

LO21 - P22 Projet Carcassonne

Rapport 3

DU Sylvain - TAVERNY Kelyan - THIBAUT Ambroise - BOUZAR Massil - OUEDRAOGO Taoufiq (Auteur responsable de ce rapport)



I - Liste des tâches effectuées

Comme mentionné dans le précédent rapport, nous nous sommes focalisés sur l'implémentation des différentes classes ainsi que leurs méthodes. De plus, nous avons aussi travaillé sur les différents modules permettant les validations et placements des tuiles et meeples. Il y a également eu un travail sur l'interface graphique.

Tâche	Responsable	Durée	Date	Reste à faire	Commentaires
Étude des Design Pattern de création	Massil	3h	28/04	/	
Étude des Design Pattern structurel	Sylvain / Taoufiq	3h	28/04	/	
Étude des Design Pattern comportementaux	Ambroise	3h15	24/04	/	
Réunion avancement Architecture suite Design Pattern	Massil / Taoufiq / Sylvain / Ambroise	2h	27/04	/	
Modification UML suite réunion	Massil / Taoufiq / Sylvain / Ambroise			/	
Réunion planification des tâches	Massil / Taoufiq / Sylvain / Ambroise / Kylian		/	/	
Arrangement Repo et organisation du code (.gitignore, premier fichier, Qt)	Ambroise	1h	01/05	/	
Préparation réunion	Ambroise	30min	01/05	/	Préparation des points à aborder pour la réunion (comment gérer les extensions ? Comment calculer et attribuer les points ?)
Préparation réunion	Massil	30min	01/05	/	Remise en cause de l'architecture sur la classe Tuile (pertinence de l'héritage et s'assurer que l'on a bien toute l'information)
réunion	Tous	1h	01/05	/	Définition des tâches de la semaine liste question à poser au

					prof pendant le TD de 02/05
Travail préparatoire developpement, test de Qt, création des branches git	Ambroise	30min	02/05	/	
Avancement Rapport 2	Massil	30min	02/05	/	
Avancement UML	Ambroise	1h	03/05	/	
Avancement UML	Ambroise	30min	04/05	/	classe Espace et modif classe tuile et contenance
Réunion Architecture UML	Massil / Taoufiq / Sylvain / Ambroise	1H15	04/05	/	Classe espace et extensions
Avancement Rapport 2	Massil	1h	06/05	/	
Implémentation du code	Taoufiq	1h	06/05	/	Classe Meeple et TypeMeeple
Implémentation du code	Taoufiq	40min	06/05	/	Classe Joueur et les énumérations(TypesTuiles et NomMeeple)
Fin UML rapport 2	Ambroise	1h30	07/05		réflexion et peaufinage
Rapport	Ambroise	45 min	07/05		MAJ UML, futures tâches
Implémentation du code	Massil	45min	08/05	/	Classe Tuile et ContenuTuile
Implémentation de code et organisation repo	Ambroise	1h	11/05	/	Class Plateau
discussion et repartition travail Qt	Ambroise / Kelyan	1h	12/05	/	
Implémentation du code	Massil	2h	12/05	/	Class Tuile et Contenu Tuile
Apprentissage XML et .ui Qt	Ambroise	1h	12/05	/	
Mise en pratique XML et définition classes Vue	Ambroise	1h30		/	
Implémentation du code et test	Massil	1h15	13/05	/	Class Tuile et Contenu Tuile
Implémentation du code	Massil	45min	13/05	/	Class Pioche
Implémentation du code	Sylvain	3h	13/05	/	Class Controller (implémentation d'un menu)
V1 VueContenuTuiles	Ambroise	2h	14/05	/	
Implémentation du code et test	Massil	4h30	14/05	/	Class Pioche
Implémentation du code	Sylvain	4h	15/05	/	Class Controller (implémentation de la

					validation placement tuile, nextTour, comptage score pour l'abbaye)
Implémentation du code et test	Massil	1h30	16/05	/	Class Pioche
Implémentation du code et test	Taoufiq	1h	16/05	/	Class Meeple
Modification UML	Taoufiq	1h	16/05	/	Réorganisation et ajout d'attributs et de méthodes pour certaines classes
Code Qt	Ambroise	2h30	16/05	/	il faut régler problème du non affichage du nom
Implémentation du code, test et réflexion	Taoufiq	3h	17/05	/	Réflexion autour des classes Meeple, Joueur et Tuile, fin d'implémentation du code pour Meeple et Joueur, réglages des erreurs
Code Qt	Ambroise	1h45	17/05	/	debug QGridLayout
Implémentation du code et test	Taoufiq	1h30	17/05	/	Implémentation de la classe Espace + test du reste des classes
Implémentation du code, test et réflexion	Taoufiq	1h30	18/05	/	Ajout de quelques méthodes pour la classe Espace et libération des espaces mémoires pour la classe Joueur
Test du controller avec les autres classes	Sylvain	3h	18/05	/	Correction des erreurs
Code QT	Ambroise	1H30	20/05	/	Fin V1 fonctionnel vueTuile
Test du controller avec les autres classes V2	Sylvain	4h	20/05	/	Corrections des erreur V2 + Changement sur la classe Plateau +PlacementTuileAuroris e fonctionnel
Implémentation du code et test	Massil	2h30	21/05	/	Class Pioche, problème sur méthode piocher pas corrigé
Implémentation du code et test	Massil	1h	21/05	/	Fin Class Pioche
Implémentation et test	Taoufiq	2h	22/05	/	Changement et test

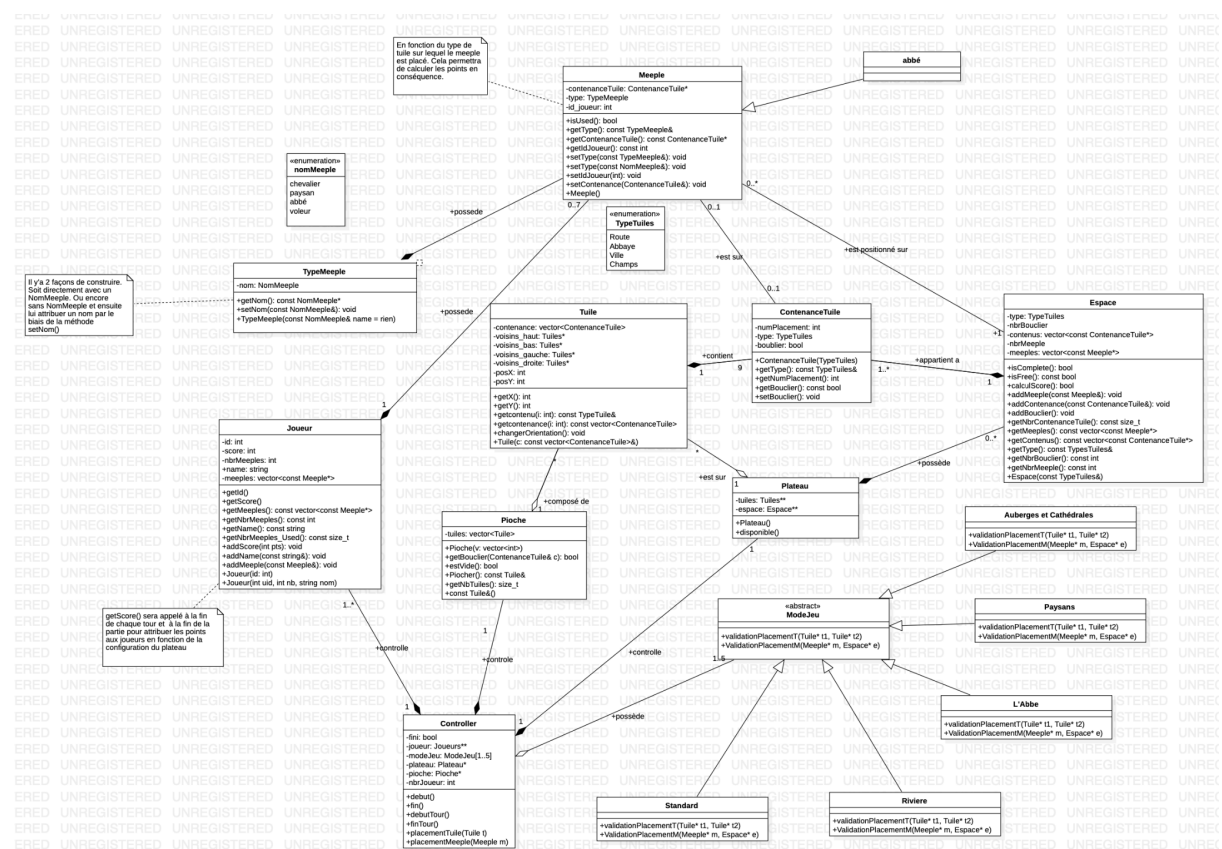
					avec merge tuile et pioche
Code QT	Ambroise	1H30	23/05	/	Avancement VuePartie : définition des différents espaces
Placement Tuile Extension	Sylvain	30 min	24/05	/	Placement Tuile dans les extensions sauf riviere
Recherche QT	Taoufiq	30min	26/05	/	Recherche pour pouvoir bien récupérer le nombre de joueurs dans la partie
Placement Tuile Rivière	Sylvain	1h40	26/05	/	PlacementTuileRiviere + test fonctionnel
Code Qt	Ambroise	3h	26/05	/	apprentissage slots, implémentation du controller dans la classe VuePartie et des methodes en consequences
Mise en commun travail Kelyan	Ambroise	1h	26/05	/	
Code Qt	Ambroise	1h45	27/05	/	resolution problème nombre de joueurs, affichage en conséquence, récupération des extensions
Réunion	Tous	1h30	27/05 et 28/05		Mise en commun des travaux réalisés et préparation du basculement de la totalité des membres sur Qt
Code Qt	Ambroise	2h	28/05		Instanciation complete (sans le plateau) de toute VuePartie. Utilisation du controller
Merge avec code de tous le monde	Ambroise	30 min	28/05		RAS
Avancement Rapport 3	Massil	45min	28/05		Rédaction III Gestion de la Tuile
Avancement VuePlateau	Sylvain/Taoufiq/Kelyan	2h	28/05	/	

Mise à jour de l'architecture : diagramme UML

Depuis le deuxième rapport, des modifications ont été apportées à notre architecture.

Des attributs et méthodes supplémentaires ont été rajoutés à la classe Espace et ContenanceTuile afin de permettre les validations de placement et placement des tuiles et des meeples.

Voici la dernière version de notre représentation en UML.



Présentation interface graphique

Une partie importante du travail réalisé depuis le dernier rapport fût de poser des bases solides de l'interface graphique en Qt. Nous avons décidé que notre jeu se décomposent en 4 fenêtre successives :

- Choix des extensions et du nombre de joueurs : Classe VueAccueil
- Renseignement du nom des joueurs : Classe VueFormNom
- Déroulement de la partie : Classe VuePartie
- Affichage des résultats de fin : Classe VueFinPartie

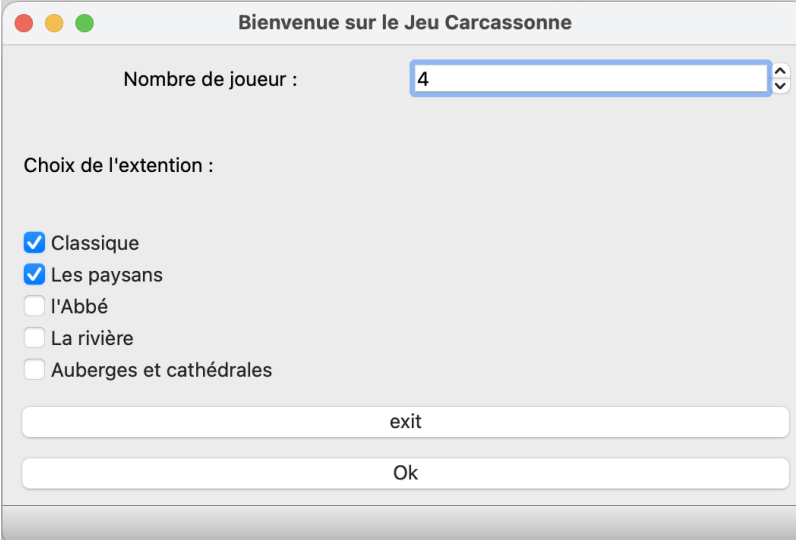
Le constructeur de VueFormNom prends en paramètre le nombre de joueurs (int) et les extensions choisies (vector<int>).

Ainsi le nombre de formulaire de VueFormNom s'adapte en fonction du nombre de joueurs.

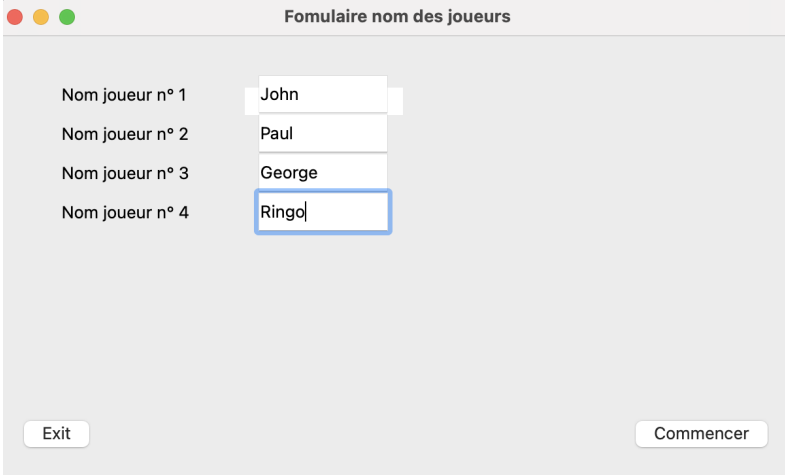
Le constructeur de VuePartie prendra en paramètre le Controller de la partie instancié grâce aux informations recueillies par VueAccueil et VueFormNom.

Pour l'instant, la fenêtre de déroulement de la partie affiche les joueurs et leurs scores ainsi que des boutons pour gérer ces scores, le joueur dont c'est le tour de jouer. La tuile que ce dernier doit placer.

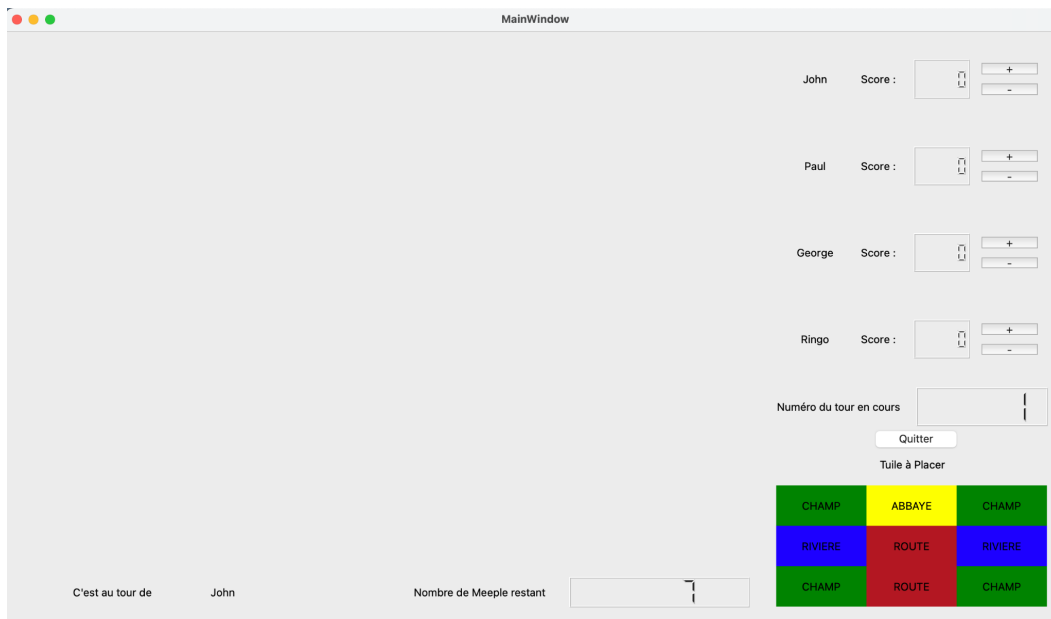
En voici un exemple :



The screenshot shows a window titled "Bienvenue sur le Jeu Carcassonne". It contains a label "Nombre de joueur :" followed by a text box with the value "4". Below this is a section titled "Choix de l'extention :" with four checkboxes: "Classique" (checked), "Les paysans" (checked), "l'Abbé" (unchecked), "La rivière" (unchecked), and "Auberges et cathédrales" (unchecked). At the bottom are two buttons: "exit" and "Ok".



The screenshot shows a window titled "Formulaire nom des joueurs". It contains four labels: "Nom joueur n° 1", "Nom joueur n° 2", "Nom joueur n° 3", and "Nom joueur n° 4". To the right of each label is a text box containing the names "John", "Paul", "George", and "Ringo" respectively. The "Ringo" text box is highlighted with a blue border. At the bottom are two buttons: "Exit" and "Commencer".



II - Liste des tâches en cours/restantes

L'ensemble des classes étant implémentées, le plus gros du travail reste le développement de l'interface avec QT.

Tâche	Responsable	Durée	Date	Pourcentage de réalisation	Importance
Réaliser la première page d'accueil Qt (choix options de la partie)	Kelyan - Ambroise		07/05		
Implémenter la classe Tuile / Contenu Tuile	Massil	4h	13/05	100%	Indispensable
Classe Joueur et les énumérations (TypesTuiles et NomMeeple)	Taoufiq	2h	08/05/2022		

Définir la classe Controller	Sylvain		08/05		
Implémenter la classe Espace					Indispensable
classe pioche	Massil	9h	22/05	100%	Indispensable
classe plateau	Ambroise				
Classe ModeJeu et classes hérité	Sylvain				
Classe Meeple et TypeMeeple	Taoufiq		07/05		
reflexion autour des structures de données (map et vector)	Tous				
interface plateau	Kelyan et Sylvain		29/05	100%	
creation differentes zone de la fenetre de Jeu	Ambroise	2h	23/05		
creation VueTuiles	Ambroise	7h45	20/05	100%	
affichage dynamique de l'espace Joueurs (avec nombre de points et incrémentation manuelle)	Ambroise	2h	23/05	100%	
Lien avec le controleur pour lancer une partie	Ambroise	3h	26/05	100%	
Finir page accueil et page affichage nom (regler probleme du nombre de joueur et affichage en conséquence) + lister les extensions choisies pour les transmettre au controleur	Ambroise	2h45	27/05	100%	
Creation et affichage VuePartie correctement en fonction des données récupérées dans VueAccueil et VueFormNom	Ambroise	2h	28/05	100%	
Merge propre de tout	Ambroise	30min	28/05	100%	
Video de présentation avec commentaires audio	tous				
Avancé et travail pour le rendu final	tous				
Gestion de l'évolution des Espaces du Plateau au fur et à mesure de la partie : Approche logique	Massil		29/05		

III - Gestion de la Tuile

Afin de contenir toute l'information permettant une bonne gestion à la fois du placement des Tuiles, mais aussi des Meeples sur celles-ci, nous avons conçu notre architecture de la manière suivante.

Chaque Tuile est composée d'un vecteur de 9 ContenuTuile réparti de la manière suivante en fonction des indices du vecteur :

0	1	2
7	8	3
6	5	4

Cela facilite le changement d'orientation puisque chaque indice i prend la valeur $i+2 \% 8$. De plus chaque Tuile possède 4 pointeurs sur chacun de ses voisins (hauts, bas gauche, droite) ainsi que des coordonnées (x,y) initialisées lors du placement sur le Plateau.

La Pioche génère les Tuiles du mode de jeu dans un vecteur de Tuile et possède une méthode piocher. Le Controller possède un pointeur de Pioche ce qui lui permet dans le déroulé de la partie d'appeler la méthode piocher au début de chaque tour.

Lorsqu'un joueur souhaite poser une Tuile à un emplacement vide(càd un voisin nullptr), la méthode ValidationPlacementTuile est alors appelée. En fonction des types de ContenuTuile et des positions relatives des tuiles concernées (placement en haut, en bas, à gauche ou à droite de la Tuile déjà posée), les tests sont effectués sur les ContenuTuile centraux de la bordure testée (càd les positions 1,3,5,7).

La méthode ValidationPlacementTuile étant polymorphique, cela nous assure que la bonne méthode sera appelée puisque chaque extension ajoute les tests sur

les ContenuTuile qui lui sont propres. Seul l'extension rivière diverge car on ne peut pas faire un demi tour immédiat

À la fin du tour d'un joueur, celui-ci peut poser un Meeple sur un ContenuTuile. Pour valider ce placement, la classe Espace expliquée dans le rapport précédent a été introduite. Le Plateau possède à la fois un vecteur de Tuile mais aussi un vecteur d'Espace. Le Controller possède un pointeur de Plateau. Cela nous permet de récupérer l'ensemble de l'information.

Un Espace regroupe les contenus de tuiles de même type adjacent. Cela permet donc de connaître l'ensemble des zones (villes, champs, etc) s'étendant sur plusieurs tuiles qui seront construites au fur et à mesure de l'évolution du plateau, c'est-à-dire à chaque fois que le placement d'une Tuile a été validé. Celle-ci contient donc des vecteurs de Meeples et de ContenuTuile. Un vecteur de Meeples a été choisi et non un pointeur, car dans le cas où plusieurs espaces d'un même type se rejoignent pour n'en former qu'un, chacun peut à l'origine contenir un Meeple, demandant alors de pouvoir stocker tous les Meeples du nouvel Espace.

Lors du placement d'un Meeple, la méthode ValidationPlacementMeeple est appelée. Pour vérifier que le placement est possible, nous avons introduit une méthode isFree() qui permet de savoir si un Espace contient déjà un Meeple (c.-à-d. que le vecteur de Meeple de l'objet Espace en question est vide) ou si l'Espace est libre. Si l'Espace est libre alors le placement est possible.

III - Cohésion de groupe et implication de chacun

Les tâches étant correctement fixées et les rôles spécifiquement attribués, cela favorise énormément la cohésion au sein de notre groupe. Ces facteurs nous permettent également de gagner en efficacité et en rapidité.

De plus, la motivation de chaque membre et les échanges réguliers que nous tenons sur le groupe Messenger mais aussi les réunions hebdomadaires que nous organisons stimulent également notre travail et suscitent encore plus de réflexion de notre part. Ce qui nous permet de poursuivre dans la logique du sujet de façon pertinente.

Certains problèmes ont été rencontrés par rapport au travail collaboratif, notamment sur des différences de nommage des fichiers, malgré l'établissement des conventions.

