BRICHET Clément METRAL Greg

LADUGUIE Alexandra PALLEAU Tom

**Rapport AP4B : Allegra**

Pour ce projet, nous avons décidé de nous orienter vers le jeu Allegra car ce projet nous semblait être le plus intéressant.

**Règles du jeu:**

Le but d’Allegra est d’obtenir le score le plus faible. Chaque joueur commence avec 4 colonnes de 3 cartes, et peu aussi jouer sur la colonne de droite du joueur à sa gauche. Le jeu commence avec toutes les cartes de tous les joueurs face cachée. A chaque tour, un joueur peut piocher une carte dans la pioche, dans la défausse, retourner une de ses cartes ou demander une carte piochée d’un autre joueur. Le jeu s’arrête un tour après qu’un joueur ait retourné toutes ses cartes. Si 3 cartes identiques sont alignées, alors elles sont retirées sans modifier l’ordre des cartes et comptent pour 0. Les cartes vont de -1 à 11.

Tout d’abord, nous avons changé le scénario du jeu pour correspondre au thème UTBM comme demandé. Les cartes seront donc remplacées par les suivantes :

-1: SEE

0: A

1: B

2: C

3: D

4: E

5: Fx

6: F

7: ABS

8: Convocation au Jury

9: TC05

10: TC06

11: Réorientation

Nous avons ensuite réalisé plusieurs scénarii pour les différentes actions possibles pour un joueur afin de bien comprendre comment organiser notre code.

**Scenario: Prendre une carte dans la défausse**

C'est le tour du joueur A. L'interface demande au joueur de faire un choix entre la pioche et la défausse. Il demande la carte de la défausse. L'interface lui demande de sélectionner la carte qu'il veut remplacer. Le joueur A sélectionne une carte. La carte du joueur est retournée si elle est face cachée et part dans la défausse. La carte de la défausse remplace celle du joueur A dans son jeu. Fin du tour du joueur A.

**Scenario: Ne rien piocher**

C'est le tour du joueur A. L'interface demande au joueur de faire un choix entre la pioche et la défausse. Le joueur choisit la pioche. L'interface retourne la première carte de la pioche. Elle demande aux joueurs si quelqu'un veut voler la carte. Les joueurs B, C, D répondent non. L'interface demande au joueur A s’il veut jouer la carte. Le joueur répond non. L'interface demande au joueur de sélectionner une de ses cartes à retourner. Le joueur clique sur la carte qu'il veut retourner. La carte se retourne. Fin du tour du joueur A.

**Scenario: Voler une carte**

C'est le tour du joueur A. L'interface demande au joueur de faire un choix entre la pioche et la défausse. Il demande de piocher une carte dans la pioche. La première carte de la pioche se retourne, tous les joueurs peuvent la voir. L’interface demande si quelqu'un veut la carte. Le joueur C clique sur "oui". Le joueur A peut voir le jeu du jour C, et l'interface lui demande s'il accepte ou non. Le joueur A accepte. Le joueur C joue la carte volée, en cliquant sur une de ses carte qui sera remplacée par cette dernière. Fin du tour du joueur C. L'interface demande au joueur A de sélectionner une carte du joueur C. Il la sélectionne. L’interface demande au joueur A de sélectionner une carte de son jeu à remplacer. Il remplace cette carte dans son jeu en cliquant sur une des cartes et la carte défaussée remplace la carte volée dans le jeu du joueur C.

**Scenario: faire une ligne de 3**

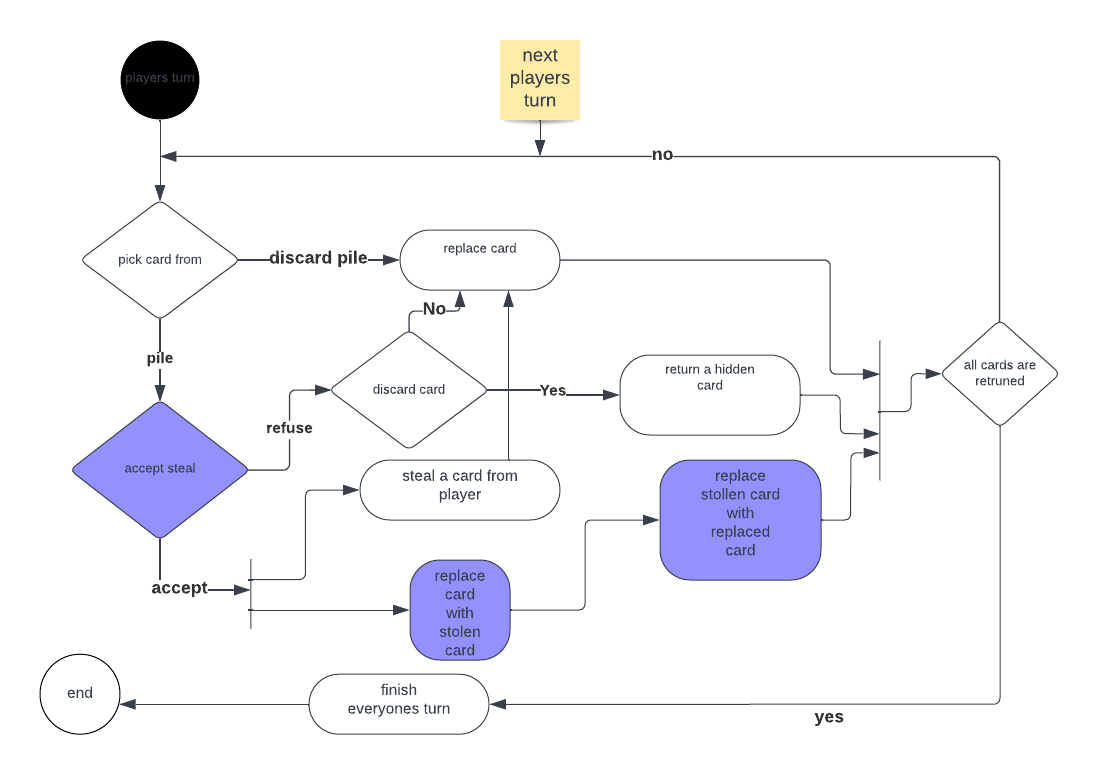
C'est le tour du joueur A. L'interface demande au joueur de faire un choix entre la pioche et la défausse. Il choisit la pioche. Le joueur A a deux 7 en ligne. L'interface retourne la première carte de la pioche. La carte est un 7. L'interface demande au joueur s’il veut jouer la carte. Le joueur sélectionne oui. L'interface demande au joueur de sélectionner une carte à échanger contre le 7. Le joueur A sélectionne la carte à côté de ses deux 7 cote a cote. La carte sélectionnée part dans la défausse, et le 7 la remplace. L'interface détecte la suite de 3x7. Elle affiche que le joueur a une suite de 3 cartes identiques. Les cartes disparaissent de l'interface (prennent la valeur 0), les trous ne sont pas comblés. Fin du tour du joueur A.

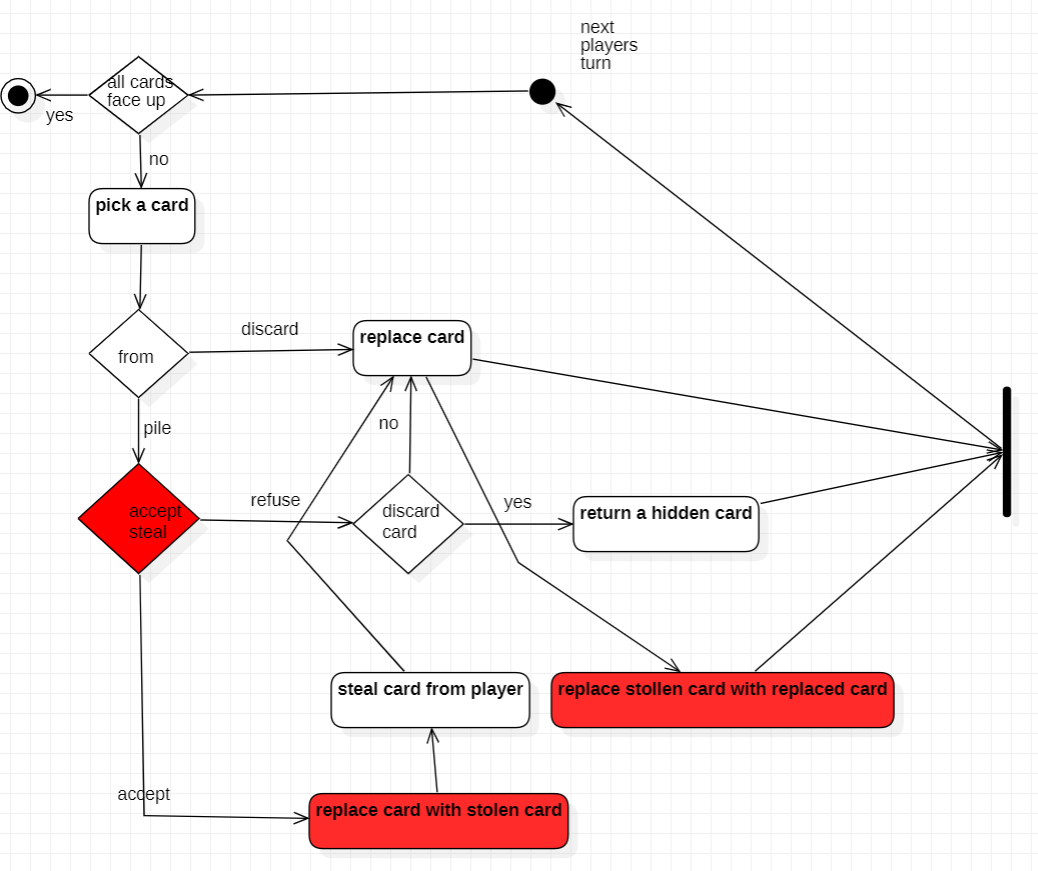
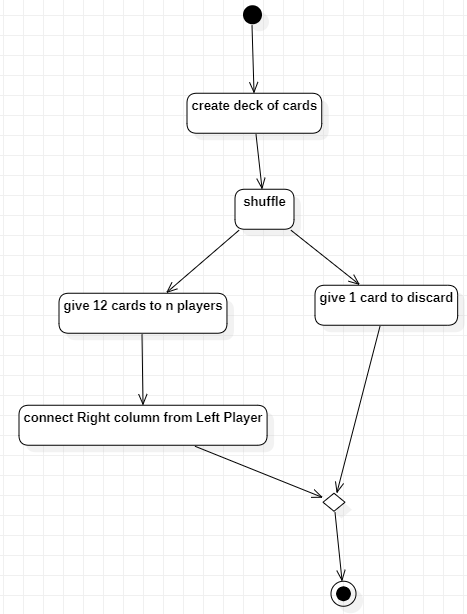
**Scénario: Gagner la partie**

Le joueur A n'a plus qu'une carte face cachée. Cas du scénario "ne rien piocher". Le joueur clique sur la seule carte qu'il peut retourner. L'interface affiche que le joueur A a retourné toutes ses cartes. Fin du tour du joueur A. Les joueurs B,C,D font leur tour. A la fin du tour du joueur D, l'interface affiche que c'est la fin de la partie. Les éventuelles cartes face cachées des joueurs sont retournées, et les points sont comptés. L'interface affiche le nom du joueur gagnant. Si ex-aequo, l'interface affiche les deux joueurs ex aequo.

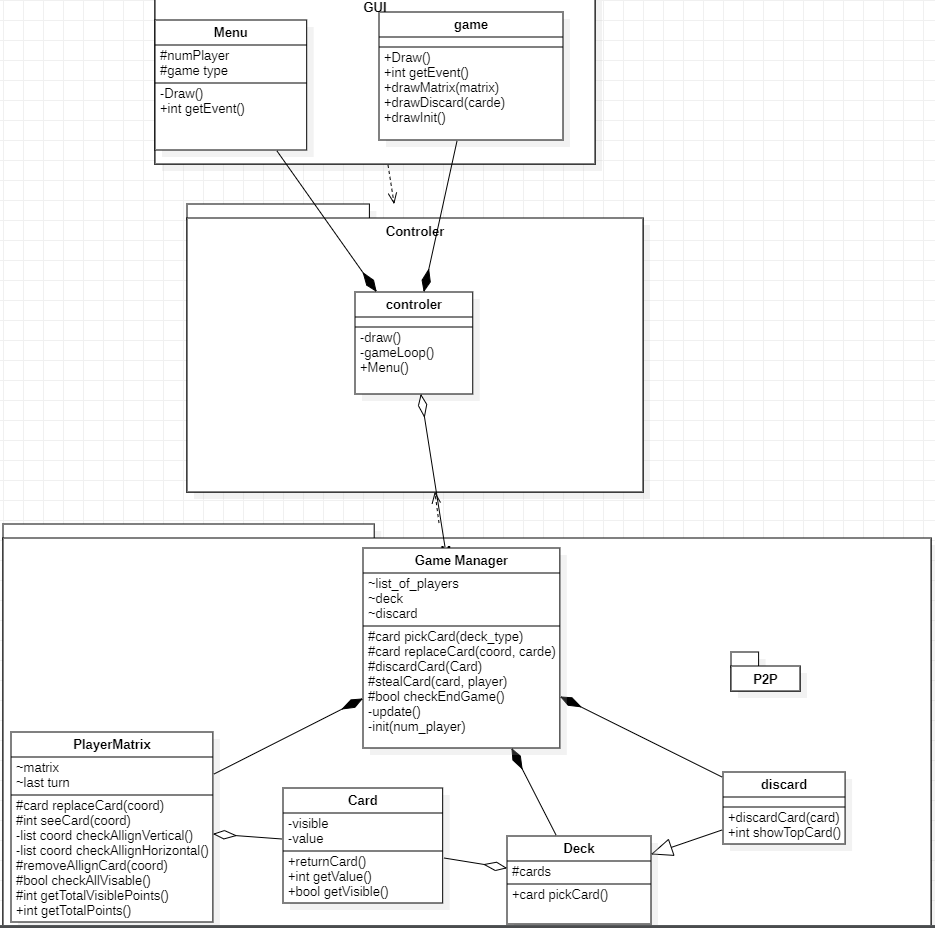
Nous avons ensuite réalisé plusieurs diagrammes pour simplifier la programmation du jeu :

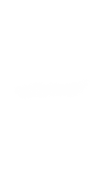
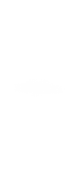
**Diagrammes d’activité**

Actions possibles du joueur lors du déroulement d’une partie

Actions du code lors du déroulement d’une partie

Actions du init() au début d’une partie

**Diagramme de classe**



La classe Card a un lien d’agrégation avec les classes Deck et PlayerMatrix, Deck et Discard ont un lien d’héritance, et PlayerMatrix, Deck et Discard ont un lien de composition avec le Game Manager.

La classe Game Manager : Elle contient la liste des joueurs, la pioche et la défausse. Ses méthodes permettent le bon déroulement du jeu, et sa mise en place au début de la partie.

La classe Discard : Elle contient deux méthodes qui permettent de mettre une carte dans la défausse et de montrer la dernière carte posée dans la défausse.

La classe PlayerMatrix : Elle contient une matrice qui correspond au jeu d’un joueur en plus de la colonne de son voisin de gauche, et un booléen last turn qui est true si un joueur a retourné toutes ses cartes. Ses méthodes permettent de gérer le jeu d’un joueur (vérifier que des cartes sont alignées, vérifier si toutes les cartes sont retournées, calculer le total des points d’un joueur, retirer les cartes alignées, etc…).

La classe Card : La classe carte contient son état (retournée ou face cachée) et sa valeur. Ses méthodes lui permettent d’obtenir la valeur d’une carte et de la retourner.

La classe Deck : Elle contient une liste de cartes, et une méthode piocher qui permet aux joueurs de piocher des cartes.

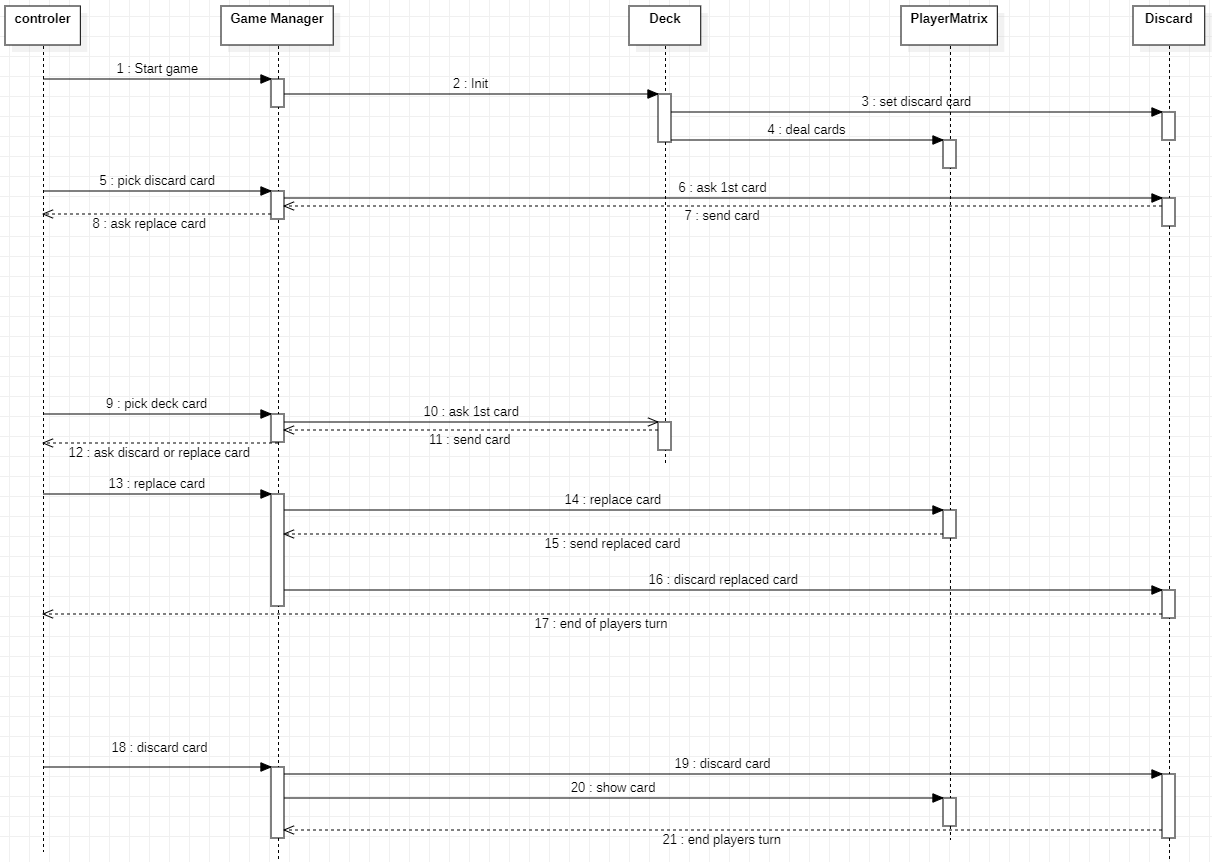
Nous n’avons pas encore trouvé de bibliothèque pour l’interface du jeu, le reste du schéma n’est donc pas forcément définitif.

Une image contenant texte, carte, groupe, plusieurs

Description générée automatiquementDiagramme de cas d’utilisation

Cas de vol de carte : Un joueur peut demander de voler la carte qu’un autre joueur a pioché. Si le joueur accepte, le joueur N peut piocher cette carte dans la pioche. A partir de ce moment-là, le joueur peut remplacer une de ses cartes par cette dernière. Ce cas ne peut donc se réaliser uniquement si le joueur N a demandé de voler la carte et qu’il l’a pioché après accord. Le joueur N doit ensuite défausser sa carte s’il en a pioché une.

**Diagramme de séquence**



Les deux cas « pick deck card » et « pick discard card » sont deux cas séparés mais nous avons choisi de les représenter sur le même diagramme. De même pour « replace card » et « discard card »

Comme expliqué plus tôt, PlayerMatrix correspond au jeu du joueur associé à la colonne de gauche de son voisin de droite. Deck correspond a la pioche, et Discard à la défausse. Le game manager va servir d’intermédiaire entre le contrôleur et le jeu, et le contrôler entre le joueur et le game manager (appui sur une touche, clic sur une option etc…).