UNIVERSITÉ LIBRE DE BRUXELLES

INFO-F209 : PROJET D'INFORMATIQUE II

WIZARD POKER

Software Requirements Document

Auteurs :

Yasin ARSLAN
Youcef BOUHARAOUA
Jalal NAH
Sacha MEDAER
Youssef SITIL
Miguel TEROL ESPINO
Jacky TRINH

Titulaires :
Joël GOOSSENS
Christian HERNALSTEEN
Assistants :
Keno MERCKX
François GÉRARD
Jacopo DE STEFANI

26 février 2016

Table des matières

Chapitre 1

Introduction

1.1 But du projet

Ce document a pour but de présenter, dans le cadre d'une deuxième année de Bachelier en Sciences informatiques, une description détaillée de notre projet multidisciplinaire. Le but, les caractéristiques, l'interface du système ainsi que les contraintes auxquelles le programme est soumis sont expliqués. Ce présent rapport est destiné aussi bien au client qu'au développeur soucieux de comprendre l'architecture du programme.

1.2 Cadre du projet

Le programme est un jeux nommé Wizard-Poker, un jeu de carte fantastique où deux utilisateurs s'affrontent dans un duel. Chaque utilisateur possède une collection de carte avec laquelle il peut créer des decks, paquet de vingt cartes avec lequel il commence un duel. Ces cartes sont uniquement de type créature.
Au début d'un duel, chaque utilisateur possède un maximum de vingt points de vie appelé PV et 3 cartes en main piochées au préalable de son deck. À chaque début de tour,le joueur concerné pioche une carte et gagne un point d'énergie. Durant son tour, un joueur peut invoquer une créature et/ou attaquer avec une de ses créatures sur le plateau de jeu. Un utilisateur gagne le duel une fois que son adversaire abandonne ou tombe à 0 points de vie.

1.3 Glossaire

Mot	**Définition**
ActionManager	Système gérant les interactions lors d'une partie
Chat	Système de conversation entre joueurs
Cible	Joueur ou carte pouvant être attaqué
Cimetière	Endroit où se retrouvent les cartes éliminées
Classement	Performances d'un joueur par rapport aux autres en fonctions de ses prestations de jeu
Client	Entité qui se connecte au serveur.
Collection	Toutes les cartes accessibles d'un joueur
Compte	Chaque joueur possède un espace réservé contenant ses informations personnelles
Contre-attaque	Offensive lancée en réponse à une attaque
Créature	Cartes ayant un coût en énergie, une valeur d'attaque et un montant de points de vie
Database	Zone de stockage
Deck	Groupe de cartes choisi par le joueur pour un duel
Duel	Combat singulier entres 2 adversaires
IA	Intelligence artificielle
Invité	Personne n'ayant pas encore de compte associé au jeu
Joueur	Participant à un duel
Main	Groupe de cartes accessible au joueur lors de son tour de jeu
Matchmaking	La mise en relation des joueurs pour le duel
Plateau de jeu	Lieu où sont disposées les cartes en action
Points d'attaque	Détermine la capacité d'une créature à infliger des dégats
Points d'énergie	Détermine la coût d'utilisation d'une carte
Points de vie (PV)	Détermine l'état de santé d'un joueur ou d'une créature
Serveur	Processus qui gère les connexions des clients et leurs interactions avec le système.
Sort	Cartes ayant un coût en énergie et un effet spécial

1.4 Historique

1.4.1 SRD

Version	Date	Auteur	Commentaire
1.0	03.12.2015	Miguel T.	Croquis Use Case.
1.0	04.12.2015	Youssef S.	Croquis Class Dia.
1.0	04.12.2015	Yasin A.	Màj Class Dia. + Croquis Use Case.
1.0	08.12.2015	Sacha M.	Squelette du SRD.
1.0	08.12.2015	Jacky T.	Croquis Séq.Diagram.
1.0	09.12.2015	Sacha M.	Introduction SRD.
1.0	11.12.2015	Youcef B.	Croquis Séq.Diagram.
1.0	12.12.2015	Youssef S.	Croquis Séq.Diagram.
1.1	14.12.2015	Jacky T.	Séq.Diagram - Déroulement d'une partie.
1.1	14.12.2015	Miguel T.	Besoin de l'utilisateur.
1.1	14.12.2015	Jalal N.	Use case - Un tour de jeu.
1.1	14.12.2015	Yasin A.	Class Diagram.
1.1	15.12.2015	Youcef B.	Séq.Diagram - Connexion + Fin de partie.
		Phase 2	
2.0	21.02.2016	Sacha M.	Structure v2.0 + introduction.
2.0	22.02.2016	Sacha M., Miguel T.	Exigences non-fonctionnelles.
2.1	24.02.2016	Jacky T.	Division du diagramme séquentielle.
2.1	24.02.2016	Jacky T.	Amélioration besoin système.
2.1	24.02.2016	Youcef B.	Amélioration besoin utilisateur.
2.1	24.02.2016	Youcef B.	Améliration besoin système.
2.1	24.02.2016	Yasin A.	Ajout des diagrammes de classe.
2.1	24.02.2016	Yasin A.	Ajustement du diagramme Use Case View.
2.2	25.02.2016	Jacky T.	Amélioration use case tour d'un joueur.
2.2	25.02.2016	Jalal N.	Diagramme d'activité.
2.2	26.02.2016	Miguel T.	Ajout besoins fonctionnels de l'utilisateur.
2.2	26.02.2016	Miguel T.	Ajout éléments glossaire.
2.3	26.02.2016	Sacha M.	Orthographe + syntaxe + index.
2.3	26.02.2016	Miguel T., Yasin A., Sacha M.	Relecture global.

1.4.2 Code

Version	Date	Auteur	Commentaire
		Phase 2	
1.0	06.02.2016	Yasin A.	Client-Serveur-Database.
1.1	09.02.2016	Jacky T., Youcef B.	Début chat.
1.1	09.02.2016	Yasin A.	Passage cpp & hpp.
1.2	10.02.2016	Jacky T.	Gestion des personnes en ligne.
1.2	11.02.2016	Jalal N.	Classement & début User.
1.2	12.02.2016	Jacky T.	Chat terminé avec ChatManager.
1.2	12.02.2016	Jacky T.	Gestion des déconnexions.
1.3	13.02.2016	Youssef S.	Deck & Collection.
1.3	13.02.2016	Yasin A.	Restructuration Deck & Collection.
~~1.3~~	~~14.02.2016~~	~~Miguel T.~~	~~CardDatabase JSON.~~
1.3	16.02.2016	Miguel T.	CardDatabase TXT.
1.3	16.02.2016	Yasin A.	Fusion CardDatabase avec Client-Serveur.
1.3	18.02.2016	Yasin A.	Création d'un deck.
1.3	19.02.2016	Jacky T.	Début Matchmaking.
1.3	19.02.2016	Yasin A.	Début Game.
1.3	20.02.2016	Jacky T.,Yasin A.	Matchmaking fonctionnel.
1.4	22.02.2016	Yasin A.	Game fonctionnel.
1.4	22.02.2016	Yasin A.	Amélioration intégrale du code.
1.5	22.02.2016	Youcef B.	Gestion collection & deck.
1.5	23.02.2016	Yasin A.	Correction de tout les warnings.
1.6	24.02.2016	Miguel T.	Testing.
1.6	25.02.2016	Jacky T. & Yasin A. Youcef B.	Correction de tout les bugs rencontrés.

1.5 Vue d'ensemble

Cette section décrit les différentes parties du document. La première, reprend les services que le système doit fournir à l'utilisateur ainsi que les contraintes de fonctionnement du système. Ceci est réalisé notamment grâce à des diagrammes use case. La deuxième partie, quant à elle, décrit en détail les fonctionnalités du système. Des diagrammes Use Case ainsi que des diagrammes UML sont utilisés.

Chapitre 2

Besoins de l'utilisateur

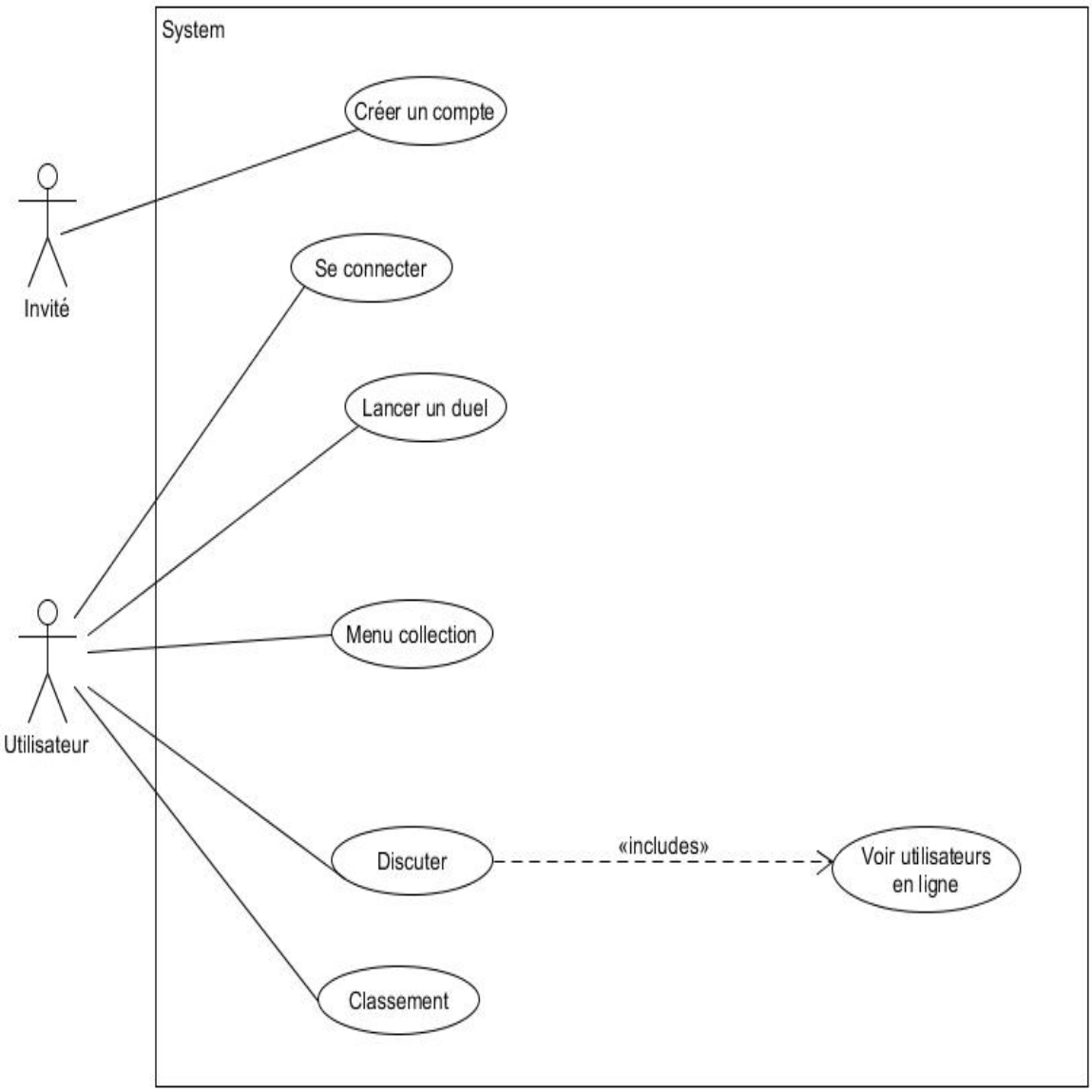

FIGURE 2.1 – Use case view

2.1 Exigences fonctionnelles

2.1.1 Inscription

- Cas général : Création d'un nouveau compte via un nom d'utilisateur unique et un mot de passe.
- Pré condition : L'utilisateur ne possède pas de compte.
- Post condition : Un nouveau compte est créé sur le serveur.
- Cas exceptionnel : Nom d'utilisateur déjà utilisé, l'utilisateur est averti et redirigé.

2.1.2 Connexion

- Cas général : Connection au serveur avec un identifiant et un mot de passe.
- Pré condition : L'utilisateur n'est pas connecté.
- Post condition : L'utilisateur est connecté au serveur.
- Cas exceptionnel : Les données introduites sont incorrectes, le serveur averti le client et celui-ci est amené à reessayer.

2.1.3 Classement

- Cas général : L'utilisateur consulte le classement des différents joueurs.
- Pré condition : L'utilisateur est connecté et est dans le menu adéquat.
- Post condition : L'utilisateur a consulté le classement.

2.1.4 Créer un deck

- Cas général : Création d'un nouveau deck avec les cartes de sa collection.
- Pré condition : L'utilisateur est connecté au serveur et il ne possède pas 5 decks complets.
- Post condition : Le deck contient 20 cartes et respecte les conditions d'un deck.

2.1.5 Matchmaking

- Cas général : L'utilisateur lance la recherche d'adversaire.
- Pré condition : L'utilisateur est bien connecté au serveur.
- Post condition : Un adversaire a été trouvé, et la liaison est créée.

2.1.6 Jouer un tour

- Cas général : L'utilisateur effectue éventuellement des actions et termine son tour.
- Pré condition : Le joueur est en train de disputer un duel.
- Post condition : Ce n'est plus son tour de jouer.
- Cas exceptionnel : Déconnection brutale d'un des joueurs. Forfait pour le joueur déconnecté.

2.1.7 Abandonner

- Cas général : L'utilisateur peut déclarer forfait durant un duel.
- Pré condition : C'est le tour de l'utilisateur.
- Post condition : Défaite et mise à jour du classement.

2.2 Exigences non fonctionnelles

2.2.1 Exigences du produit

2.2.1.1 Exigences d'efficacité

- L'interface graphique doit être parfaitement synchronisée en temps réel pour chaque client en fonction des actions individuelles de ceux-ci.

2.2.1.2 Exigences de fiabilité

- Une commande incorrecte introduite par l'utilisateur doit pouvoir lui être signalé sans interruption du programme.

2.2.1.3 Exigences d'ergonomie

- L'interface graphique doit être orgranisée et facile d'utilisation.
- De la documentation pour utilisateur doit être disponible.
- La prise en main du système doit être abordable.
- Les messages d'erreur doivent indiqués comment résoudre ceux-ci.
- Un tutoriel d'introduction est mis à disposition du joueur débutant.

2.2.2 Exigences d'organisation

2.2.2.1 Exigences de livraison

- Le programme complet doit être livré au plus tard pour le 25 mars 2016.

2.2.3 Exigences externes

2.2.3.1 Exigences de l'éthique

- Aucun propos diffamatoires ou racistes ne peut être tenus sur le chat du jeu sous peine d'exclusion.

2.3 Exigences de domaine

Le jeux Wizard Poker est un jeu stratégique de carte à jouer. L'utilisateur peut créer un personnage et le faire évoluer. Wizard Poker comporte un seul gamemode, le duel. Le programme doit essayer de respecter les termes propres à ce genre de jeux pour une prise en main aisée.

Chapitre 3

Besoins du système

3.1 Exigences fonctionnelles

3.1.1 Inscription

L'utilisateur, désireux de jouer, devra passer par un système d'inscription auprès du serveur. Il devra donner un nom d'utilisateur unique ainsi qu'un mot de passe quelconque. Si le nom d'utilisateur est déjà utilisé sur la database, le serveur va avertir l'utilisateur et lui redemander un nouveau nom d'utilisateur. Une fois l'inscription réussi, le serveur va mettre à jour la database et l'utilisateur pourra se connecter.

3.1.2 Connexion

L'utilisateur se connecte grâce à son nom d'utilisateur et son mot de passe. La tentative de connection passe par le serveur. Si l'utilisateur est déjà connecté, il sera averti. La connection échoue si le nom d'utilisateur n'existe pas dans la database ou si le mot de passe ne correspond pas au nom d'utilisateur donné.

3.1.3 Classement

L'utilisateur, une fois connecté, pourra consulté le classement qui recense tout les comptes inscrit sur le serveur. À chaque compte est associé un nombre de victoires et un nombre de défaites. Le classement détermine le niveau d'un joueur par rapport aux autres en fonction de la différence entre le nombre de victoires et de défaites. L'utilisateur peut consulter le classement depuis le menu principal.

3.1.4 Créer un deck

L'utilisateur choisit ses cartes pour composer un nouveau deck. Ces cartes doivent exister dans la collection de l'utilisateur. Le deck devra également respecter les contraintes d'un deck. On ne peut avoir plus de 2 fois la même carte et le deck comporte exactement 20 cartes.

3.1.5 Matchmaking

Le matchmaking se charge d'établir une connection entre 2 utilisateurs. Une fois que 2 utilisateurs lance le matchmaking, ils deviennent adversaires sur le même duel. Chaque joueur reçoit comme information sur son advesaire, son nom d'utilisateur et leur nombre de victoires et défaites.

3.1.6 Jouer un tour

L'utilisateur peut choisir une action dans son menu. Il peut jouer une carte, attaquer avec une créature sur le terrain, discuter avec un autre utilisateur, terminer son tour ou abandonner le duel.

3.1.7 Abandonner

Tout le monde s'est déjà rendu dans une situation où la victoire n'est plus du tout envisageable. Afin de gagner du temps, l'utilisateur a la possibilité d'abandonner. Le duel prendra alors fin, l'adversaire sera déclaré gagnant.

3.2 Exigences non fonctionnelles

3.2.1 Exigences du produit

3.2.1.1 Exigences de portabilité

- Le programme doit pouvoir s'exécuter sur les ordinateurs des salles machines, plus connu sous le nom de PaDi, Parc Didactique informatique de l'Université Libre de Bruxelles.

3.2.1.2 Exigences de fiabilité

- En cas d'interruption abrupte d'un duel, les joueurs doivent pouvoir récupérer les informations de celle-ci.
- La database doit pouvoir être sécurisé et les utilisateurs ne peuvent y avoir accès sans passer par le serveur.

3.2.2 Exigences d'organisation

3.2.2.1 Exigences d'implémentation

- Le programme doit être implémenté en C++14 et doit pouvoir être compilé à l'aide de G++-5.

3.2.3 Exigences externes

3.2.3.1 Exigences sur la legislation

- Le système s'engage à ne fournir aucunes informations personnelles à propos des utilisateurs excepté leur nom d'utilisateur.

3.2.3.2 Exigences de l'interopérabilité

- Deux joueurs doivent pouvoir s'affronter et communiquer indépendamment de la machine sur laquelle le jeux s'exécute pour autant que celle-ci respecte les exigences du produit.

3.3 Design et fonctionnement du système

3.3.1 Diagramme de classe

3.3.1.1 Général

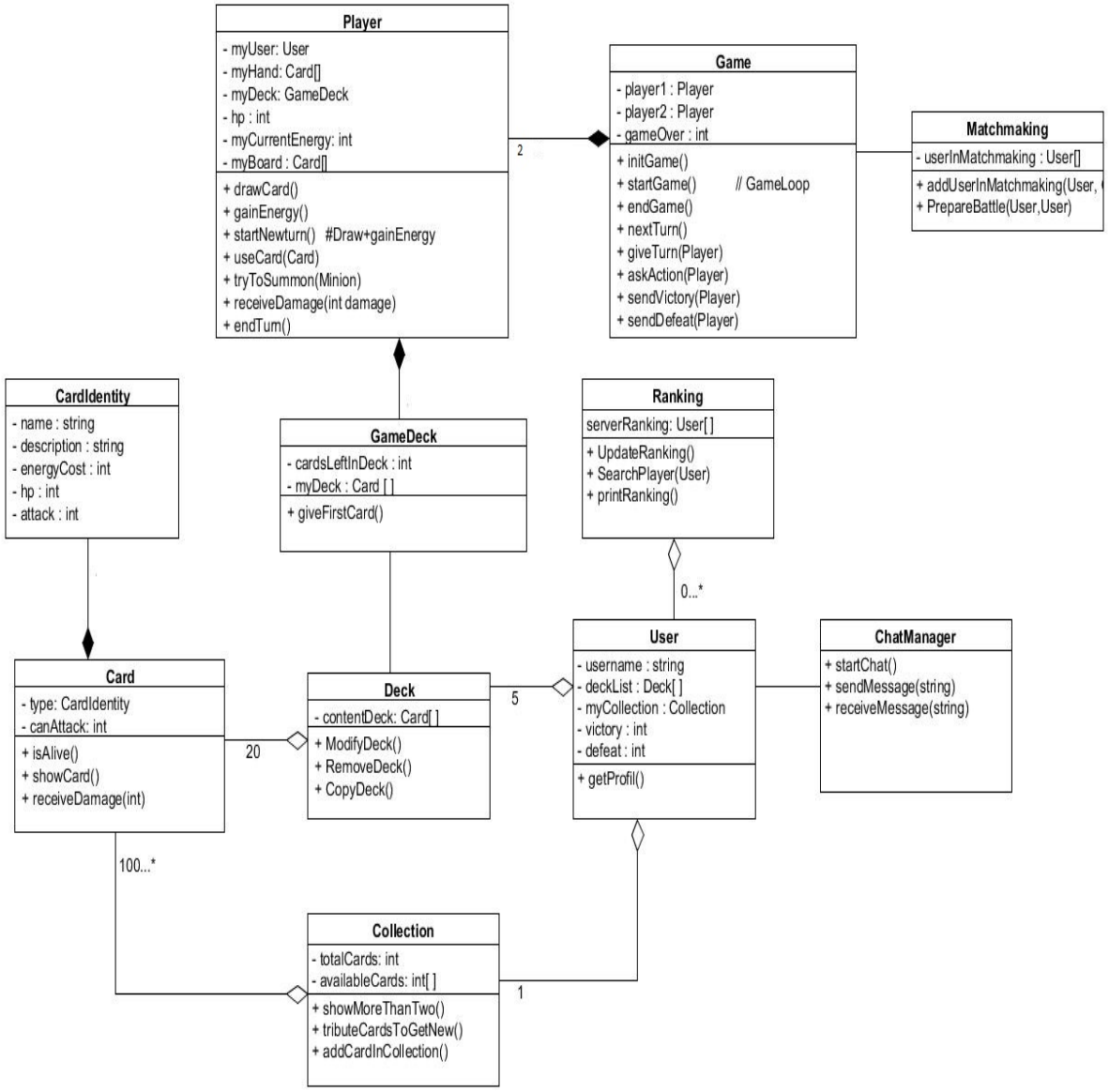

FIGURE 3.1 – Diagramme de classe général

Analyse

Diagramme généralisé illustrant les relations entre les principaux objets.

3.3.1.2 Client-Serveur-Database

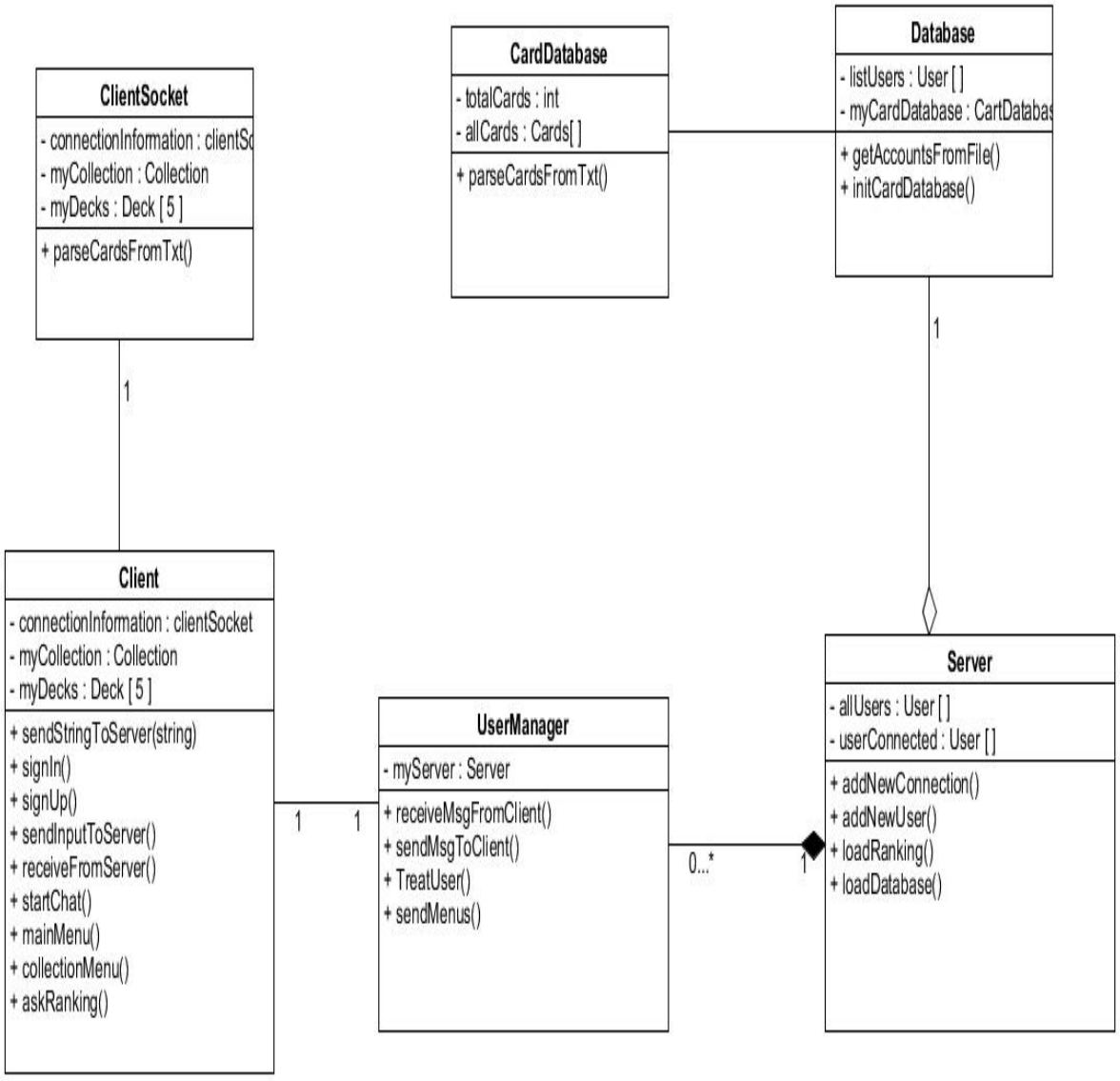

FIGURE 3.2 – Diagramme de classe client-serveur

Analyse

Diagramme de classe reprenant l'idée générale des objets instanciés pour les échanges entre le client et le serveur. La database est également représenté comme étant lié au serveur. Celle-ci sera chargé via des fichiers txt.

3.3.2 Diagramme de séquence

3.3.2.1 Lancement du jeu et invocation

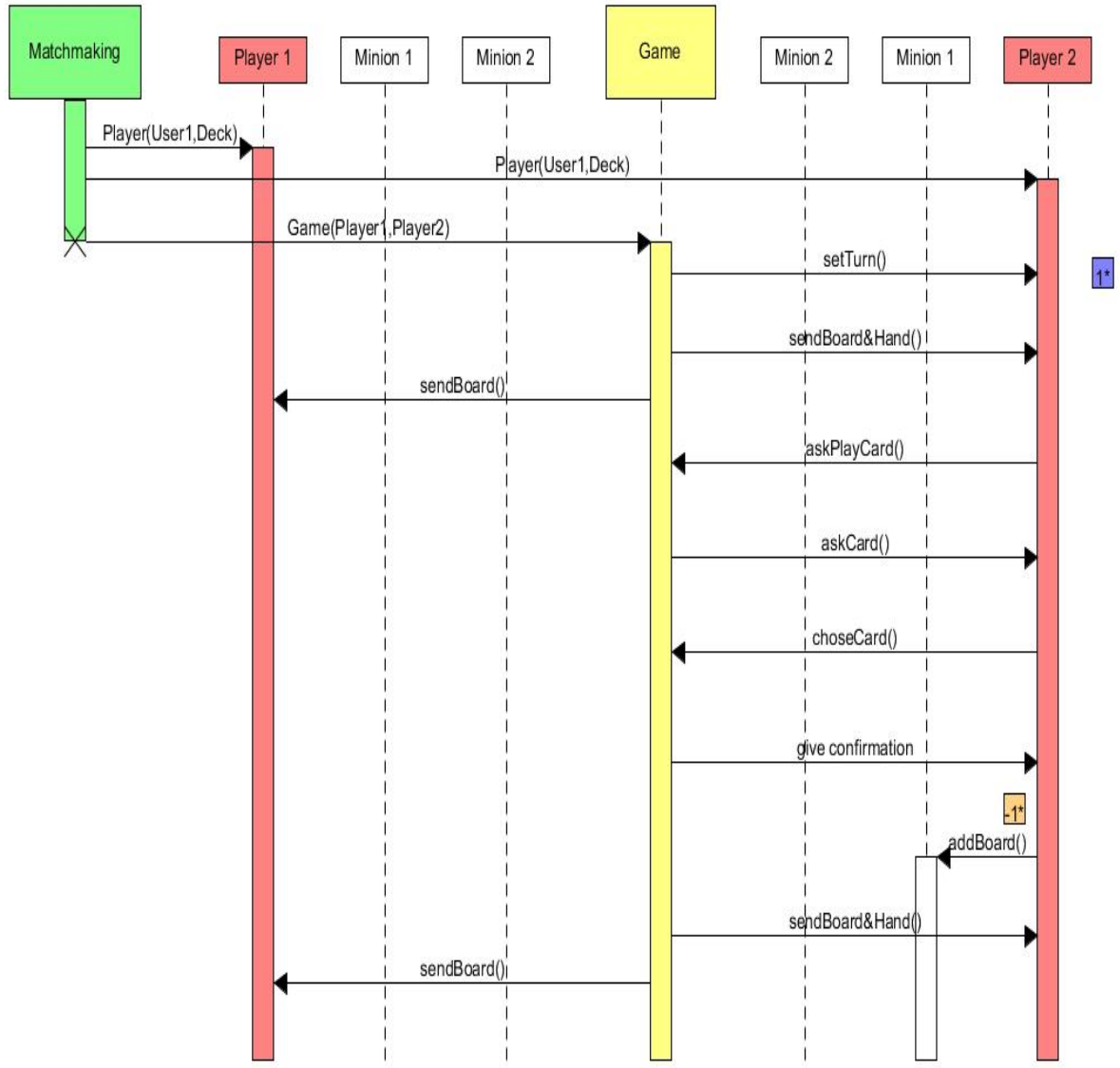

FIGURE 3.3 – Diagramme de séquence d'un début de partie et invocation

Analyse

Ce diagramme de séquence montre les interactions entre les différents objets au lancement du jeu et lors des invocations de créatures.

3.3.2.2 Attaque d'une créature envers une autre

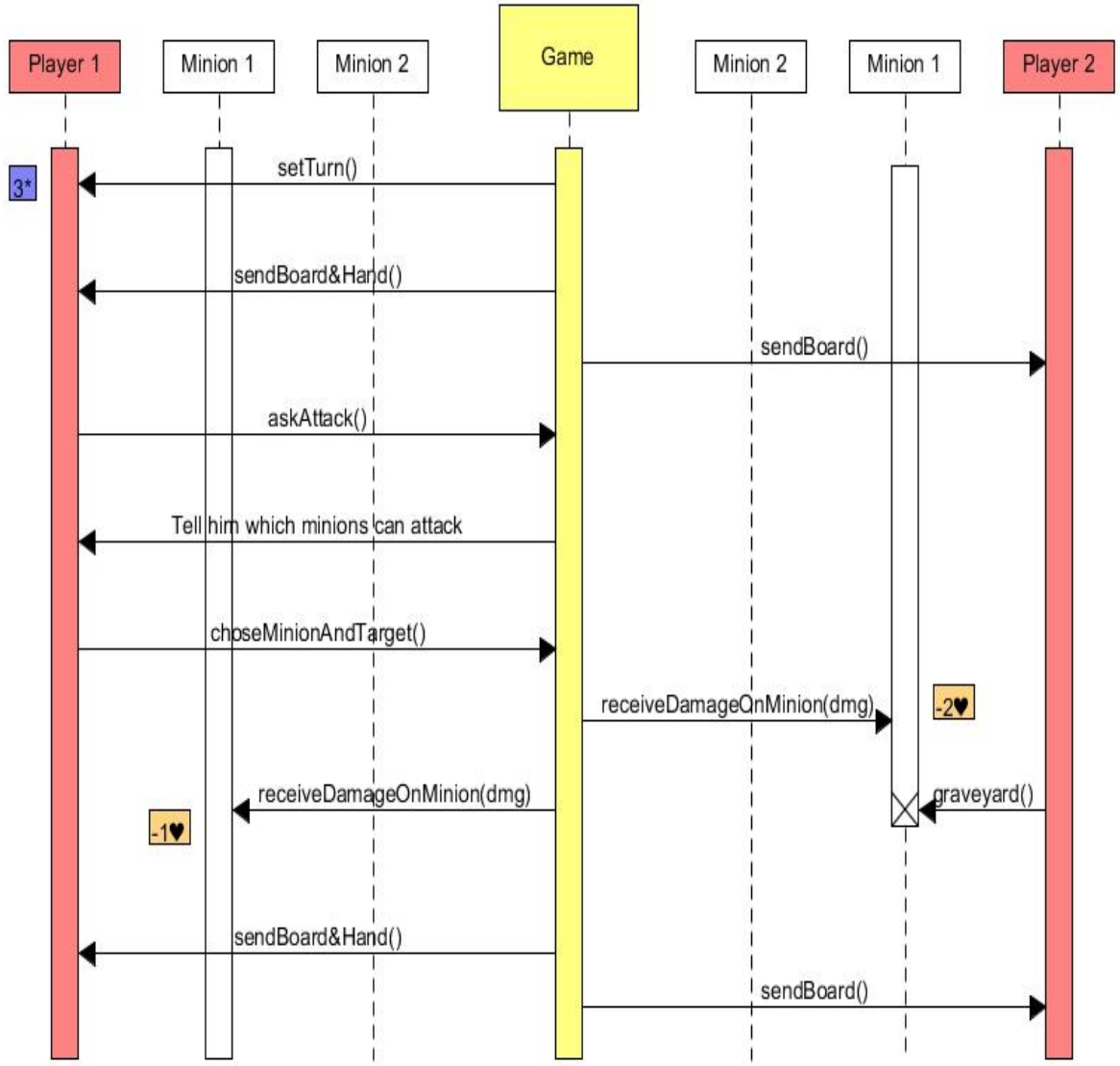

FIGURE 3.4 – Attaque d'une créature

Analyse

Ce diagramme de séquence montre les interactions entre les différents objets lors d'une attaque d'une créature contre une autre.

3.3.2.3 Attaque d'une créature envers un joueur adverse

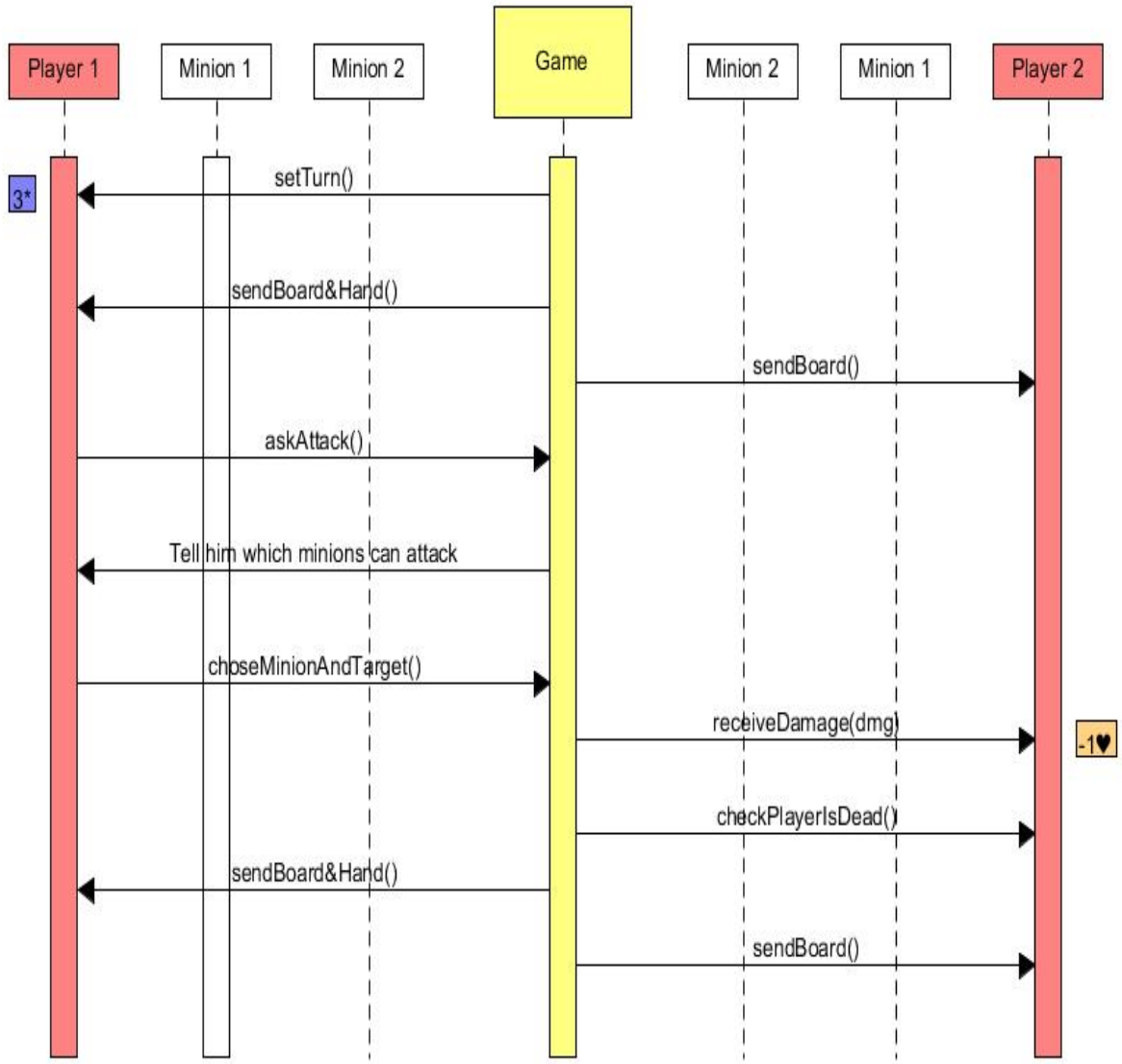

FIGURE 3.5 – Attaque d'une créature

Analyse

Ce diagramme de séquence montre les interactions entre les différents objets lors d'une attaque d'une créature contre le joueur adverse.

3.3.3 Use Case

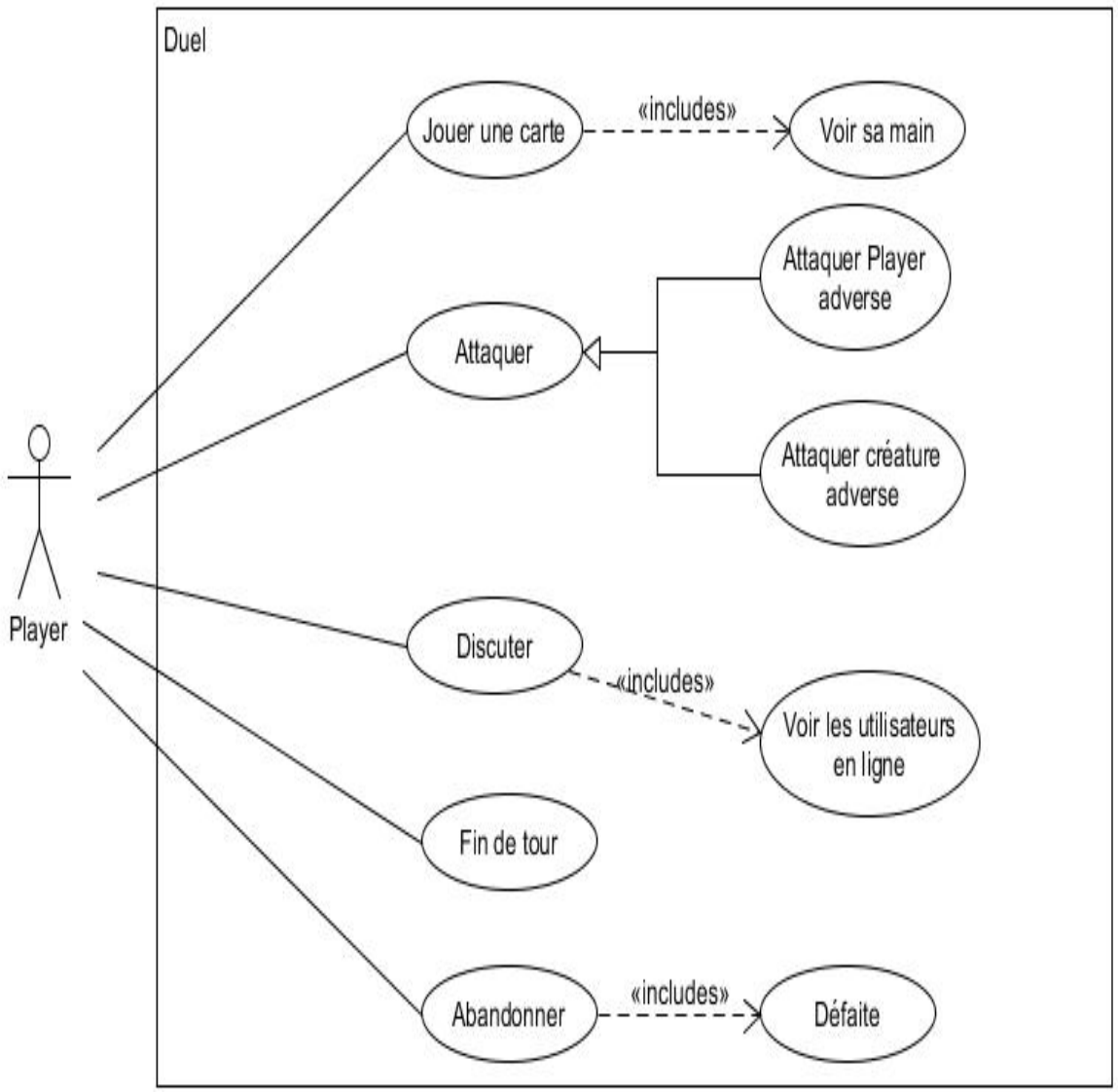

FIGURE 3.6 – Use case représentant les actions lors d'un tour

Analyse

Ce use case retrace les différentes actions possibles lors d'un tour.

3.3.4 Diagramme d'activité

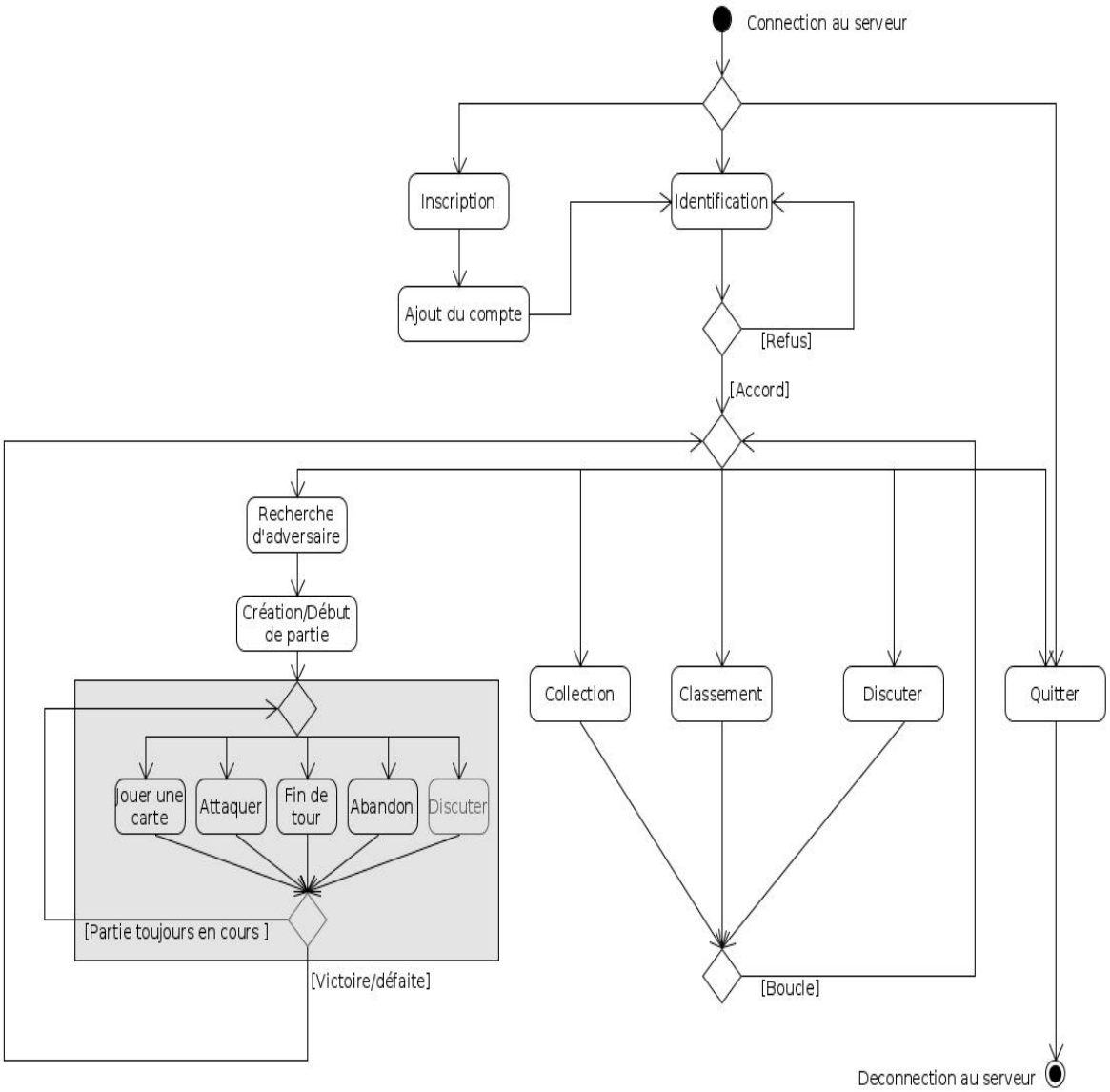

FIGURE 3.7 – Diagramme d'activité client-server

Analyse

Ce diagramme d'activité représente les différentes opérations en termes d'action que l'utilisateur peut réaliser.

Index