SAE 1.01 Timeline – Rapport de projet

Hugo COLLIN, Xin ZHANG, Groupe Orange

I. Avancée du projet

Nous avons terminé de programmer, commenter et documenter les classes Carte, MainCarte, TestCarte, Paquet, TestPaquet, Frise et TestFrise, et nous venons de terminer les classes Jeu et ProgJeu.

II. Déroulement de notre travail

12 heures étaient prévues dans l'emploi du temps, ainsi qu'un TP préparatoire de 2 heures.

Voici la répartition de notre temps de travail :

2h	Découverte du sujet via un TP de préparation à la SAE, et réalisation des classes Carte et Paquet d'après le sujet de TD, afin de créer des paquets de cartes.
1h	Modification de la classe Carte du TP afin qu'elle corresponde aux consignes de la SAE, création de la classe MainCarte qui vérifie le fonctionnement de la classe Carte.
1h	Réalisation de TestCarte qui teste que la classe Carte a le comportement souhaité.
2h	Modification de la classe Paquet du TP afin qu'elle corresponde aux consignes de la SAE.
2h	Réalisation de TestPaquet qui teste que la classe Paquet a le comportement souhaité.
1h30	Codage de Frise qui crée une frise chronologique contenant un tableau de cartes.
1h30	Réalisation de TestFrise qui teste que la classe Frise a le comportement souhaité.
2h	Codage des classes Jeu et ProgJeu qui permettent de créer une partie de jeu.
1h	Ecriture de ce compte-rendu.

III. Difficultés rencontrées

Voici les difficultés que nous avons rencontrées :

- Comprendre l'utilisation de méthodes sur des attributs d'objets qui sont les attributs d'un autre objet.
- Nous avons passé beaucoup de temps à faire nos classes de test et à analyser les bugs de nos classes.
- Certaines méthodes pourraient être optimisées.

IV. Programmation de la classe Jeu

La classe Jeu.java possède 3 attributs :

- **mainJ** et **pioche** de type *Paquet* qui correspondent respectivement au paquet de cartes dans la main du joueur et au paquet de cartes dont les cartes peuvent être rajoutées à **mainJ**
- **frise** de type *Frise* qui correspond à la frise sur laquelle sont placées les cartes de la main du joueur.

Elle possède également un constructeur prenant en paramètres un entier **tailleMain** correspondant au nombre de cartes de la main du joueur, et une chaine de caractère **fichier** correspondant au chemin relatif menant au fichier qui contient l'ensemble des cartes du jeu. Ce constructeur crée un nouvel objet de type Jeu avec un attribut **frise** contenant une frise vide, un attribut **pioche** avec les cartes contenues dans fichier, et un attribut **mainJ** contenant **tailleMain** cartes piochées au hasard dans **pioche**.

La classe possède enfin 2 méthodes :

- La méthode tour ne prend aucun paramètre, ne renvoie aucun résultat, et lance un tour de jeu. On affiche la frise et la main du joueur à l'écran, puis on demande au joueur de choisir une carte qu'il va placer dans la frise. Si la frise est vide, on rajoute directement la carte à la frise. Sinon, on demande après quelle carte de la frise il souhaite placer sa carte. On affiche alors entre quelles cartes de la frise le joueur veut placer sa carte.
  Si la date de la carte de la main du joueur est comprise entre la date de la carte sélectionnée et la date de la carte suivante, on ajoute la carte du joueur sur la frise après la carte sélectionnée, puis on supprime la carte de la main du joueur. Dans le cas contraire, on supprime directement la carte de la main du joueur et on rajoute une carte à mainJ tirée au hasard dans la pioche.
  Cette méthode est exécutée par ProgJeu.java tant que la partie n'est pas terminée.
- La méthode resultatPartie ne prend aucun paramètre et renvoie un entier correspondant à l'état de la partie. Si la main du joueur ne contient plus de cartes, on affiche dans le terminal que le joueur a gagné et on retourne 1, ce qui met fin à l'exécution de ProgJeu.java. Sinon, si la pioche est vide, on affiche que je joueur a perdu puis on retourne 2, ce qui met fin à l'exécution de ProgJeu.java. Sinon, on retourne 0 et ProgJeu.java exécute un nouveau tour.