l'état d'une place de parking (prise : *true* et libre : *false*)

  Le constructeur de la classe créé un dictionnaire clé-valeur avec le nom du capteur et son état - A l'initialisation tout les capteurs sont à *false*

  Les fonctions supplémentaires permettent de récupérer l'état d'un ou plusieurs capteurs ainsi que de créer des places supplémentaires

- **Classe Parking**
  Cette classe instancie un objet parking représenté qui possède plusieurs fonctions (gestion de la barrière, des codes d'ouverture et mise à jour du nombre de place

  Le point de la relation entre le parking et la classe *sensor* est que le pattern observateur n'est pas utilisé dans sa fonction première. En effet, l'observable (le *sensor*) ne transmet pas son état à l'observateur (le parking). Le résultat est une mise à jour brute de l'ensemble des états des places sous l'impulsion du parking (*polling*)

- *Tests unitaires* Les tests unitaires sont implémenté correctement mais sont peu claires et peu commentés

1.1 Critique

Selon nous, le *pattern* observateur n'est pas exploité ici. Cette sous exploitation implique une mise à jour brute de l'ensemble des états des places sous l'impulsion du parking

Notre vision (et l'implémentation qui va en résulter) est un abonnement de l'observateur à son observable. Si l'état d'un capteur change, celui-ci va notifier son observateur qui mettra à jour le nombre de place libre. (voir diagramme de classe)

1.2 Amélioration

- Ajout d'une classe **Zone** qui sera l'observateur
- Sensor devient l'observable
- la classe **parking** prend le rôle de gestionnaire.
- Pas de base de donnée. le programmme sera de type Super-Loop mettant à jour l'état au démarrage en parcourant les capteurs.
- Crée un code pour simuler un parking de base
- Inversion de true et false : une place libre est à true

2 Analyse de qualité

La qualité du code repris n'est pas évaluable selon les termes prévu car si les métriques sont expliqués et listé, aucune valeur n'est visée pour ces derniers. Avant de continuer le projet, il est donc nécessaire de fixer des valeurs pour les différents métriques prévus par le groupe précédent

3 Plan de travail

06.12.17 - Description du projet - Critique et première idée 08.12.17 - Réunion Objectifs

1. Repenser la classe Sensor - Amaury & Benoit
   Pattern Observateur
2. Repenser les critères de qualité - Emile
   Les métriques sont expliqués mais pas de critère
3. Code de simulation du parking Tom - Emile

4. Etoffer et vérifier les conventions de codages - Everybody
5. Clarifier et étoffer tests unitaires - Tom - Emile

6. Jenkins - Thomas