Celine Ozkaya
Remy Théolier
Lise Leclef
Raphaël Chauvier
Sacha Malaterre

LO21
Projet Splendor Duel
-
Premier Compte Rendu

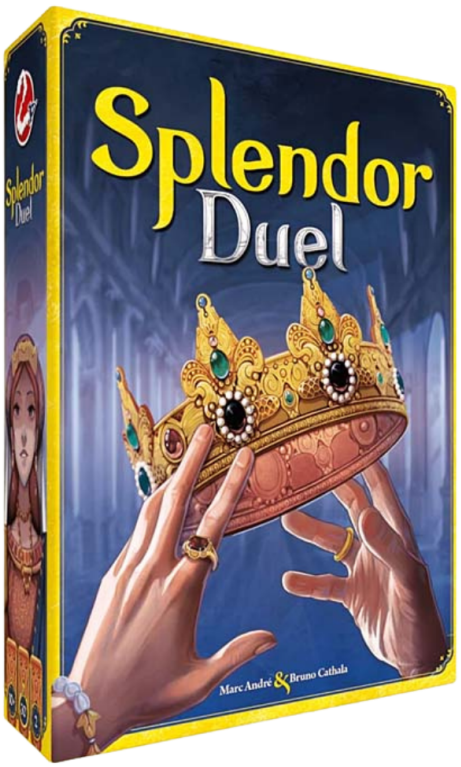


Table des matières :

I - Présentation du projet

Dans le cadre de l'UV LO21, notre mission consiste à créer en C++ le jeu Splendor Duel, avec une interface réalisée à l'aide de Qt. Tout au long du développement, nous devrons fournir trois rapports d'avancement, puis un rapport final accompagné d'une vidéo explicative du jeu ainsi que le code source du projet.

II - Premières approches pratiques

Au cours des premières semaines, notre principal objectif était de prendre connaissance du jeu lui-même et de commencer à réfléchir à la manière dont nous pourrions organiser les différentes parties du code. Pour ce faire, nous avons obtenu une copie du jeu Splendor Duel et avons consacré environ 2,5 h à découvrir ses mécanismes.

Le jeu se caractérise par une grande densité d'éléments, comprenant notamment :

→ 67 cartes Joaillerie réparties en 3 niveaux de rareté (·, ··, ···)

→ 1 tuile Victoire

→ 3 figurines Privilège

→ 25 jetons, comprenant 4 jetons de chacune des cinq couleurs de gemmes (bleu, blanc, vert, noir et rouge), 2 jetons Perles et 3 jetons Or

→ 1 plateau de jeu

→ 4 cartes Royales

Cependant, les règles du jeu restent relativement simples. On peut généralement diviser un tour de jeu en deux catégories d'actions : les actions optionnelles et les actions obligatoires. Pour vous donner une meilleure compréhension des possibilités d'action et de leurs conséquences, voici un tableau récapitulatif :

Splendor Duel

		TOUR JOUEUR 1 ACTION OPTIONELLE	
	MISE EN PLACE	**ACTION OPTIONNELLE 1 UTILISER UN PRIVILÈGE**	**ACTION OPTIONNELLE 2 REMPLIR LE PLATEAU DE JEU**
CONTRAINTES			
JOUEUR 1	• Premier joueur	• -1 privilège • +1 Gemme ou perle (PAS OR)	
JOUEUR 2	• Deuxième joueur • +1 Privilège		• +1 privilège
PRIVILÈGES	• 3 privilèges disponibles	• +1 privilège dispo	
PLATEAU	• Plateau : Répartir aléatoirement 25 jetons	• -1 gemme/perle	• Tant que le sac n'est pas vide et que le plateau n'est pas plein : prendre un jeton du sac et le mettre sur le plateau
CARTES ROYALES	• Cartes royales : 4 cartes disponibles		
CARTES JOAILLERIE	• 5 niveau 1 • 4 niveau 2 • 3 niveau 3		

1 SEULE SUR 3 ACTION OBLIGEÉ PAR TOUR

ACTION OBLIGATOIRE 1 PRENDRE JUSQU'À 3 JETONS GEMME ET/OU PERLE.	**ACTION OBLIGATOIRE 2 PRENDRE 1 JETON OR POUR RÉSERVER 1 CARTE JOAILLERIE**	**ACTION OBLIGATOIRE 3 ACHETER 1 CARTE JOAILLERIE**	**FIN DU TOUR**	**CONDITIONS DE VICTOIRE**
	AU MOINS 1 OR SUR LE PLATEAU JOUEUR A MOINS DE 3 CARTES RÉSERVÉES			
• Si ligne/colonne/diagonale de 3 jetons (HORS OR) aligné : possible de prendre le 3 • Possible d'en prendre uniquement 2 ou 1 (HORS OR) • +1 à +3 gemmes/perles	• +1 Or • +1 carte dans réserve	• jeton : -prix de la carte • Capacité de la carte	• Check <10 jetons sinon : jeter de jetons	• +=20 points de prestiges sur toutes ses cartes • +=10 couronnes sur ses cartes joaillerie • +=10 points de bonus dans ses cartes d'une meme couleur
• Si les 3 gemmes prises sont identiques ou si 2 perles sont récupérées : +1 privilège				
• -1 à -3 gemmes/perles alignées		• sac : +prix de la carte		
		• Si j=3 couronnes : -1 carte royale • Si j=6 couronnes: -1 carte royale		
	• Carte prise dans pyramide : -1 carte, +1 nouvelle carte • Peu importe où la carte est prise : -1 carte pioche	• Si la carte n'était pas reservée : -1 carte		

III - Décompositions des tâches

A. Tâches effectuées :

1. Compréhension et analyse des règles du jeu
2. Première réflexion sur les classes et méthodes à implémenter
3. Première conception de l'UML en spécifiant les classes et les liens

B. Tâches restantes :

1. Révision et amélioration UML (08/10/2023 - 19/10/2023)
    - Réviser et améliorer le diagramme UML en tenant compte des commentaires de la première version
2. Architecture approfondie (08/10/2023 - 19/10/2023)
    - Réfléchir à l'architecture globale du logiciel :

      - Comment gérer les jetons qui sont posés sur le Plateau (coordonnées, etc), et en possession des joueurs, ainsi que gérer le nombre limite.

      - Comment prendre en compte les règles spécifiques aux privilèges (par exemple l'utilisation de privilèges, les limitations à 3 privilèges dans tout le jeu)

      - Etc…

3. Implémentation initiale (20/10/2023 - 02/11/2023)
    - Commencer/continuer à coder les classes et les fonctionnalités de base du jeu Splendor Duel en C++
4. Finalisation de la première implémentation (03/11/2023 - 16/11/2023)
    - Travailler sur les fonctions et les mécanismes du jeu qui ne sont pas directement liés aux classes, améliorer les classes existantes avec les nouvelles connaissances acquises
    - Voir parmi les design patterns du cours si certains peuvent nous aider à résoudre certaines de nos difficultés à créer notre architecture
5. Conception des images (03/11/2023 - 16/11/2023)
    - Créer les graphismes et les images nécessaires pour le plateau, les cartes et les jetons du jeu
6. Conception interface graphique avec Qt (17/11/2023 - 23/11/2023)
    - Créer une conception graphique pour l'interface utilisateur du jeu Splendor Duel
7. Implémentation interface (24/11/2023 - 15/12/2023)

- Intégrer l'interface utilisateur avec le code du jeu et s'assurer que tout fonctionne de manière fluide avec Qt, en C++

8. Finalisation (07/12/2023 - 15/12/2023)
    - Finalisation du projet : batteries de tests en plus des tests existants, étude finale des fuites de mémoire, etc.

IV - Diagramme de Gantt

Le diagramme et ses données sont disponibles à la fin du document.

V - Résumé prévisionnel des classes et méthodes

Pour pouvoir programmer ce jeu, nous avons pour l'instant choisi d'établir les classes suivantes :

La classe SplendorDuel représente une partie du jeu.
Les méthodes de la classe seront les suivantes :

- void lancerPartie() : Commence le jeu
- void finirPartie() : Finit le jeu

La classe Joueur contient les informations sur un joueur, elle a donc comme attributs nom (le nom du joueur), privilège (le nombre de privilèges que le joueur possède) et jetons (les jetons qu'il a).
Les méthodes de la classe seront les suivantes :

- void ajouterJeton(Jeton jeton) : Rajoute un jeton au joueur
- void retirerJeton(Jeton jeton) : Retire un jeton au joueur
- void reserverCarte(Carte carte) : Rajoute une carte à la réserve du joueur
- void acheterCarte(Carte carte) : Achète une carte mise en jeu
- void depenserJeton(Jeton jeton) : Retire un jeton au joueur
- void utiliserPrivilege(Privilege privilege) : Utilise un privilège pour faire une action (prendre un jeton)
- int getPrestiges() : Retourne le nombre de points de prestige (points sur une rangée)
- int getPoints() : Retourne le nombre de points
- int getCouronnes() : Retourne le nombre de couronnes

La classe Plateau correspond au plateau de jeu, elle est donc constituée d'une grille de jetons représentant le plateau.
Les méthodes de la classe seront les suivantes :

- void realign() : Redistribue les jetons sur le plateau
- void ajouterJeton(Jeton jeton) : Rajoute un jeton au plateau
- void retirerJeton(Jeton jeton) : Retire un jeton du plateau

La classe LigneDeCarte a comme attribut niveau, qui représente le niveau des cartes joaillerie présentes dans la ligne.

La classe SacJeton est constituée de plusieurs jetons, qui sont mis dans l'attribut jetons qui est une matrice.
Les méthodes de la classe seront les suivantes :

- ➢ void ajouterJeton(Jeton jeton) : Rajoute un jeton au sac
- ➢ void retirerJeton(Jeton jeton) : Retire un jeton du sac

La classe Cartejoaillerie permet de représenter les cartes joaillerie du jeu, elle a donc comme attribut niveau (son niveau), cout (son prix), point_de_prestige (les points de prestiges), capacite (sa capacité), couronne (le nombre de couronnes) et Bonus (le bonus qu'elle apporte).
Les méthodes de la classe seront les suivantes :

- ➢ string getNiveau() : renvoie le niveau de la carte
- ➢ int getCout() : renvoie le prix de la carte
- ➢ int getPrestige() : renvoie les points de prestige de la carte
- ➢ int getCouronne() : renvoie les couronnes de la carte
- ➢ Capacite getCapacite() : renvoie la capacité de la carte
- ➢ jeton getBonus() : renvoie le bonus de la carte

La classe CarteRoyale représente les cartes royales du jeu, elle a comme attributs point_de_prestige et capacite.
Les méthodes de la classe seront les suivantes :

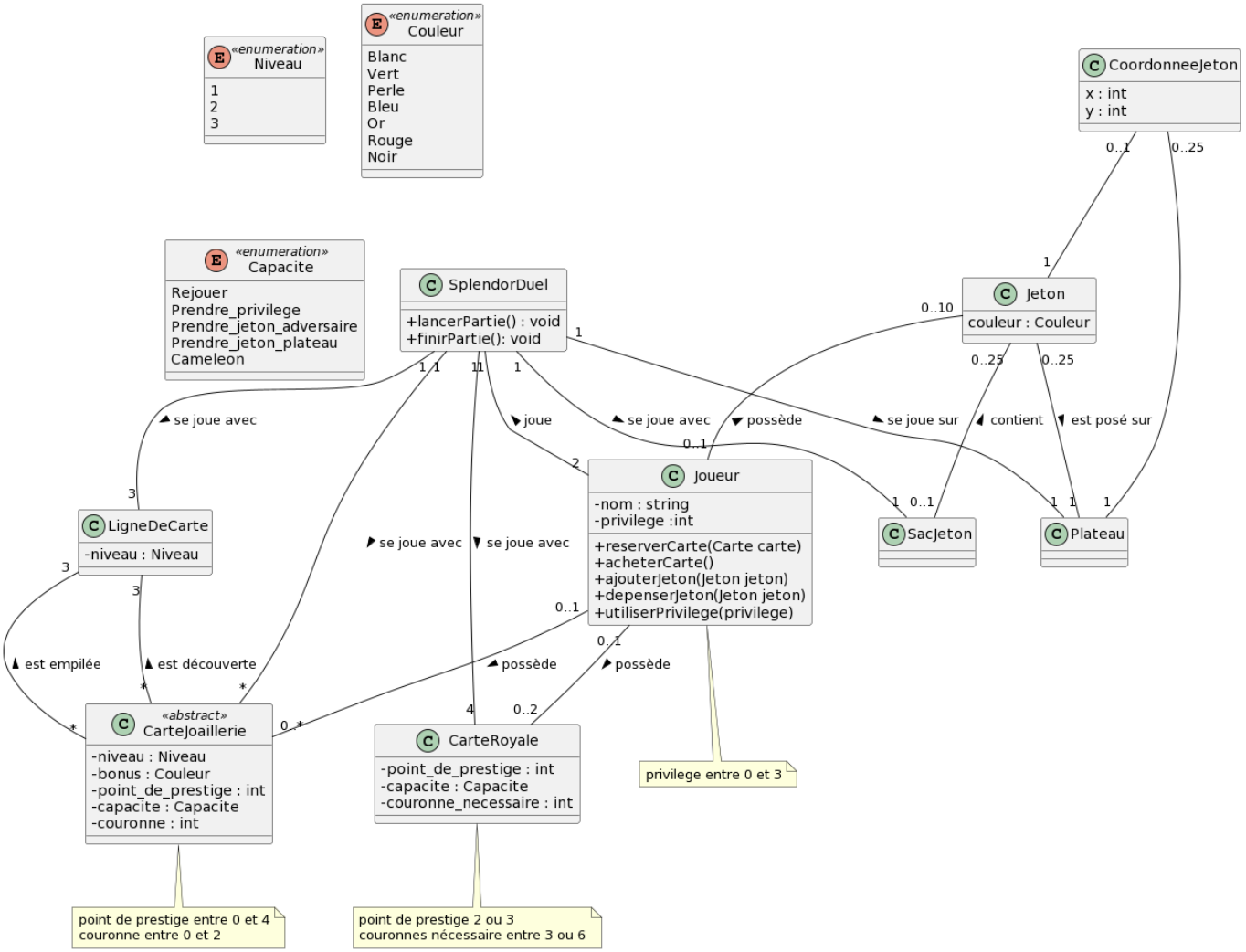
- ➢ int getPrestige() : renvoie le nombre de points de prestige
- ➢ capacite getCapacite() : renvoie la capacité de la carte

La classe Jeton représente un jeton du jeu, elle a comme attribut nom qui est donc la couleur/type du jeton.
Les méthodes de la classe seront les suivantes :

- ➢ void setNom(string nom) : définit le nom du jeton (modifiable avec le système d'identifiant unique)
- ➢ string getNom() : renvoie le nom du jeton

VI - Diagramme UML

Les constructeurs, destructeurs, setter et getter des classes ne sont pas représentés sur l'UML afin de ne pas surcharger le graphe.



Projet LO21

7 oct. 2023

Dates du projet	14 sept. 2023 - 16 déc. 2023
Avancée	32%
Tâches	11
Ressources	5

Tâches

Nom	Date de début	Date de fin
Étude du Sujet	14/09/2023	07/10/2023
Analyser en profondeur les exigences du jeu Splendor Duel, les règles du jeu, et les fonctionnalités nécessaires.		
Shéma prévisionnel des classes et méthodes	21/09/2023	07/10/2023
Documenter les objets méthodes essentiels du projet, les fonctionnalités clés.		
Conception Initiale UML	28/09/2023	07/10/2023
Créer un diagramme UML initial pour représenter la structure et les relations des classes.		
UML 2	08/10/2023	19/10/2023
Réviser et améliorer le diagramme UML en tenant compte des commentaires de la première version.		
Architecture approfondie	08/10/2023	19/10/2023
Réfléchir à l'architecture globale du logiciel.		
Implémentation initial	20/10/2023	02/11/2023
Commencer/continuer à coder les classes et les fonctionnalités de base du jeu Splendor Duel en C++.		
Finalisation de la première implémentation	03/11/2023	16/11/2023
Travailler sur les fonctions et les mécanismes du jeu qui ne sont pas directement liés aux classes, améliorer les classes existantes avec les nouvelles connaissances acquises.		
Conception des images	03/11/2023	16/11/2023
Créer les graphismes et les images nécessaires pour le plateau, les cartes et les jetons du jeu.		
Conception interface graphique avec Qt	17/11/2023	23/11/2023
Créer une conception graphique pour l'interface utilisateur du jeu Splendor Duel.		
Implémentation interface	24/11/2023	15/12/2023
Intégrer l'interface utilisateur avec le code du jeu et s'assurer que tout fonctionne de manière fluide avec Qt, en C++.		
Finalisation	07/12/2023	15/12/2023
Finalisation du projet : batteries de tests en plus des tests existants, étude finale des fuites de mémoire, etc...		

Ressources

Nom	Rôle par défaut
Raphaël	Développeur
Céline	Développeur
Lise	Développeur
Sacha	Développeur
Remy	Développeur

Diagramme de Gantt

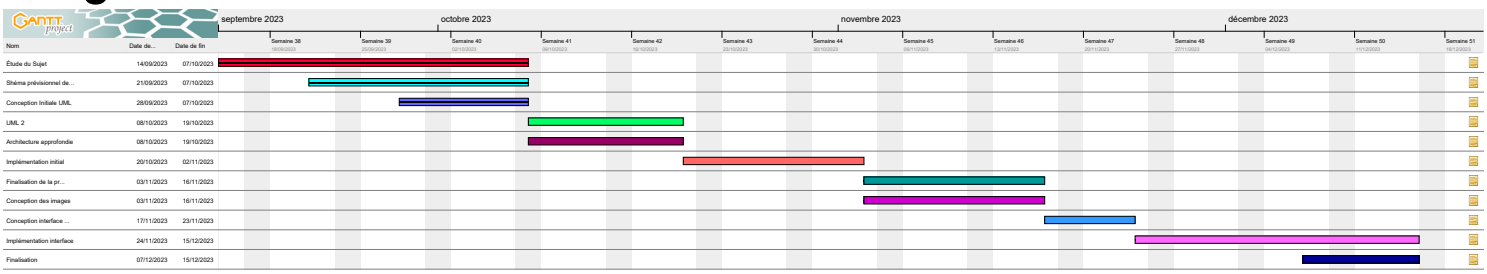


Diagramme des Ressources