



**UNIVERSITÉ
CAEN
NORMANDIE**

Pour l'unité d'enseignement de Méthode de Conception
Licence 3 informatique

Jeu de stratégie au tour par tour

Auteurs :

Michel YABA BILONGO
Mamadou Bobo DIALLO
Jean Faya KOUROUMA
Desty MPASSI MATONDO

Encadrant :

Yann MATHET

Version 0.1
26 novembre 2018

0.1 Introduction

Ce document présente la réalisation d'un jeu de combat de stratégie au tour par tour qui oppose deux et plusieurs joueurs sur une grille de taille paramétrable en deux dimensions. L'objectif est d'appliquer ce que nous avons appris en Méthodes de Conception notamment l'utilisation des designs patterns vus en cours.

0.2 Vue globale

Pour la conception, nous avons utilisé une architecture **MVC (Modèle-Vue-Contrôleur)**. Elle est constituée de trois packages qui sont : le Modèle, la Vue et le Contrôleur. Les classes principales et les packages sont illustrés dans la figure suivante.

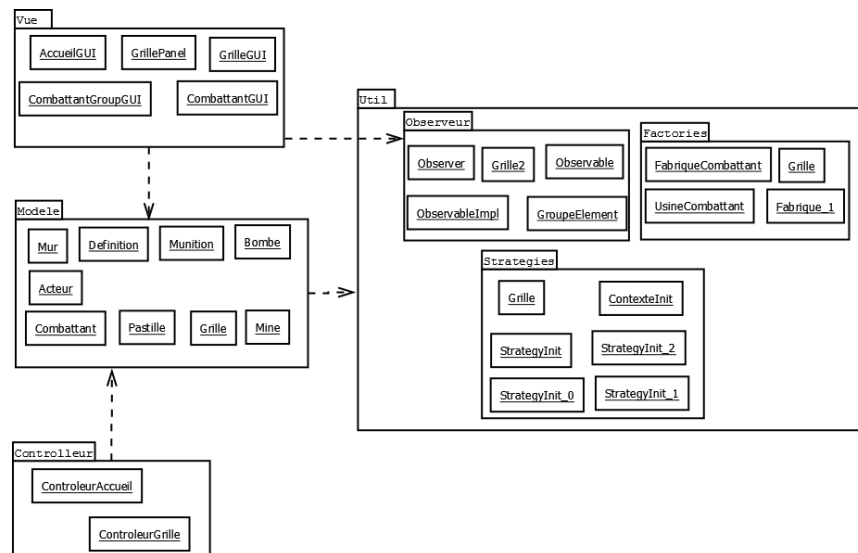


FIGURE 1 – *packages* avec leurs classes

0.3 Conception détaillée

Le *package* **modèle** contient neuf classes :

- Une classe **Munition** représente les munitions (balles) d'un combattant, lorsqu'elles touchent un combattant, elles diminuent son énergie.

- La classe **Mûr** représente un mûr dans la grille. Dans le plateau du jeu, un mûr peut protéger un combattant contre une balle, lorsque le mûr est entre deux combattants se tirant. Les mûrs délimitent le plateau du jeu.
- La classe **Définition** contient des constantes qui représentent les différents éléments de la grille.
- La classe **Combattant** définit un combattant avec toutes les actions qu'il peut effectuer.
- La classe **Pastille** représente une pastille dans une grille. Elle permet de recharger l'énergie d'un combattant.
- La classe **Grille** représente le plateau du jeu. Il contient les combattants, les pastilles, les bombes, les mines et les mûrs.
- La classe **Bombe** correspond à une bombe dans la grille. Elle est déposée par le combattant, la bombe explose au bout d'un temps donné.
- La classe **Mine** correspond à une mine dans la grille. Elle explose au contact avec un combattant.
- L'interface **Acteur** qui contient des méthodes communes des classes : Combattant, Pastille, Munition, Mine, Bombe.

Ensuite le *package* **controlleur** possède deux classes :

- Une classe **ControlleurAccueil** qui vérifie la véracité des données entrées (nombre de lignes, de colonnes et de combattant) par l'utilisateur.
- Une classe **ControlleurGrille** qui contrôle toutes les interactions entre les différents éléments de la grille.

Enfin le *package* **vue** contient cinq classes :

- La classe **AccueilGUI** est une interface graphique qui demande des informations sur le jeu dont le nombre de lignes, de colonnes et de combattants.
- La classe **GrilleGUI** qui affiche le plateau (grille) avec les informations du jeu.
- La classe **GrillePanel** qui dessine le plateau (grille) du jeu.
- La classe **CombattantGroupGUI** qui initialise un tableau qui contient les informations d'un combattant. **CombattantGUI** qui affiche les informations d'un combattant dans le plateau du jeu.

0.4 Répartition des tâches

Nous sommes répartis les différentes tâches du projet à l'aide de trello dont voici le lien :
<https://trello.com/b/E40AEyYD/planning-de-travail-et-repartition-des-t%C3%A2ches>

0.5 Options supplémentaires

0.5.1 Reinitialiser

Elle permet de relancer le jeu. En gardant, les données fournis au lancement du jeu.

Nous avons implementé une option.

0.6 Diagramme de classes

Ce sont les diagrammes de classes des différents designs patterns implémentés :

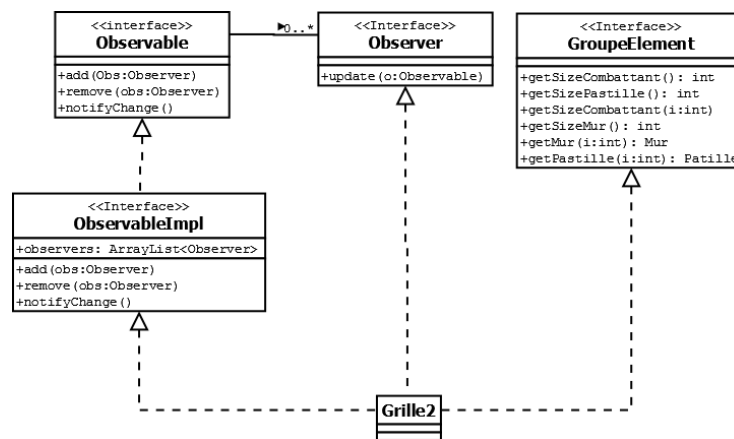


FIGURE 2 – *classes* du design pattern observer

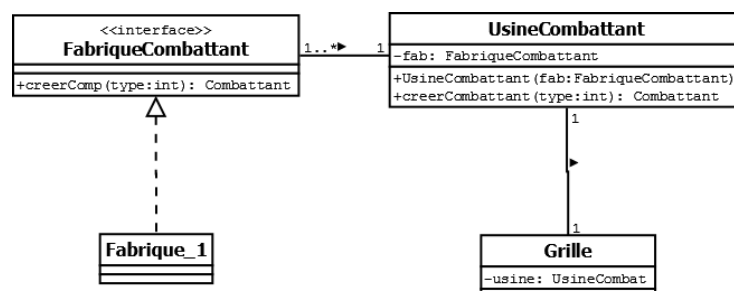
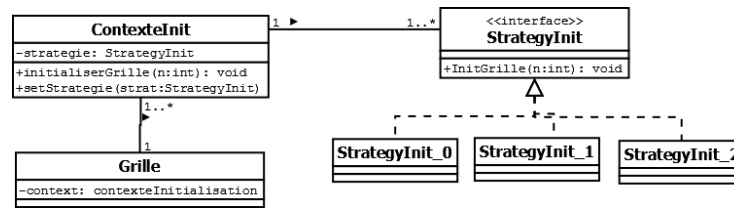
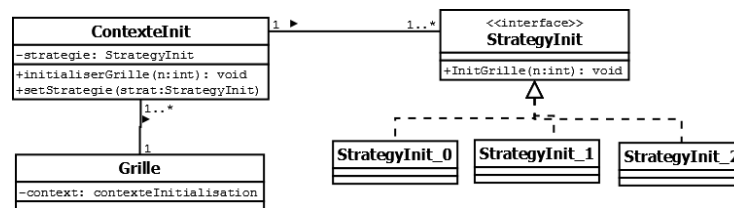


FIGURE 3 – *classes* du design pattern factory

FIGURE 4 – *classes* du design pattern stratégie (Grille)FIGURE 5 – *classes* du design pattern stratégie (Jeu)

0.7 Problèmes rencontrés

Pendant la phase de conception, nous avons rencontrés des difficulté comme l’attribution des tours entre combattants, l’implémentation des patterns Proxy et Observer.

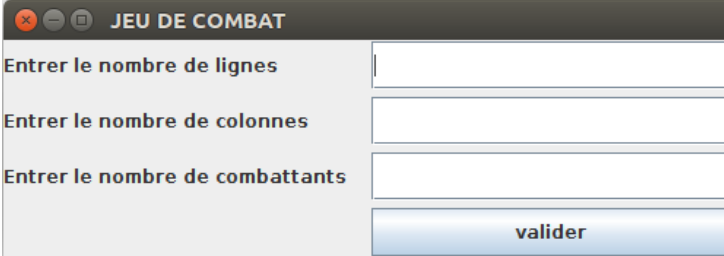
0.8 Solutions apportées

Pour pallier à l’attribution des tours entre combattants, nous avons choisi d’implémenter en attribuant un nombre aléatoire de 0 à n combattants et en fonction du combattant choisi un autre nombre aléatoire entre 0 et 7 est généré pour choisir l’action à effectuer.

0.9 Conclusion

Ce projet avait pour but de réaliser un jeu de combat de stratégie au tour par tour. A travers l’analyse, la conception et la réalisation du jeu, nous avons enrichis et consolider nos connaissances dans la manipulation des différents design pattern(MVC, Factory, Proxy, Strategy, Observer). Les challenges étaient nombreux, notre motivation nous a permis de surmonter à bien les difficultés rencontrées du projet afin de produire une application fonctionnelle qui respecte les règles du jeu.

0.10 Captures des écrans du jeu stratégie



JEU DE COMBAT

Entrez le nombre de lignes

Entrez le nombre de colonnes

Entrez le nombre de combattants

valider

FIGURE 6 – *Vue* au lancement du jeu.

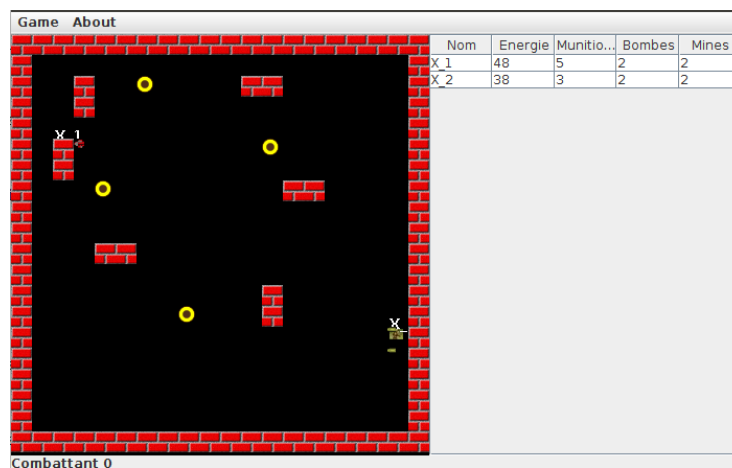


FIGURE 7 – *Vue* dans la stratégie par défaut.

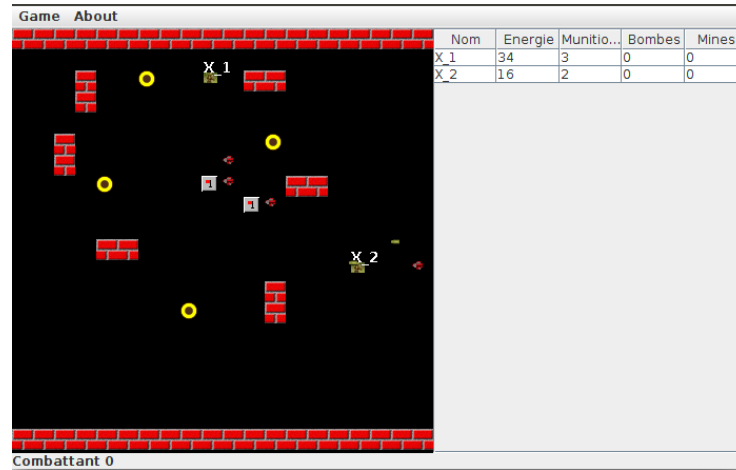
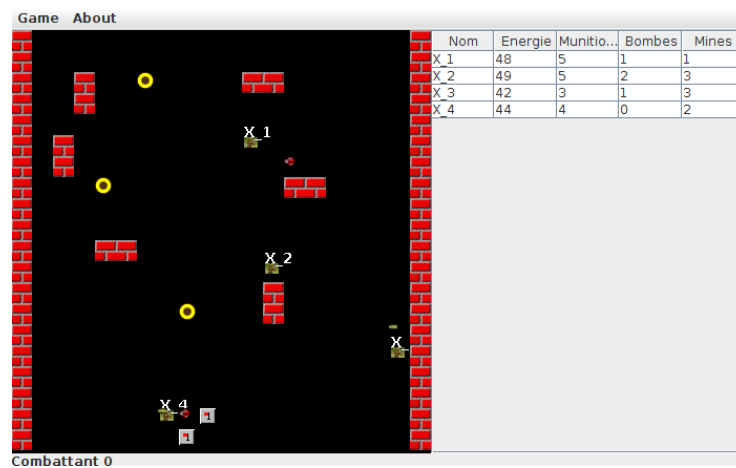
FIGURE 8 – *Vue* dans la stratégie 1FIGURE 9 – *Vue* dans la stratégie 2

FIGURE 10 – *Vue* partie terminée