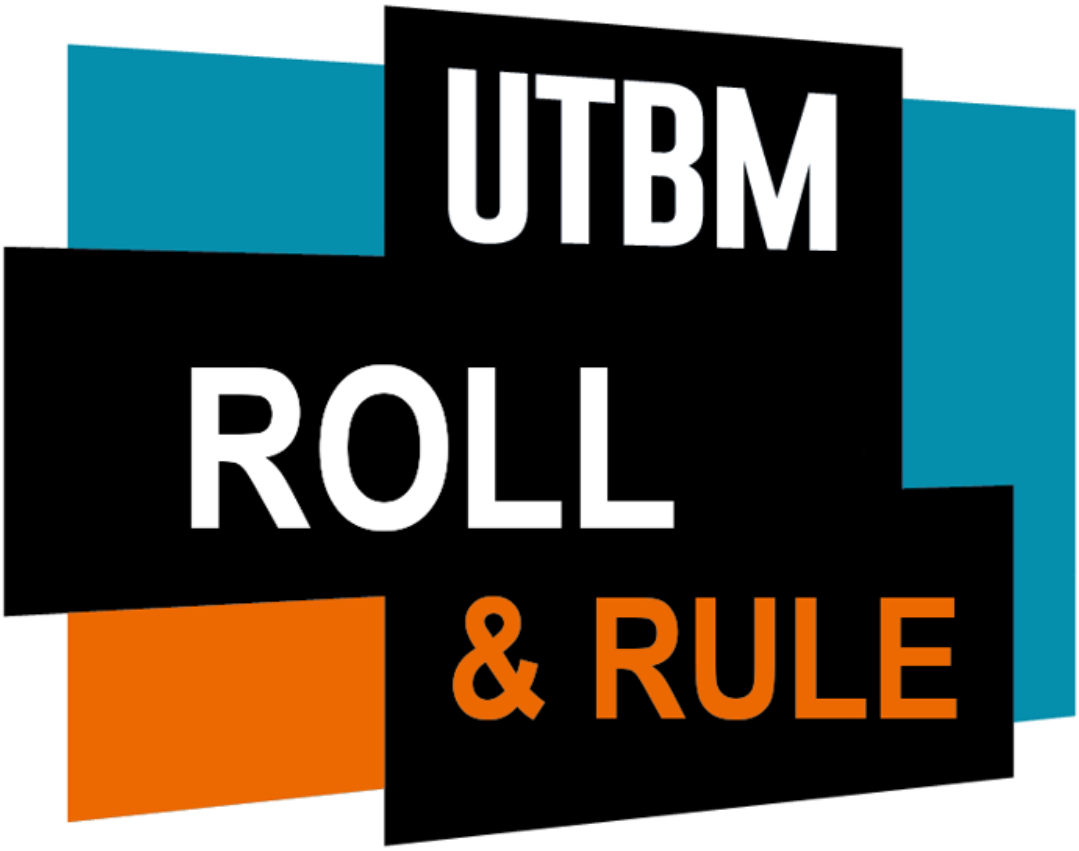
Rapport de conception



AP4B – UTBM : Roll & Rule

Réalisé par

Alexis CORREARD

Titouan ESCANDE

Simon MENICOT

Esteban GOMEZ

Table des matières

1. Introduction

Le cours d'AP4A/B nous a permis de découvrir la Programmation Orientée Objet (POO) à travers le langage C++ dans un premier temps, puis à travers le langage JAVA. Il nous a également fait découvrir le langage de modélisation UML ainsi que la méthodologie dans le développement d'un programme informatique. Afin de mettre en pratique les connaissances acquises, un projet à réaliser en équipe nous a été donné. Le but du projet est de créer une version numérique de l'un des jeux de société proposé. Il est également nécessaire de modifier le jeu pour y intégrer des éléments liés à l'UTBM. Nous avons décidé de choisir le jeu Troyes Dice. Le principe original du jeu est de recréer la cité de Troyes en construisant des bâtiments et en accueillant des habitants appartenant à différentes catégories sociales, pour marquer le plus de points de victoire possible en fin de partie.

2. Analyse du jeu

2.1 Jeu initial

L'objectif du jeu est de réunir le plus de points de victoire à la fin des 8 jours de jeu. Il y a deux moyens d'obtenir des points de victoire : avec des bâtiments et avec des habitants. A chaque tour, chaque jour étant composé de 2 tours, le joueur doit construire un bâtiment. Il y a 2 types de bâtiments : les bâtiments de fonction permettent de gagner des habitants et les bâtiments de prestige permettent d'effectuer des actions spéciales améliorant la ville. Celle-ci est ainsi composée de trois quartiers : le quartier noble, possédant des palais des comtes (fonction) et des forteresses permettant de protéger des parties de la ville (prestige), le quartier civil, avec des hôtels de ville (fonction) et des halles pour gagner des habitants ou des ressources (prestige), et le quartier religieux, comprenant des évêchés (fonction) et des cathédrales qui augmentent la valeur en points de victoire des bâtiments (prestige). Cependant, le joueur ne peut pas construire n'importe quel bâtiment. Ceux qu'il peut construire sont indiqués par des dés, qui sont au nombre de 4. Trois d'entre eux prennent à chaque tour une couleur différente qui désigne un quartier, et coûtent plus ou moins de ressources suivant la valeur du dé. Une fois un des dés choisis, sa couleur et sa valeur peuvent être modifiées en utilisant également des ressources. D'ailleurs, le joueur peut aussi choisir de gagner des ressources plutôt que de construire un bâtiment. Le dernier dé est de couleur noire. C'est un malus, il ne peut pas être choisi par le joueur. A partir du jour 3, il a pour effet de détruire une zone de la ville, si celle-ci n'est pas protégée par une forteresse, empêchant alors toute nouvelle construction jusqu'à la fin de la partie. Les bâtiments déjà construits ne sont pas détruits.

2.2 Modifications apportées au jeu initial

Pour adapter ce jeu à l'environnement UTBM, nous avons décidé de remplacer les termes liés à la ville de Troyes et au Moyen-Âge par d'autres liés à l'univers de l'UTBM. Ainsi, les tours de jeu représentent les 8 années, divisées chacune en 2 semestres, pour aller du post-bac au doctorat. Les nobles deviennent les personnels, avec comme élément de prestige des contrats de subvention et comme élément de fonction les pôles administratifs, les civils se transforment en étudiants, avec la constitution d'une association (prestige) et des résidences étudiantes (fonction), et les religieux deviennent les enseignants-chercheurs, qui publient des thèses (prestige) et travaillent dans des laboratoires de recherche (fonction). De ce fait, les ressources associées aux personnels sont des fonds, celles des étudiants sont de la motivation et celles des enseignants-chercheurs de la connaissance. Le dé noir crée ici des coupures de budget, protégées donc par les contrats de subvention.

Les règles du jeu adaptées sont fournis en pièce jointe.

3. Choix de conception

Afin de représenter au mieux nos choix de conception, nous avons choisi de concevoir différents diagrammes UML.

3.1 Diagramme des cas d'utilisation

Nous avons dans un premier temps décidé de réaliser un diagramme des cas d'utilisation, pour nous permettre de mieux comprendre ce que l'utilisateur doit faire dans notre jeu.

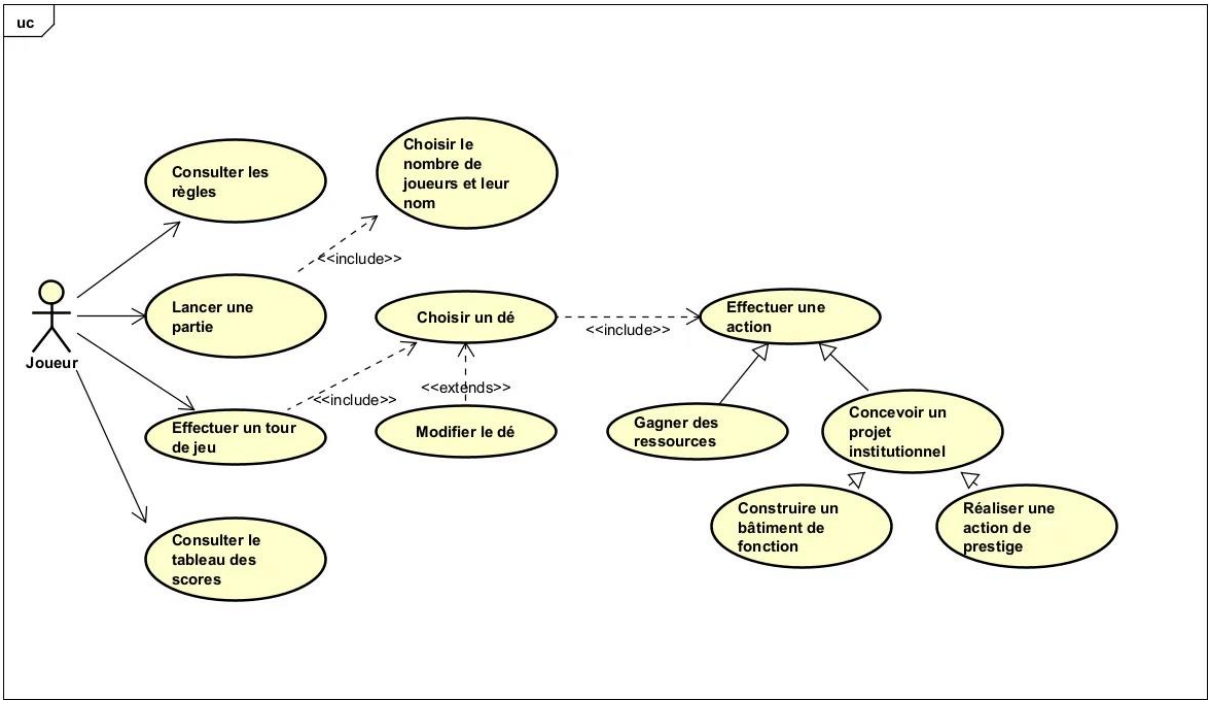


Figure 1: Diagramme des cas d'utilisation

L'utilisateur peut donc réaliser 4 actions principales :

- Consulter les règles lui permet de demander au programme d'afficher les règles du jeu
- Lancer une partie permet de lancer le programme d'initialisation de la partie du programme
- Effectuer un tour de jeu permet au joueur, dans le cadre d'une partie, de jouer son tour de jeu
- Consulter le tableau des scores permet d'afficher les scores précédemment réalisés au cours d'autres parties du jeu

Quand le joueur lance la partie, il **doit** (<<include>>) aussi choisir le nombre de joueurs inclus dans la partie et leur noms respectifs. Une feuille de jeu vierge sera alors attribuée à chaque joueur.

L'action d'effectuer un tour de jeu, quant à elle, entraîne plusieurs autres actions : L'utilisateur **doit** (<<include>>) choisir un dé de la roue, et il **peut** (<<extends>>), s'il le souhaite, le modifier. Il **doit** (<<include>>) ensuite effectuer une action : Il peut décider de gagner des ressources, **ou** de concevoir un projet (représenté à l'aide de l'héritage). Dans le cas où il choisit de concevoir un projet, il a le choix entre construire un bâtiment de fonction, ou réaliser une action de prestige (représenté à l'aide de l'héritage).

3.2 Diagramme de classe

Vu que nous faisons de la POO, il nous parut indispensable de faire un diagramme de classe, afin d'identifier les différentes classes qu'il nous faudra implémenter.

Le diagramme de classe de notre projet est situé dans l'annexe, ainsi qu'en pièce jointe de ce rapport.

Ainsi, nous avons le plateau de jeu qui est notre classe principale, qui rassemble les différents éléments du jeu. Il est composé de la roue des semestres, avec les salles et les dés qui permettent aux joueurs de construire les bâtiments. Il possède aussi les feuilles des joueurs, elles-mêmes composées des trois secteurs, qui possèdent toutes les informations liées au joueur et aux actions qu'il a faites.

3.3 Diagramme d'objet

Le diagramme d'objet sert à visualiser les instances des classes que nous utilisons dans le jeu. On repère la feuille de jeu de chaque joueur, ainsi que les 3 instances de secteur pour chaque feuille de chaque joueur. En parallèle, on voit aussi les 9 instances de Salle qui sont communes à tous les joueurs, ainsi que les 4 instances de dés, communs à tous également.

Le diagramme d'objet de notre projet est situé dans l'annexe, ainsi qu'en pièce jointe de ce rapport.

Sur le diagramme, certaines variables ont des valeurs assignées. Par exemple sur l'objet Personnels de la feuille 1, on voit que batFonction et actPrestige contiennent des 0 et des 1, les 1 signifiant l'existence du bâtiment ou de l'action et les 0 leur absence. Il existe d'autres variables avec des valeurs spécifiques, comme par exemple les dés qui possèdent une variable isTransparent qui vaut 1 dans le cas ou le dé et transparent, et 0 si le dé est noir. La valeur de certaine variable est amenée à être modifiée pendant l'exécution du programme.

Initialement, les feuilles des joueurs sont identiques. Les joueurs utilisent les mêmes dés, qui sont lancés pour tous (la roue des semestres étant commune à tout le monde). Les feuilles des joueurs sont ensuite amenées à être distingués par les choix des joueurs qui peuvent être différents.

En résumé, en se plaçant du point de vue d'un joueur, il y a 2 types d'instances, les instances communes comme les dés et les salles, et les instances spécifiques à chaque joueur (leur feuille avec leurs valeurs).

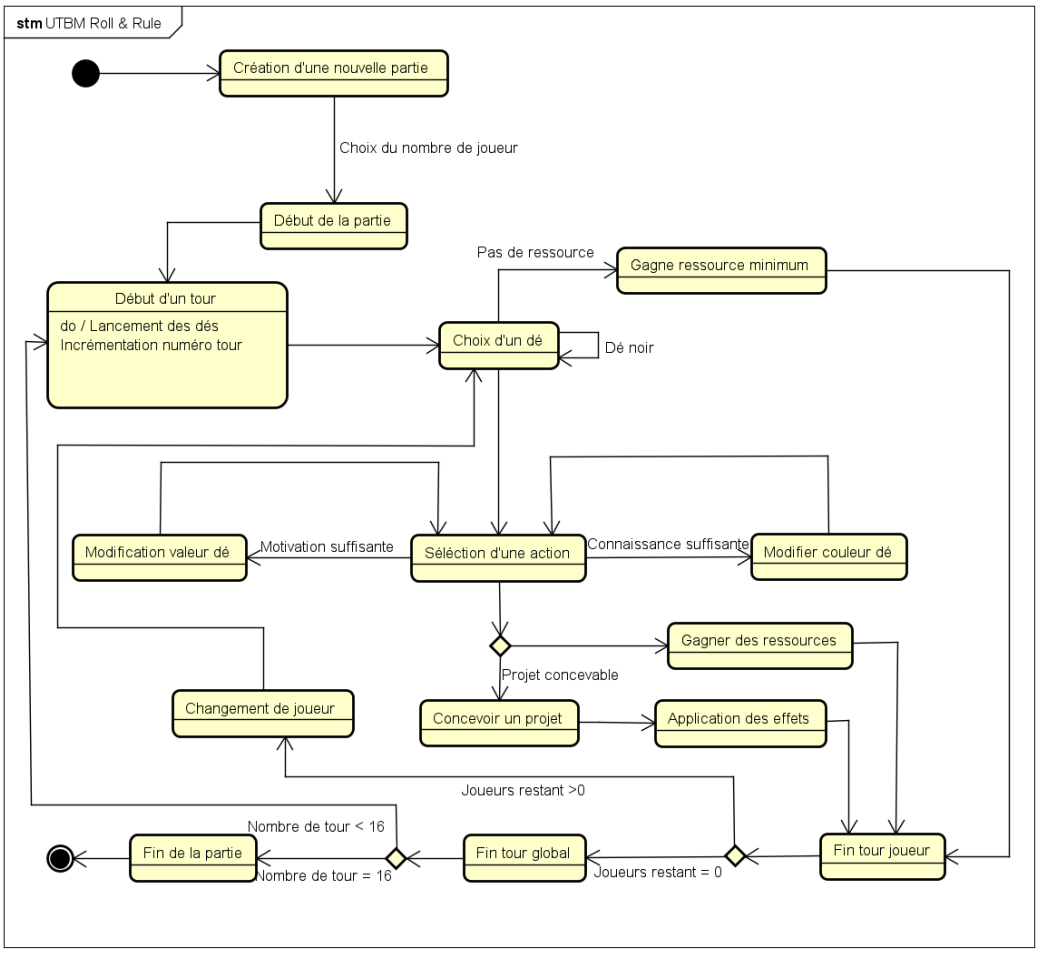
3.4 Diagramme d'état transition

Le diagramme d'état transition permet de facilement comprendre le déroulement du jeu, du point de vue du système. On repère facilement la boucle de jeu, où à chaque tour, le joueur choisit un dé, puis il a plusieurs choix :

- Modifier la valeur et/ou la couleur du dé. Le joueur peut réaliser plusieurs modifications du dé sélectionné. Le choix des différentes actions est ensuite reproposé au joueur.
- Concevoir un projet (Construction d'un bâtiment de fonction ou réalisation d'une action de prestige)
- Gagner des ressources

Dans le cas où le joueur ne peut choisir aucune de ces actions (ressources insuffisantes), il gagne des ressources minimales.

Après cela le tour du joueur se termine et c'est au joueur suivant de jouer. Si tous les joueurs ont joué, on passe au tour de jeu suivant en incrémentant le compteur de tour.



3.5 Diagramme de séquence

Le diagramme de séquence explique en détail le déroulement d'une partie avec les différentes actions possibles durant un tour de jeu. Il permet également de visualiser les différents échanges entre le joueur, le système, et les différents objets nécessaires au fonctionnement du jeu.

Le diagramme de séquence de notre projet est situé dans l'annexe, ainsi qu'en pièce jointe de ce rapport.

Le diagramme de séquence commence à la demande de création de la partie par le joueur. Plusieurs initialisations sont alors faites pour instancier les différents objets.

Le jeu entre ensuite dans une boucle globale qui englobe les 16 tours de jeu (loop [n<=16]). Pour chaque tour de jeu, le système lance les dés et détruit la colonne correspondant à la valeur du dé noir sur les feuilles des joueurs, dans le cas où on est au moins au 3ème tour. Il positionne les dés sur la roue, et récupère les différentes informations nécessaires par la suite.

Il y a ensuite une vérification (alt) : si le dé noir est en première position sur la roue, et que le joueur n'a aucune ressource, alors il ne peut pas sélectionner de dé et récupère des ressources. Dans le cas contraire, il peut choisir au moins un dé (loop [dé = noir OU ressources insuffisantes]). Dans ce cas-là, on ne rentre pas dans les blocs suivants, et on passe directement à la fin du tour.

Une fois le dé sélectionné (si le joueur a pu en sélectionner un), il y a ensuite une autre boucle qui permet au joueur de modifier celui-ci. Le joueur sort de cette boucle en indiquant qu'il ne veut pas/plus modifier le dé sélectionné.

Le système demande ensuite quelle action il souhaite réaliser avec le dé sélectionné (modifié ou non). Dans le cas d'une récupération de ressources, les ressources récupérées sont directement transmises par le système sur sa feuille de joueur. Dans le cas d'une conception d'un projet, le joueur peut choisir quel type de projet concevoir. Tant que sa réponse est incorrecte (projet déjà conçu, ou projet bloqué par une coupure de budget), il lui est redemandé quelle action il souhaite réaliser.

Ensuite le tour du jouer est terminé. Si ce n'était pas le 16ème tour, o recommence, sinon le système calcule le score final et met fin au jeu après avoir transmis le score au joueur.

4. Organisation du projet

4.1 Outils de travail

Nous avons utilisé différents outils pour nous coordonner : un dépôt GitHub pour mettre en commun les différents fichiers, une page Notion pour nous organiser et noter nos idées et une conversation pour communiquer entre nous.

Ci-dessous le lien vers la page github, avec notamment : les règles du jeu adaptées à la version UTBM : Roll & Rule, une feuille de joueur également adaptée à notre version, et les fichiers des diagrammes UML pour pouvoir les visualiser plus facilement.

https://github.com/SimonMenicot/ProjetAP4B

4.2 Organisation du groupe

Nous avons pu travailler sur le projet à la fois pendant les séances de TP, mais aussi lors de réunions organisées de notre côté. Nous avons ainsi pu nous répartir les différentes tâches du projet au fil de son avancement.

Voici ce sur quoi chaque membre du groupe a pu travailler :

- Diagrammes UML (correction, vérification) : Titouan, Simon, Alexis, Esteban
    - Diagramme des cas d'utilisation : Alexis
    - Diagramme de classe : Simon
    - Diagramme d'état transition : Titouan
    - Diagramme d'objet : Alexis
    - Diagramme de séquence : Alexis
- Rapport de conception : Titouan, Simon, Alexis, Esteban
- Règles du jeu adaptées : Simon, Alexis
- Coordination et organisation du projet : Alexis
- Implémentation des classes en java (model) : Alexis, Esteban
- Logique du jeu (controller) : Titouan, Simon
- Interface console : Titouan
- Interface graphique (visuel): Alexis
- Interface graphique (interaction) : Simon
- Vidéo : Esteban

5. Conclusion

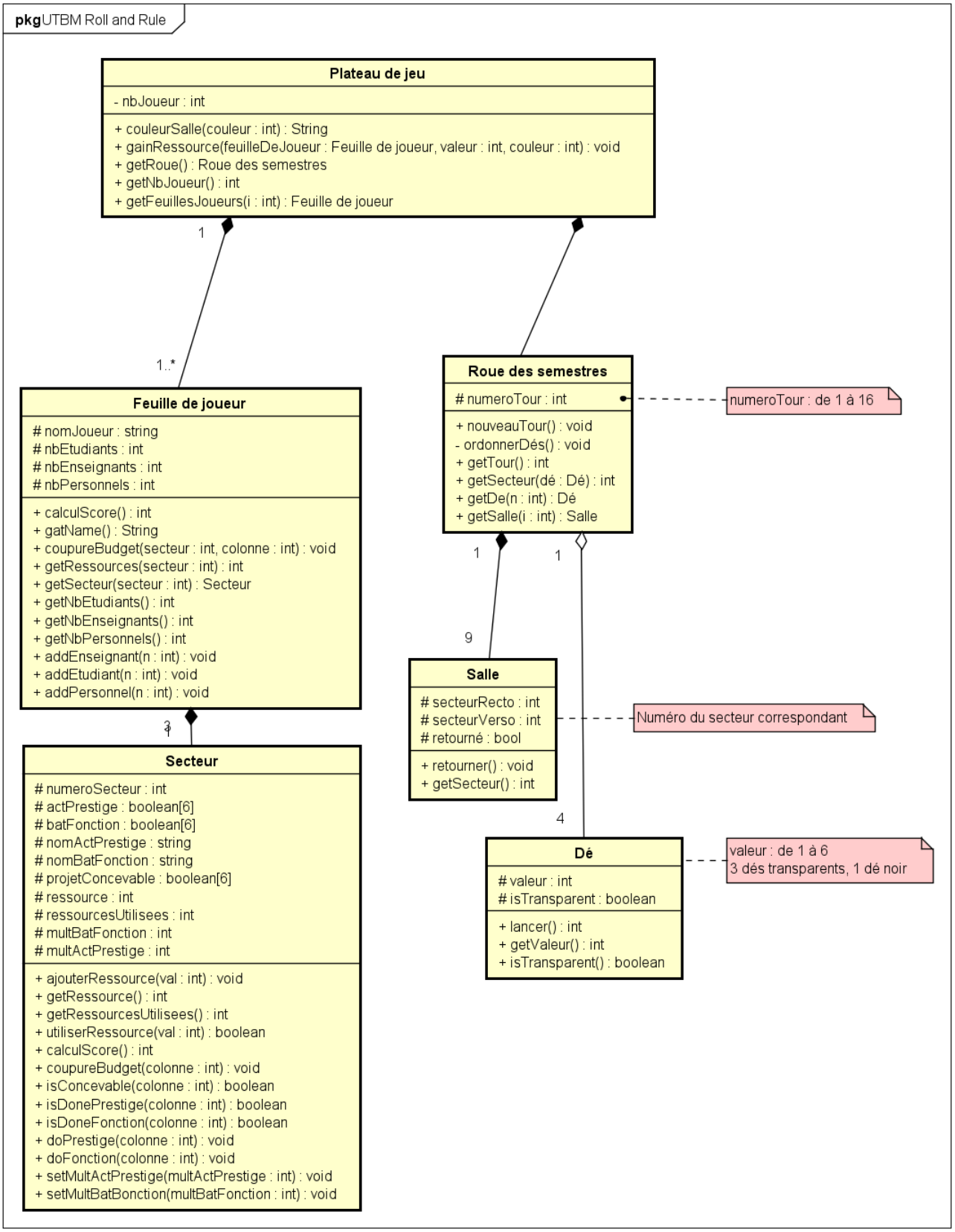
Les choix de conception que nous avons fait devraient nous permettre d'avoir un programme qui fonctionne après avoir développé l'ensemble des fonctionnalités décrites dans nos différent diagramme UML.

Nous avons fait le choix pour l'instant d'implémenter uniquement le jeu de base avec les fonctionnalités nécessaires pour que le jeu soit fonctionnel. Par la suite, en fonction du temps qu'il nous restera pour travailler sur le projet, nous pourrons ainsi encore améliorer notre projet en rajoutant diverses fonctionnalités comme l'extension du jeu, ou le tableau d'honneur.

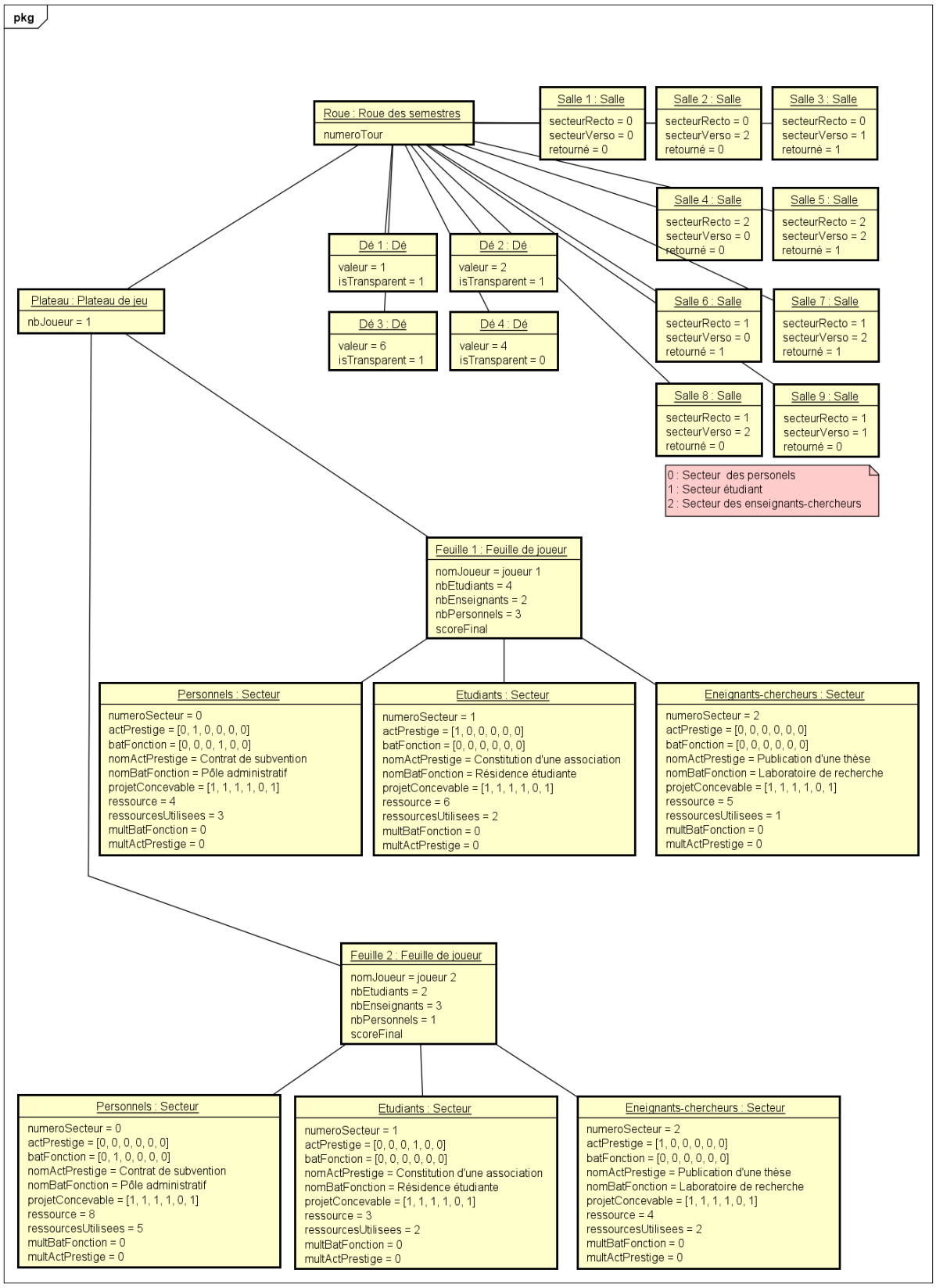
Travailler en premier sur la conception du projet, à travers les diagrammes UML, nous a permis de structurer correctement celui-ci, pour pouvoir ensuite travailler efficacement dessus.

Annexes

Annexe 1 : Diagramme de classe (cf fichier Diagramme de classe.asta)



Annexe 2 : Diagramme d'objets (cf fichier Diagramme d'objet.asta)



Annexe 3 : Diagramme de séquence (cf fichier Diagramme de séquence.asta)