



UNIVERSITÉ LIBRE DE BRUXELLES

INFO-F-209

PROJET D'ANNÉE 2

---

## The Pro Quidditch Manager

---

*Auteurs :*

Anthony CACCIA,  
Antoine CARPENTIER,  
Titouan CHRISTOPHE,  
Jerôme HELLINCKX,  
Florentin HENNECKER,  
Mircea MUDURA

*Professeurs :*

Christian HERALSTEEEN,  
Joël GOOSSENS

*Assistants :*

Stéphane FERNANDES MEDEIROS,  
Naïm QACHRI,  
Mohamed Amine YOUSSEF

2ème année de bachelier  
2013 - 2014

# Table des matières

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	But du projet . . . . .	3
1.1.1	Concept . . . . .	3
1.1.2	Le Quidditch . . . . .	3
1.2	Historique du document . . . . .	3
1.3	Glossaire . . . . .	5
<b>2</b>	<b>Besoins de l'utilisateur</b>	<b>7</b>
2.1	Exigences fonctionnelles . . . . .	7
2.1.1	En bref . . . . .	7
2.1.2	Écran de démarrage . . . . .	7
2.1.3	Menu principal . . . . .	9
2.1.4	Gestion d'équipe . . . . .	10
2.1.5	Gestion du stade . . . . .	12
2.1.6	Procédure d'enchère sur un joueur présent sur le Player Market . . . . .	15
2.1.7	Le championnat . . . . .	17
2.1.8	Matches amicaux . . . . .	19
2.1.9	Phase de jeu . . . . .	21
2.1.10	Cas d'utilisation indirects . . . . .	25
2.2	Exigences non fonctionnelles . . . . .	26
2.3	Exigences de domaine . . . . .	26
<b>3</b>	<b>Besoins du système</b>	<b>27</b>
3.1	Exigences fonctionnelles . . . . .	27
3.1.1	Système - partie gestion . . . . .	27
3.1.2	Cas d'utilisation du championnat . . . . .	31
3.2	Exigences non fonctionnelles . . . . .	34
3.3	Design et fonctionnement du système . . . . .	34
3.3.1	Composants . . . . .	34
3.3.2	Envoyer des messages . . . . .	34
3.3.3	Phase de jeu . . . . .	35
3.3.4	Le gestionnaire de connexion . . . . .	35
3.3.5	Phase de gestion . . . . .	39
3.3.6	Terrain . . . . .	39
<b>A</b>	<b>À propos de la librairie graphique</b>	<b>43</b>
<b>B</b>	<b>Dans le futur...</b>	<b>44</b>

<b>C</b>	<b>Diagramme de classes général</b>	<b>45</b>
<b>D</b>	<b>Diagramme de composants</b>	<b>46</b>

# Chapitre 1

## Introduction

### 1.1 | But du projet

Le projet de jeu électronique **The Pro Quidditch Manager** est né des besoins de la formation en sciences informatique à l'ULB (en deuxième année de bachelier). Ce projet regroupe 6 étudiants et leur fixe pour objectif de créer un jeu révolutionnaire sur le thème du Quidditch. Ce projet permet de mettre en oeuvre les concepts théoriques rencontrés dans les autres cours de BA2 sciences informatiques, et permet en outre d'approcher les problématiques de gestion de groupe et de maintien de projet sur le long terme.

#### 1.1.1 | Concept

Dans ce jeu, vous affronterez d'autres *managers* en ligne dans une course ensorcelée vers la gloire et la richesse. Vous prendrez des décisions stratégiques pour votre équipe en recrutant des coaches, en achetant, revendant et entraînant des joueurs, et en faisant prospérer votre stade. Sur **The Pro Quidditch Manager**, vous bâtissez votre propre *dreamteam* de Quidditch !

#### 1.1.2 | Le Quidditch

Selon le souhait du client, les règles utilisées dans le jeu seront, autant que possible, proches des règles décrites sur Wikipédia<sup>1</sup>.

#### *Copyright*

Le mot Quidditch, créé par J.K. Rowling, appartient actuellement à la société *Warner Bros*. Le logo sur la page de garde appartient à l'*International Quidditch Association*.

### 1.2 | Historique du document

---

1. <http://en.wikipedia.org/wiki/Quidditch>, dernière consultation le 18/12/2013

<b>Version</b>	<b>Éditeur</b>	<b>Date</b>	<b>Commentaire</b>
1.0	(Global)	19/12/2013	Cas d'utilisations, classes, description générale
1.1	Anthony	17/02/2014	Mise à jour du diagramme de gestion
1.2	Anthony	17/02/2014	Désempiguation EN-FR
1.3	Anthony	18/02/2014	Mise à jour de divers diagrammes
1.4	Anthony	19/02/2014	Détails des sponsoring et coaches ajoutés
1.5	Titou	19/02/2014	Adaptation du diagramme JSON
1.6	Mircea	20/02/2014	Ajout d'une section à propos des enchères
1.7	Anthony	21/02/2014	Diagramme de classes général ajouté
2.0	Anthony	21/02/2014	Release phase 2
2.1	Anthony	11/03/2014	Nouvelle page de garde
2.2	Anthony	14/03/2014	Big diagramme de composants ajouté en annexe
2.3	Anthony	14/03/2014	Diagrammes de classes mis à jours
2.4	Anthony	14/03/2014	Suppression des parties non implémentées dans le texte
2.5	Anthony	14/03/2014	Suppression des parties non implémentées dans les use cases
2.6	Anthony / Mircea	14/03/2014	Ajout des diagrammes d'activité

## 1.3 | Glossaire

**aptitude** Valeur numérique entre 0 et 20 qui représente une caractéristique spécifique d'un membre.

**attrapeur** Joueur qui peut attraper le *vif d'or*.

**batteur** Joueur possédant une batte qui peut taper les cognards.

**cas d'utilisation** Décrit les fonctionnalités du système du point de vue de l'*utilisateur* de ce système.

**client** Programme installé sur la machine de l'utilisateur qui lui permet de se connecter au monde merveilleux de **The Pro Quidditch Manager**.

**coach** Un coach est un personnage de jeu membre d'une *équipe*, intervenant dans le cadre des *entraînements*. Il possède une série de caractéristiques, tout comme les *joueurs*. La valeur de ces caractéristiques influencera sur l'efficacité et l'apport d'un entraînement. Il n'intervient par contre aucunement dans un match de Quidditch.

**cognard** Balle de Quidditch qui peut être frappée par un *batteur*, pouvant éventuellement heurter les autres joueurs au cours d'un match.

**coup** Action qu'effectue un joueur. Il peut s'agir d'un mouvement (constitué de plusieurs déplacements), d'un sort, d'une interception de balle ou d'une frappe de la balle.

**entraînement** Action déclenchée par l'*utilisateur* permettant d'améliorer les *aptitudes* des joueurs. Un entraînement est divisé en groupes d'entraînements, spécifiés par l'*utilisateur*. Lancer un entraînement entraîne une augmentation de la fatigue des joueurs.

**gardien** Joueur qui doit rester dans la zone des buts et qui peut attraper ou dévier le *souaffle*.

**groupe d'entraînement** Ensemble de *joueurs*, d'*aptitudes* et de *coachs* qui intervient lors du déclenchement d'un *entraînement*. Cet ensemble détermine le degré d'amélioration des *aptitudes*.

**installation** Bâtiment/structure que peut posséder le stade d'une équipe.

**joueur** Un joueur est un personnage du jeu membre d'une *équipe*. Il participe aux matches de Quidditch selon ses caractéristiques et possède des *équipements*.

**manager** Personnage incarné par l'utilisateur dans le jeu. Il possède une *équipe*.

**membre** Désigne un joueur ou un coach d'une équipe.

**monde** Environnement virtuel du jeu. Il comprend, sans s'y limiter, les équipes des *utilisateurs* déconnectés, le marché des joueurs et des accessoires, la programmation des tournois et championnats.

**poursuiveur** Joueur pouvant attraper le *souaffle* et le lancer.

**serveur** Programme qui exécute le *monde*. Il accepte les connexions des *clients* et gère les matches et l'organisation des championnats.

**souaffle** Balle de Quidditch qui peut être attrapée, passée et lancée dans les goals par les *poursuiveurs* au cours d'un match.

**sponsor** Mécène qui accepte de financer une *équipe* sous certaines conditions (ratio de victoires, équipe possède un certain nombre de joueurs, ...).

**stade** Ensemble des installations appartenant à un *manager*.

**sélection d'équipe** L'ensemble des *joueurs* qui jouent sur le terrain (correspond à *squad* dans les diagrammes).

**tour** Ensemble de *coups* qu'effectuent les *joueurs* d'un *utilisateur* en une unité de temps.

**utilisateur** Personne utilisant un des programmes du projet The Pro Quidditch Manager.

**vif d'or** Une petite balle dorée qui vole de son propre gré aléatoirement sur tout le terrain. S'il est attrapé par un *attrapeur* le match prend fin.

**équipe** Un ensemble de *joueurs* et de *coachs* géré par un *manager*.

**équipement** Objet porté par les *joueurs* pendant les matches de Quidditch, influençant leurs aptitudes.

# Chapitre 2

## Besoins de l'utilisateur

### 2.1 | Exigences fonctionnelles

Cette section traite des exigences fonctionnelles du jeu. Elle est divisée en plusieurs sous-sections : la gestion de l'*équipe*, la gestion du stade, la gestion des entraînements, la gestion des transferts, le management de l'équipe, l'inscription aux championnats et aux matches amicaux et le jeu sur le terrain.

#### 2.1.1 | En bref

Ce jeu permettra à des utilisateurs du monde entier de créer une équipe de Quidditch et de la faire évoluer lors d'entraînements, de matches et de championnats en réseau. Chaque utilisateur démarera avec une équipe basique et un simple terrain dans son stade, et aura la possibilité d'ajouter des installations. Le jeu fournira un monde virtuel permettant d'engager des *coachs*, d'acheter et de vendre des *joueurs*, et de gagner des équipements et de l'argent, notamment avec les installations du stade. Les matchs seront joués en tour par tour, en suivant les règles officielles du Quidditch. Pour participer à un championnat, l'utilisateur devra s'y inscrire et jouera ensuite des matches fixés à une date prévue à l'avance.

Pour décrire plus particulièrement ces exigences fonctionnelles, nous utiliserons des diagrammes de *cas d'utilisation*. Notons que dans les cas d'utilisation qui vont suivre, les rubriques non spécifiées sont à prendre comme *Néant*.

#### 2.1.2 | Écran de démarrage

##### 2.1.2.1 Connexion (Login)

###### Précondition

Le programme est lancé.

###### Cas général

Deux champs de textes sont présentés à l'*utilisateur*. Il y entre son nom et son mot de passe. Quand les deux champs contiennent du texte, un bouton "envoi" apparaît. L'utilisateur clique dessus et le contenu des champs est envoyé au serveur de jeu.

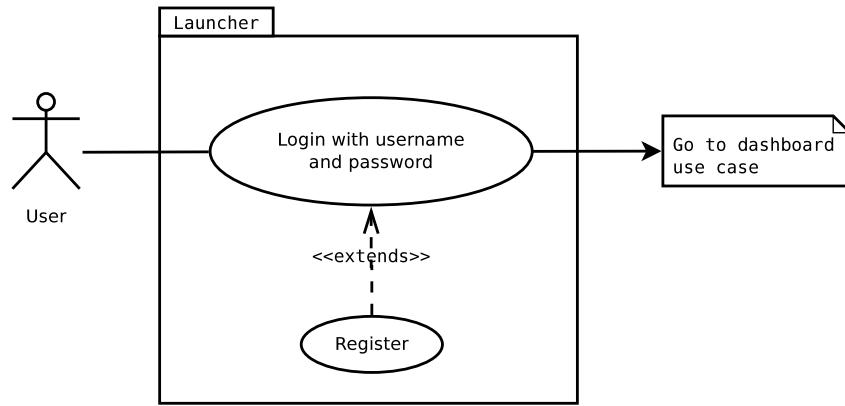


FIGURE 2.1 – Cas d'utilisation : écran de démarrage

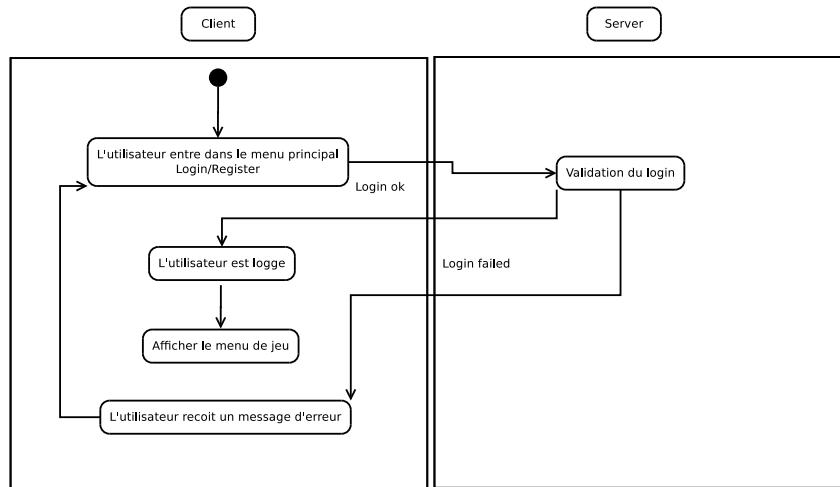


FIGURE 2.2 – Cas d'utilisation : Diagramme d'activité : écran de démarrage

### Cas exceptionnel

Un message d'erreur est affiché à l'utilisateur.

### Postcondition

L'utilisateur est connecté et arrive sur le dashboard.

#### **2.1.2.2   Créer un compte (Register)**

### Cas général

Étend le cas d'utilisation 2.1.2.1.

### Postcondition

Le compte est créé.

### 2.1.3 | Menu principal

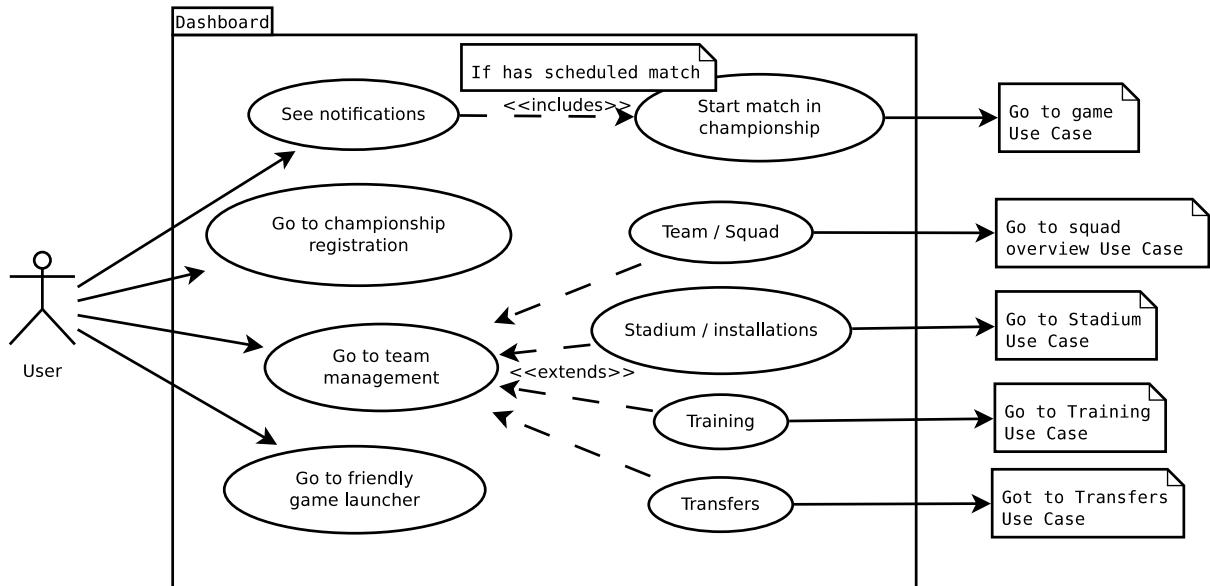


FIGURE 2.3 – Cas d'utilisation : dashboard

#### 2.1.3.1 Cas d'utilisation par défaut

##### Précondition

L'utilisateur est connecté.

##### Cas général

L'utilisateur clique sur un bouton, qui l'emmène dans une autre rubrique du jeu, dont il pourra éventuellement revenir à l'aide d'un bouton "retour".

#### 2.1.3.2 Voir les notifications (see notifications)

Étend le use-case 2.1.3.1

##### Cas général

Une zone de notifications est affichée à l'utilisateur, avec les récents changements qui le concernent. S'il doit jouer un match, un bouton lui permet d'entrer dans le match.

##### Postcondition

L'utilisateur arrive au cas d'utilisation de la phase de jeu.

#### 2.1.3.3 Autres

##### Postcondition

L'utilisateur arrive dans le cas d'utilisation décrit sur la figure 2.3.

## 2.1.4 | Gestion d'équipe

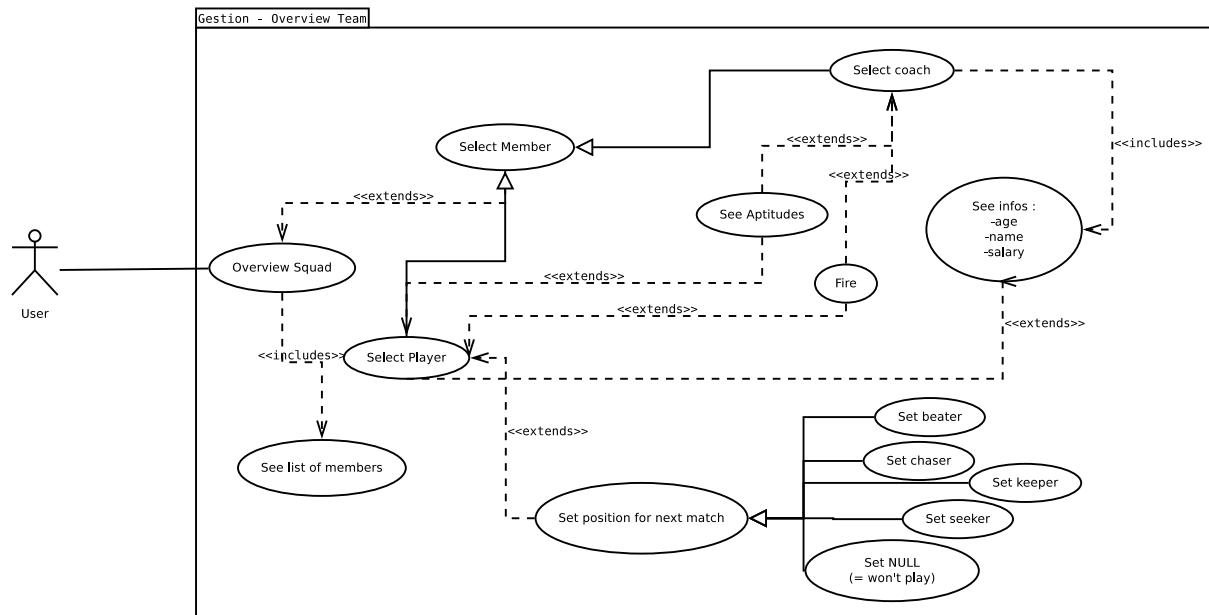


FIGURE 2.4 – Cas d'utilisation : gestion d'équipe

### 2.1.4.1 Gestion d'équipe (*overview squad*)

Relations avec d'autres cas d'utilisations

Étendu par ??.

Pré-conditions

— L'utilisateur est authentifié.

Cas général

L'utilisateur décide d'obtenir un rapide aperçu de son effectif, et clique sur l'icône correspondant à "gestion d'équipe", ce qui a pour effet d'afficher une liste de membres de l'équipe, avec laquelle il peut interagir pour obtenir des informations concernant lesdits membres.

Postconditions

Une liste de membres de l'équipe, englobant ainsi les joueurs et les coaches, est affichée.

### 2.1.4.2 Sélectionner un joueur (*select player*)

Relations avec d'autres cas d'utilisations

Spécialise ??.

Étendu par 2.1.4.3, ?? et 2.1.4.4.

### **Pré-conditions**

- L'utilisateur a sélectionné un membre, il s'agit d'un joueur.

### **Cas général**

Voir cas d'utilisations ??.

### **Post-conditions**

- De plus amples informations sont affichées concernant le joueur sélectionné (âge, nom salaire).
- Nouvelles actions disponibles.

#### ***2.1.4.3 Voir les aptitudes (see aptitudes)***

##### **Relations avec d'autres cas d'utilisations**

Étend ?? et 2.1.4.2.

### **Pré-conditions**

- L'utilisateur a sélectionné un membre.

### **Cas général**

Après avoir sélectionné un membre, l'utilisateur choisit d'afficher les aptitudes concernant ledit membre, en cliquant simplement sur une icône.

### **Post-conditions**

- Les aptitudes spécifiques au membre sélectionné sont affichées.

#### ***2.1.4.4 Choisir la position pour le prochain match (set position...)***

##### **Relations avec d'autres cas d'utilisations**

Étend 2.1.4.2.

### **Pré-conditions**

L'utilisateur a sélectionné un joueur.

### **Cas général**

La position choisie par l'utilisateur pour le joueur sélectionné n'est pas encore attribuée, c'est donc ce joueur qui remplira cette fonction tant qu'aucune modification n'est apportée par l'utilisateur.

## Cas exceptionnels

- La position est déjà attribuée, dans ce cas l'utilisateur en est averti et peut décider de remplacer le joueur actuel par le joueur sélectionné.
- L'utilisateur met la position à "NULL" ce qui signifie que, jusqu'à modification ultérieure, le joueur sélectionné ne jouera pas le prochain match.

## Post-condition

Le joueur sélectionné est mis à la position choisie par l'utilisateur pour le prochain match. Cette action est réversible jusqu'audit match.

### 2.1.5 | Gestion du stade

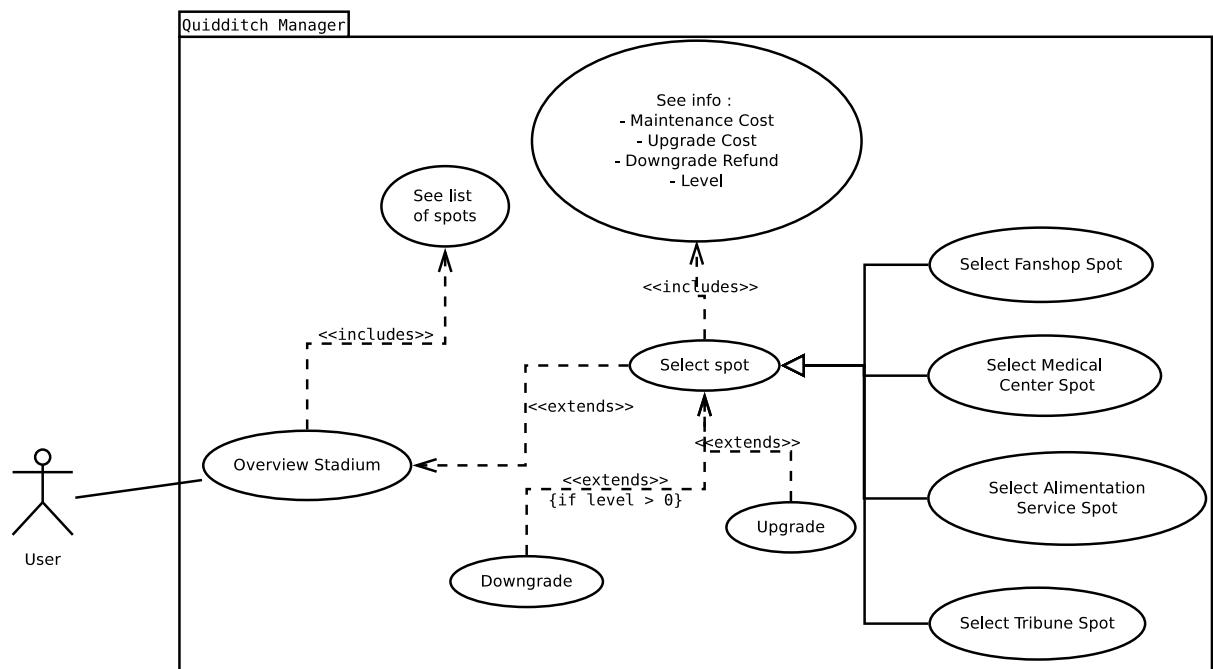


FIGURE 2.5 – Cas d'utilisation : gestion du stade

#### 2.1.5.1 Gestion du stade (overview stadium)

##### Relations avec d'autres cas d'utilisations

Étendu par ??.

##### Pré-conditions

- Le joueur est authentifié.

##### Cas général

L'utilisateur clique sur l'icône "gestion du stade", ce qui affiche une liste d'endroits chacun spécifique à une installation, avec laquelle il peut interagir.

### **Post-conditions**

- Une liste d'endroits spécifiques à une installation est affichée.

#### ***2.1.5.2 Améliorer / monter d'un niveau (upgrade)***

##### **Relations avec d'autres cas d'utilisations**

Étend ??.

### **Pré-conditions**

- Un endroit est sélectionné (peu importe lequel).
- Le niveau maximum de l'installation relative à l'endroit n'est pas atteint (à déterminer).
- L'équipe possède les fonds nécessaires.

### **Cas général**

L'utilisateur souhaite améliorer une installation, après avoir sélectionné l'endroit relatif à cette installation, il clique alors sur une icône "améliorer" ce qui augmente le niveau de l'installation de 1, et augmente ses bienfaits (argent gagné pour les services d'alimentations, fanshop ; nombre de places pour les tribunes ; soins prodigués pour le centre médical).

### **Post-conditions**

- L'installation gagne un niveau (meilleures stats mais coûts d'entretiens plus élevés).

#### ***2.1.5.3 Diminuer d'un niveau (downgrade)***

(Fonctionnalité non implémentée)

##### **Relations avec d'autres cas d'utilisations**

Étend ??.

### **Pré-conditions**

- Un endroit est sélectionné.
- Le niveau de l'installation est plus grand que 0.

### **Cas général**

L'utilisateur souhaite diminuer le niveau de son installation (pour mauvaise situation financière par exemple), il clique alors sur l'icône "dégrader", ce qui diminue le niveau de l'installation de 1.

### **Post-conditions**

Diminue le niveau de l'installation de 1 (équipe gagne un pourcentage des fonds déversés pour avoir atteint ce niveau, les coûts d'entretien diminuent, mais les stats baissent).

#### 2.1.5.4 Aller aux transferts (goto transfers)

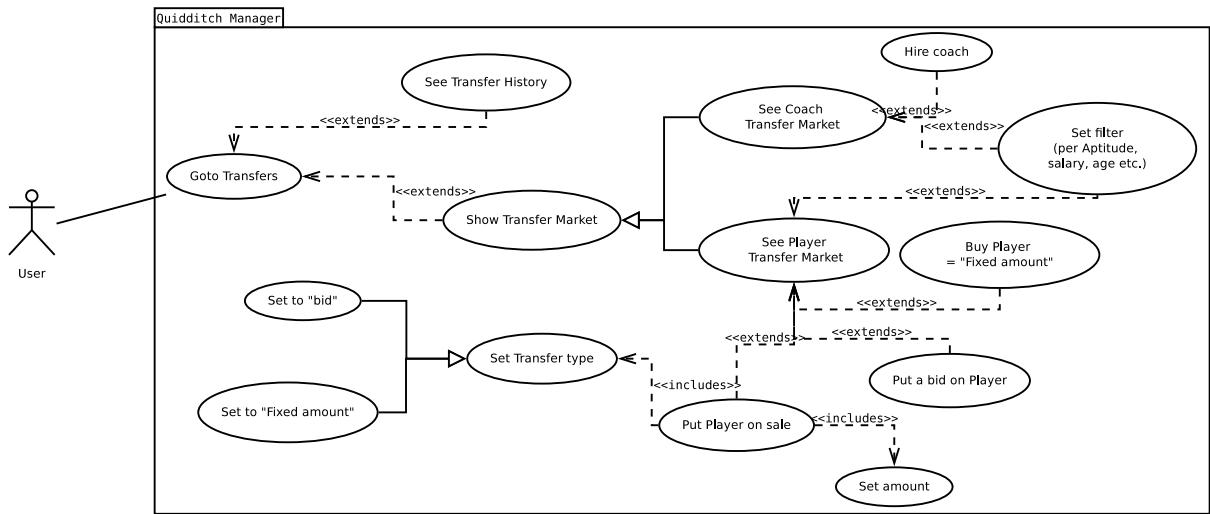


FIGURE 2.6 – Cas d'utilisation : aller aux transferts

#### Relations avec d'autres cas d'utilisations

Étendu par ?? et ??.

#### Pré-conditions

- L'utilisateur est authentifié.

#### Cas général

L'utilisateur clique sur l'icône "Aller aux Transferts", et a ensuite la possibilité de voir son historique de transferts ou de voir le marché des transferts.

#### 2.1.5.5 Acheter/mettre une offre sur un joueur (buy/put a bid on player)

#### Relations avec d'autres cas d'utilisations

#### Pré-conditions

- L'utilisateur se situe dans le marché des transferts des joueurs.
- Un joueur est sélectionné.
- L'équipe de l'utilisateur ne possède pas déjà le nombre maximal de joueurs que peut posséder une équipe.
- L'équipe de l'utilisateur possède les fonds nécessaires.
- Si enchère : l'utilisateur rentre l'enchère désirée qui doit être plus élevée que l'actuelle.

## Cas général

En recherchant des joueurs dans le marché des transferts des joueurs, l'utilisateur peut acheter ou mettre une enchère sur un joueur, dépendant du type d'achat dont il s'agit (type déterminé par l'équipe vendeuse lors de la mise en vente, voir 2.1.5.6). Si l'achat est un succès, le joueur est retiré du marché des transferts et ajouté à l'équipe de l'utilisateur. Si l'enchère est un succès, l'utilisateur remporte le joueur jusqu'à nouvelle enchère mise sur le joueur (notons que l'utilisateur ne peut enchérir sur un joueur sur lequel il a déjà enchéri).

## Post-conditions

- Si achat direct : l'équipe de l'utilisateur est constituée d'un joueur supplémentaire.
- Si vente "à l'enchère" : le nom de l'équipe de l'utilisateur est indiquée sur le joueur dans la liste du marché des transferts, signalant qu'il s'agit de l'équipe la plus offrante actuellement, et qui "remportera" le joueur si aucune autre enchère n'est mise avant la date d'expiration.

### **2.1.5.6 Mettre un joueur en vente (put player on sale)**

#### Relations avec d'autres cas d'utilisations

Étend ??.

## Pré-conditions

- L'utilisateur se situe sur le marché des transferts.
- L'utilisateur sélectionne un joueur.
- L'utilisateur choisit le type de transfert : enchère ou vente directe.
- Dans les deux cas l'utilisateur indique le prix (de base/de vente).
- L'utilisateur a au moins 7 joueurs dans son équipe qui ne sont pas en vente.

## Cas général

L'utilisateur, situé dans le marché de transferts des joueurs, choisit l'option de mettre un joueur en vente. Il sélectionne le joueur parmi tous les joueurs de son équipe, et indique le type de vente ainsi que le prix. Le joueur mis en vente apparaît alors dans le marché des transferts.

## Post-conditions

- Le joueur choisi est mis en vente avec les critères de l'utilisateur (type et prix).
- Ce joueur apparaît dans le marché de transferts des joueurs. Le temps qu'il y reste est à déterminer.

### **2.1.6 | Procédure d'enchère sur un joueur présent sur le Player Market**

- L'utilisateur choisit le joueur sur lequel il veut enchérir. Si ce joueur appartient à son équipe, il est automatiquement déconnecté de cette enchère. Si le joueur ne fait pas partie de son équipe il est autorisé à enchérir
- Si ce n'est pas le tour d'un utilisateur d'encherir, il est mis dans une file d'attente. Si c'est son tour, une étape de vérification de l'enchère est initialisée.

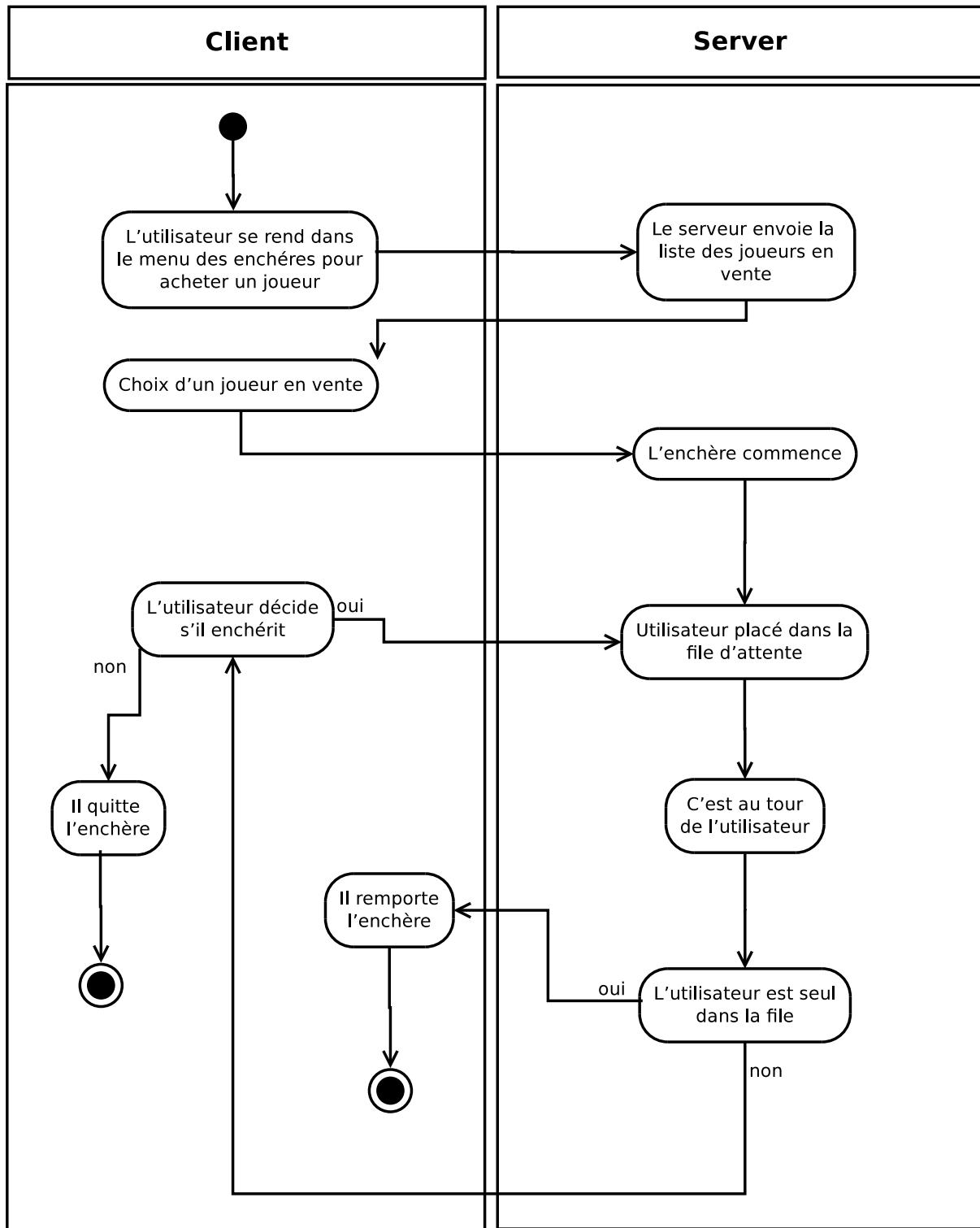


FIGURE 2.7 – Déroulement d'une enchère

- S'il a déjà la meilleure enchère il est mis dans une file d'attente, dans le cas contraire il lui est proposé de surenchérir. S'il refuse, il perd l'enchère.
- S'il augmente l'enchère il est mis dans une file d'attente, et est inclus dans le tour suivant.

- Si l'utilisateur est le seul restant dans l'enchère c'est lui qui gagne et ajoute le joueur à son lot. Dans le cas contraire un tour suivant d'enchère se déroule tant qu'il reste plus d'un utilisateurs participants.

### 2.1.7 | Le championnat

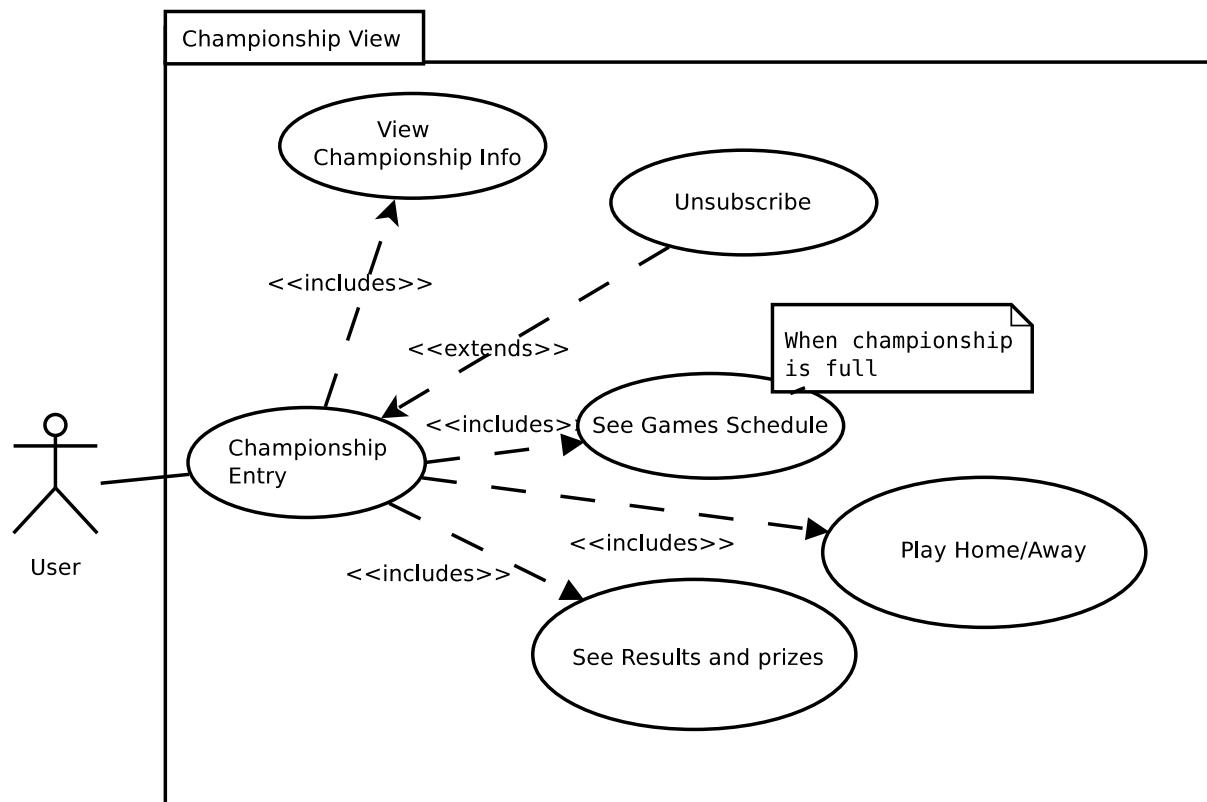


FIGURE 2.8 – Cas d'utilisation : championnat

#### 2.1.7.1 *Inscription à un championnat (championship entry)*

(Fonctionnalité non implémentée)

##### Précondition

L'utilisateur n'est pas inscrit à un autre championnat et le niveau de l'équipe de l'utilisateur est inclus entre les bornes de niveau définies par le championnat.

##### Cas général

L'utilisateur s'inscrit dans un championnat spécifique.

#### 2.1.7.2 *Vue des informations du championnat (championship info)*

##### Précondition

L'utilisateur est inscrit à ce championnat.

### Cas général

Le joueur a accès aux informations des équipes concurrentes (nom et niveau de l'équipe, listing des joueurs,...).

#### ***2.1.7.3 Désinscription d'une équipe (unsubscribe)***

##### Précondition

L'utilisateur est inscrit, le championnat n'est pas encore complet et n'a pas encore commencé.

### Cas général

L'utilisateur retire son équipe du championnat.

##### Postcondition

L'utilisateur ne participe plus au championnat et n'a plus accès à toutes les informations dudit championnat.

#### ***2.1.7.4 Réception de la grille des matches (games schedule)***

##### Précondition

L'utilisateur est inscrit au championnat et le nombre d'équipes est au complet.

### Cas général

Chaque équipe reçoit la grille des matches qui contient la date, l'heure et l'adversaire de chaque match.

#### ***2.1.7.5 Jeu des matches (play)***

##### Précondition

Le nombre d'équipes est au complet et la grille de matches a été distribuée.

### Cas général

Les utilisateurs disputent un match aller et un match retour l'un après l'autre.

### Cas exceptionnels

Un des utilisateurs n'est pas présent : l'ordinateur résout le jeu automatiquement.

##### Postcondition

Le vainqueur d'un match gagne 3 points au classement général, le perdant ne gagne aucun point, et les 2 équipes gagnent 1 point en cas de match nul. Les aptitudes des joueurs s'améliorent.

### 2.1.7.6 Réception des résultats (results)

Précondition

Tous les matches ont été joués.

Cas général

L'utilisateur voit le classement général des matches et reçoit éventuellement des prix, de l'expérience et de l'argent.

## 2.1.8 | Matches amicaux

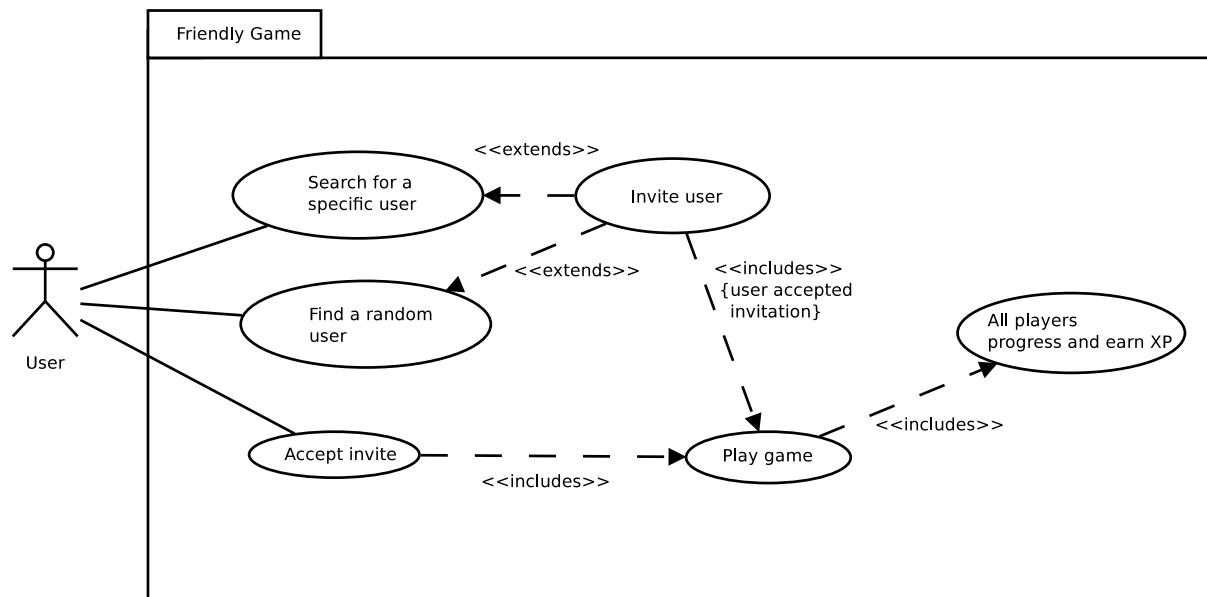


FIGURE 2.9 – Cas d'utilisation : matches amicaux

### 2.1.8.1 Recherche d'un utilisateur (search for spec. user)

Cas général

L'utilisateur peut entrer le nom d'un autre utilisateur et lancer sa recherche parmi tous les utilisateurs du jeu, connectés ou non.

Postcondition

Une liste des utilisateurs pouvant correspondre à la recherche est affichée.

### 2.1.8.2 Trouver un utilisateur aléatoire (find random user)

Cas général

L'utilisateur demande un autre utilisateur aléatoire, qui a plus ou moins son niveau et qui est connecté au jeu.

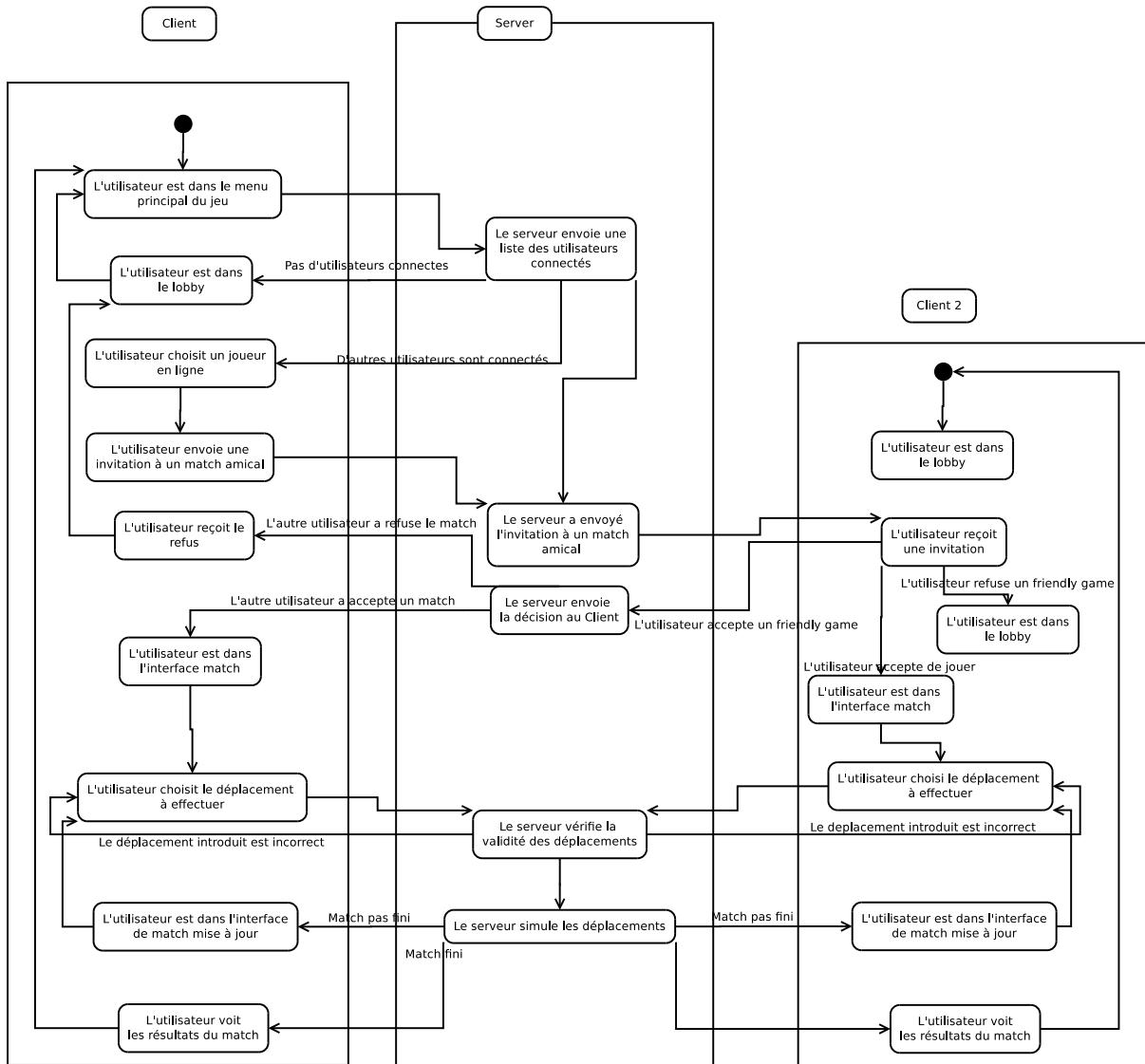


FIGURE 2.10 – Cas d'utilisation : diagramme d'activité : matches amicaux

### Postcondition

Un utilisateur aléatoire connecté, s'il y en a, est affiché.

#### 2.1.8.3 Inviter un utilisateur (*invite user*)

##### Cas général

L'utilisateur envoie une invitation à jouer un match amical à un autre utilisateur.

### Postcondition

L'invitation est en attente chez l'autre utilisateur.

#### 2.1.8.4 Accepter une invitation (*accept invite*)

## Précondition

Un autre utilisateur a invité l'utilisateur courant à jouer, et cet utilisateur est en ligne.

## Cas général

L'utilisateur accepte cette invitation.

### ***2.1.8.5 Jouer une partie (play game)***

## Précondition

Les deux utilisateurs ont accepté de jouer ensemble et sont tous les deux connectés.

## Cas général

La partie commence.

### ***2.1.8.6 Fin de partie***

## Précondition

Le jeu est terminé.

## Postcondition

Les joueurs des deux utilisateurs progressent.

## **2.1.9 | Phase de jeu**

Dans ce mode de jeu, l'utilisateur déplace ses joueurs sur un terrain où se déplacent les joueurs d'une autre équipe et des balles. L'utilisateur joue au tour par tour jusqu'à ce que le match soit fini. L'utilisateur doit voir une représentation du terrain de jeu, doit pouvoir y sélectionner un joueur et lui ordonner des actions comme un déplacement, un lancement de balle ou un lancement de sort. Lors de chaque tour, la vue du terrain est rafraîchie pour refléter l'état courant du match, les joueurs choisissent chacun simultanément les coups à jouer pour chacun de leurs joueurs, dans une limite de temps impartie, et chaque joueur ne pouvant se déplacer que de la distance qu'il peut parcourir en un tour. Quand le joueur a terminé de jouer, ou que le temps est dépassé, les coups à jouer sont envoyés au serveur, qui va calculer les déplacements, les points gagnés et les collisions qui interviennent durant ce tour.

### ***2.1.9.1 Sélection d'un joueur (select player)***

## Précondition

L'utilisateur est connecté au match et le temps de jeu pour ce tour n'est pas dépassé.

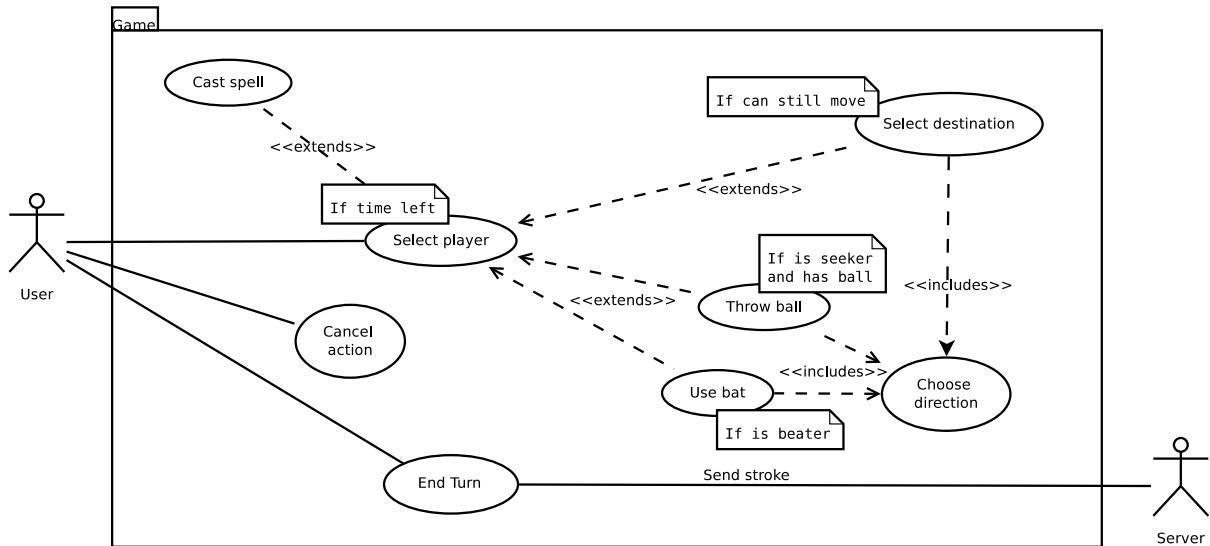


FIGURE 2.11 – Cas d'utilisation : match de Quidditch

### Cas général

L'utilisateur clique sur un joueur. L'ensemble des cases accessibles en ligne droite depuis la position du joueur est mis en évidence. Des icônes apparaissent selon les particularités du joueur : frapper la balle, attraper le vif d'or, lancer un sort d'amélioration, etc<sup>1</sup>...

### Cas exceptionnels

Le joueur n'appartient pas à la *sélection d'équipe* de l'utilisateur.

### Postcondition

le joueur est sélectionné.

#### 2.1.9.2 Sélection d'une direction (*choose direction*)

### Précondition

Un joueur est sélectionné ou une action du joueur sélectionné requiert une direction.

### Cas général

L'utilisateur clique sur une case accessible en ligne droite.

### Cas exceptionnel

La case de destination n'est pas accessible à cause des règles de l'action à effectuer.

---

1. L'ensemble des coups jouables sera déterminé plus loin dans le développement du jeu, à partir des constats sur l'équilibrage du jeu

## Postcondition

L'action est ajoutée à la liste des coups de l'utilisateur pour ce tour ; une icône est ajoutée au groupe d'icônes des actions de l'utilisateur pour ce tour.

### **2.1.9.3 Sélection d'une destination (select destination)**

#### Précondition

Un joueur est sélectionné.

#### Cas général

L'utilisateur clique sur une des cases accessibles en ligne droite (*mises en évidence au cas d'utilisation 2.1.9.1*) selon le cas d'utilisation 2.1.9.2.

#### Postcondition

Si le joueur continue un mouvement, ce déplacement est ajouté au mouvement du joueur. Sinon, le joueur commence un mouvement, et une icône est ajoutée au groupe d'icône des actions du tour.

### **2.1.9.4 Lancement d'une balle (throw ball)**

#### Précondition

Le joueur sélectionné est un *poursuiveur* et il a le *souaffle*.

#### Cas général

Continue selon le cas d'utilisation 2.1.9.2

### **2.1.9.5 Utiliser la batte (use batt)**

#### Précondition

Le joueur sélectionné est un *batteur*.

#### Cas général

Le joueur essaye de taper dans un *cognard*. Lorsque le mouvement sera effectivement joué, si le joueur est au même moment sur la même case que la balle, celle-ci partira dans la direction spécifiée, sinon le joueur tape dans le vent. Continue selon le cas d'utilisation 2.1.9.2 .

### **2.1.9.6 Annuler un coup (cancel action)**

#### Précondition

L'utilisateur a au moins joué un coup durant ce *tour*.

### Cas général

L'utilisateur effectue un clic droit sur une des icônes de coups joués.

### Postcondition

Le coup est retiré de la liste des coups pour ce tour.

#### ***2.1.9.7 Attraper le vif d'or (catch golden snitch)***

### Précondition

L'attrapeur est assez proche du vif d'or pour le voir et l'attraper (selon ses caractéristiques).

### Cas général

Se déroule comme au cas d'utilisation ??.

### Postcondition

Lorsque le tour est joué : si le sort réussit le match s'arrête et l'équipe du joueur remporte 150 points.

#### ***2.1.9.8 Terminer son tour volontairement (end turn)***

### Précondition

Le temps de jeu pour ce tour n'est pas dépassé.

### Cas général

L'utilisateur clique sur le bouton "terminer le tour". Continue selon le cas d'utilisation 2.1.9.10.

#### ***2.1.9.9 Fin de temps de jeu***

### Précondition

Le temps de jeu imparti pour ce tour est dépassé.

### Cas général

Continue selon le cas d'utilisation 2.1.9.10.

#### ***2.1.9.10 Terminer un tour***

### Cas général

La liste des coups pour l'utilisateur est envoyée au serveur et les icônes qui les représentent disparaissent. Si les deux utilisateurs terminent leurs tours avant la fin du temps imparti, le tour est exécuté directement.

## Postcondition

Tous les coups sont confirmés et l'utilisateur ne peut plus les modifier. L'utilisateur attend que le tour de son adversaire soit lui aussi terminé. Les coups sont joués sur le serveur et l'utilisateur voit le nouvel état de jeu.

### 2.1.10 | Cas d'utilisation indirects

Ces cas d'utilisation décrivent comment évolue une partie de jeu si certaines actions des joueurs ont lieu, mais sur lesquelles l'utilisateur n'a pas influence directe.

#### 2.1.10.1 Attraper le souaffle

##### Précondition

Un *poursuiveur* et le *souaffle* sont sur une même case à un instant donné.

##### Cas général

Selon les caractéristiques du joueur, la probabilité que la balle soit attrapée est plus ou moins élevée. Un tir aléatoire détermine si le joueur l'attrape ou non.

##### Postcondition

Si le tir aléatoire a déterminé que le joueur attrape la balle, il l'a en main et peut la lancer.

#### 2.1.10.2 Marquer un but

##### Précondition

Un *poursuiveur* a lancé le souaffle et celui-ci passe par une case goal de l'*équipe* adverse.

##### Cas général

La balle est placée une case derrière le goal, à l'arrêt.

##### Postcondition

L'équipe qui ne possède pas la case goal à travers laquelle le souaffle est passé reçoit 10 points.

#### 2.1.10.3 Sortir la balle du terrain

##### Précondition

Une balle dépasse la limite du terrain.

##### Postcondition

La balle est arrêtée à la dernière position occupée sur le terrain.

## 2.2 | Exigences non fonctionnelles

Cette section traite des exigences non fonctionnelles des besoins de l'utilisateur. Ces exigences peuvent être ramenées à la liste suivante :

- L'utilisateur souhaite une belle interface graphique ;
- La latence lors des matches ne doit pas être trop élevée (pas plus d'une seconde) ;
- La phase de jeu doit être rapide et réactive, le temps d'attente de chaque tour ne doit pas être trop long, de manière à ce qu'un des deux utilisateurs ne puisse pas bloquer le jeu ;
- L'attente entre l'inscription à un championnat et les premiers matches ne doit pas être plus longue qu'un ou deux jours ;
- Dans les championnats, le niveau entre les équipes doit être relativement proche ;
- L'utilisateur souhaite interagir le plus possible avec les autres utilisateurs, étant donné qu'il s'agit d'un jeu en ligne ;
- Le monde doit évoluer, c'est-à-dire que l'utilisateur doit remarquer une différence dans son expérience de jeu au fur et à mesure que son équipe s'améliore ;
- Le monde doit être varié, c'est-à-dire que l'utilisateur s'attend à une certaine variété d'équipements, de joueurs, de coaches, d'installations, etc.

## 2.3 | Exigences de domaine

- Le jeu cible principalement les adolescents et fans de l'univers de la saga *Harry Potter*, il doit donc être facile à apprêhender et fidèle à l'idée qu'aurait un fan du Quidditch ;
- Le jeu doit s'installer facilement, pour pouvoir toucher un grand public.

# Chapitre 3

## Besoins du système

### 3.1 | Exigences fonctionnelles

Cette section décrira ce que le système doit être capable de faire. À nouveau, nous employons des cas d'utilisations pour décrire particulièrement les caractéristiques du système.

#### 3.1.1 | Système - partie gestion

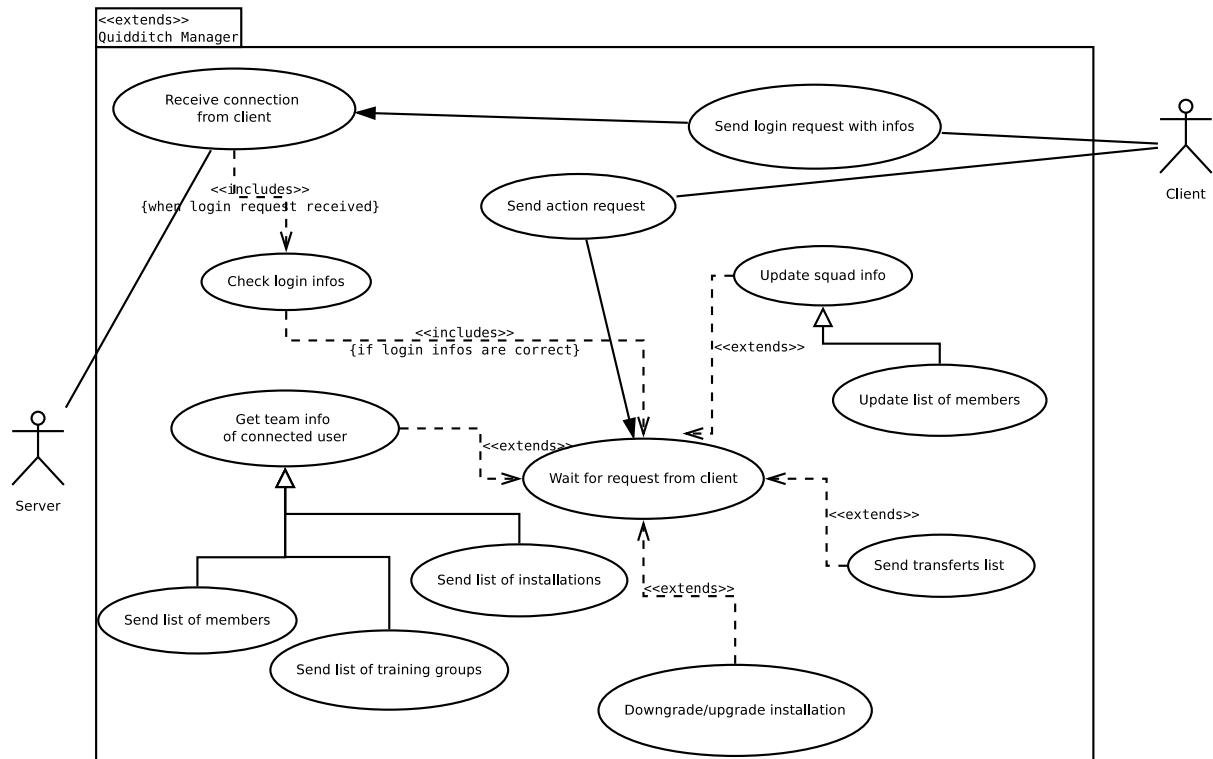


FIGURE 3.1 – Cas d'utilisation : Système - partie gestion

##### 3.1.1.1 Connexion d'un utilisateur (*connection from client*)

## Relations avec d'autres cas d'utilisation

Inclut 3.1.1.2.

### Précondition

Le *serveur* est allumé.

### Cas général

Le serveur reçoit les informations de connexion d'un utilisateur.

#### **3.1.1.2 Vérification des informations de connexion (check login info)**

## Relations avec d'autres cas d'utilisation

Est inclus par 3.1.1.1.

Inclut 3.1.1.3.

### Cas général

Le couple Nom d'utilisateur - Mot de passe est correct.

### Cas exceptionnel

Le couple Nom d'utilisateur - Mot de passe est incorrect.

### Postcondition

L'utilisateur est connecté.

#### **3.1.1.3 Attente des requêtes du client (wait for request)**

## Relations avec d'autres cas d'utilisation

Est inclus par 3.1.1.2. Est étendu par 3.1.1.4, 3.1.1.5, 3.1.1.6, 3.1.1.7, 3.1.1.8.

### Précondition

L'utilisateur est connecté.

### Cas général

Le *serveur* reçoit des messages reconnaissables du *client* et y répond en envoyant des données ou en commençant une action.

### Cas exceptionnel

Le serveur ne reçoit pas de message du client ou reçoit un message qui n'est pas reconnaissable.

## Postcondition

Le serveur a envoyé les informations nécessaires, ou a stocké les informations envoyées par le client et continue à attendre des messages du client.

### ***3.1.1.4 Mise à jour des informations de l'équipe (update squad info)***

#### Relations avec d'autres cas d'utilisation

Étend 3.1.1.3.

## Précondition

L'utilisateur a envoyé une liste de changements à faire dans l'équipe qu'il possède.

### Cas général

Le *serveur* met à jour sa copie des données à propos de l'équipe de l'utilisateur.

## Postcondition

L'équipe du joueur est à jour sur le serveur et le serveur continue à attendre des messages du *client*.

### ***3.1.1.5 Mise à jour des informations d'une installation (installation attribute)***

#### Relations avec d'autres cas d'utilisation

Étend 3.1.1.3.

## Précondition

L'utilisateur a changé les informations d'une installation qu'il possède.

### Cas général

Le *serveur* met à jour sa copie des données à propos de l'installation.

## Postcondition

Les données de l'installation du joueur est à jour sur le serveur et le serveur continue à attendre des messages du *client*.

### ***3.1.1.6 Envoi de la liste des joueurs disponibles pour un transfert (transfers list)***

#### Relations avec d'autres cas d'utilisation

Étend 3.1.1.3.

### **Précondition**

L'utilisateur a demandé à voir la liste des joueurs qu'il peut acheter.

### **Cas général**

Le *serveur* envoie cette liste.

### **Postcondition**

Le *client* a reçu la liste et le serveur continue à attendre des messages du client.

#### ***3.1.1.7 Envoi de la liste du matériel disponible à l'achat (list of gears)***

##### **Relations avec d'autres cas d'utilisation**

Étend 3.1.1.3.

### **Précondition**

L'utilisateur a demandé à voir la liste du matériel qu'il peut acheter.

### **Cas général**

Le *serveur* envoie cette liste.

### **Postcondition**

Le *client* a reçu la liste et le serveur continue à attendre des messages du client.

#### ***3.1.1.8 Envoi des informations liées à l'utilisateur connecté (get team info)***

##### **Relations avec d'autres cas d'utilisation**

Étend 3.1.1.3.

### **Précondition**

L'utilisateur a demandé à recevoir la liste de ses joueurs, de ses installations ou de ses groupes d'entraînement, par exemple lorsqu'il désire avoir un aperçu de son stade.

### **Cas général**

Le *serveur* envoie la liste demandée.

### **Postcondition**

Le *client* a reçu la liste et le serveur continue à attendre des messages du client.

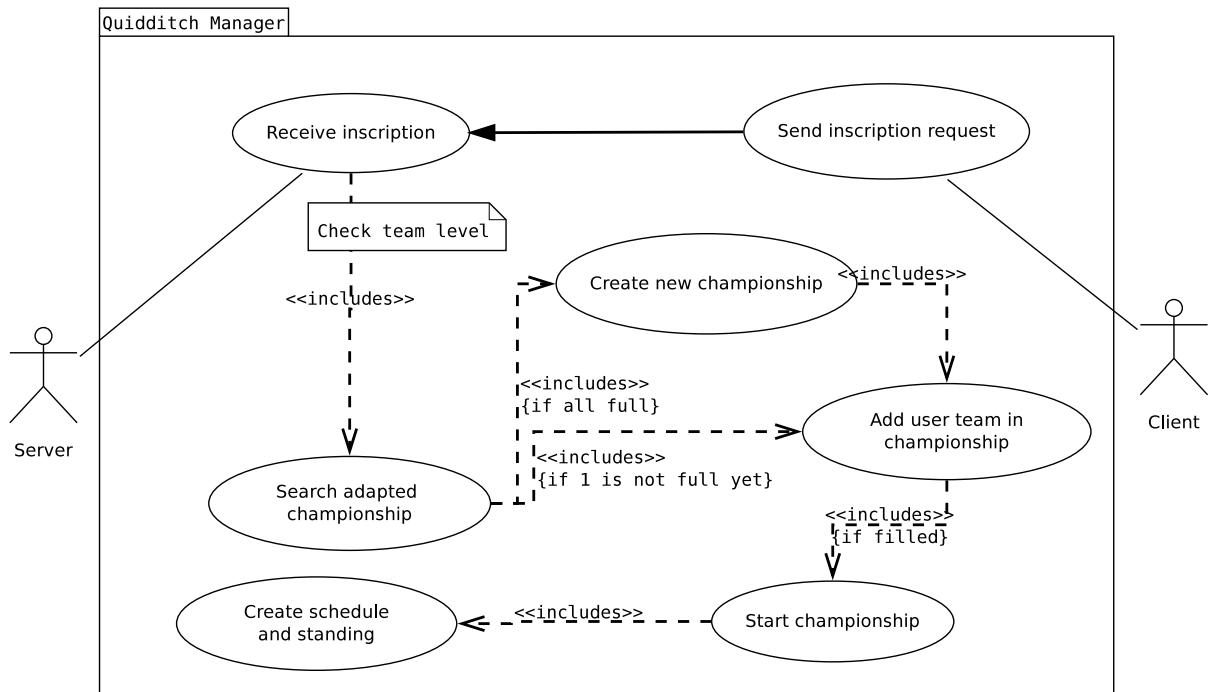


FIGURE 3.2 – Cas d'utilisation : Système - championnat

### 3.1.2 | Cas d'utilisation du championnat

#### 3.1.2.1 *Recevoir une inscription (receive inscription)*

**Relations avec d'autres cas d'utilisation**

Reçoit un message de 3.1.2.2.

Inclut 3.1.2.3.

**Pré-conditions**

- Le serveur est allumé.

**Cas général**

Le serveur reçoit une demande d'inscription à un championnat, cette demande venant d'un client (utilisateur).

**Post-conditions**

- Une demande d'inscription doit être traitée.

#### 3.1.2.2 *Envoyer une requête d'inscription (send inscription request)*

**Relations avec d'autres cas d'utilisation**

Envoie un message à 3.1.2.1.

## Cas général

### Post-conditions

- Une requête d'inscription à un championnat est envoyée au serveur.

### ***3.1.2.3 Rechercher un championnat approprié (search adapted championship)***

#### Relations avec d'autres cas d'utilisation

Inclus par 3.1.2.1.

Inclut 3.1.2.4 et 3.1.2.5.

### Pré-conditions

- Une demande d'inscription a été reçue.

## Cas général

Le serveur effectue une recherche parmi tous les championnats dont le niveau est plus ou moins équivalent à celui de l'équipe de l'utilisateur effectuant la demande.

### ***3.1.2.4 Créer un nouveau championnat (create new championship)***

#### Relations avec d'autres cas d'utilisation

Inclus par 3.1.2.3.

Inclut 3.1.2.5.

### Pré-conditions

- Un championnat adéquat a été recherché.
- Aucun championnat adéquat libre (qui n'est pas plein donc).

## Cas général

Le serveur n'a pas trouvé de championnat correspondant au niveau de l'équipe effectuant la demande d'inscription, il en crée donc un ayant ce niveau.

### Post-conditions

- Un nouveau championnat dont le niveau est mis à celui de l'équipe effectuant la demande d'inscription est créé.

### ***3.1.2.5 Ajouter l'équipe dans le championnat (add user team in championship)***

#### Relations avec d'autres cas d'utilisation

Est inclus par 3.1.2.3 et 3.1.2.4.

Inclut 3.1.2.6.

## Pré-conditions

- Un championnat a été trouvé ou créé.

## Cas général

L'équipe recherchant un championnat adéquat est ajouté au championnat trouvé.

## Post-conditions

- Le championnat trouvé possède une équipe en plus.

### *3.1.2.6 Démarrer le championnat (start championship)*

#### Relations avec d'autres cas d'utilisation

Est inclus par 3.1.2.5.

Inclut 3.1.2.7.

## Pré-conditions

- L'équipe précédemment ajoutée au championnat a complété ce dernier (le championnat est donc plein).

## Cas général

L'équipe qui recherchait un championnat adéquat est ajouté à celui-ci. Si cette équipe remplit le dernier emplacement libre de ce championnat, le championnat peut démarrer.

## Post-conditions

- Le championnat est démarré.

### *3.1.2.7 Crée programme des matches et le classement (create scheduling and starting)*

#### Relations avec d'autres cas d'utilisation

Inclus par 3.1.2.6.

## Pré-conditions

- Un nouveau championnat démarre.

## Cas général

Lors du démarrage d'un nouveau championnat, le serveur crée un programme des matches (dates, adversaires) et un classement regroupant les équipes du championnat et leurs points respectifs.

## Post-conditions

- Un programme des matches et un classement est créé pour le championnat.

## 3.2 | Exigences non fonctionnelles

Les exigences non fonctionnelles concernant les besoins du système peuvent être ramenées à la liste suivante :

- L'utilisateur ne peut pas modifier les données de son équipe (ce qui revient à tricher) ;
- L'ordinateur de l'*utilisateur* doit être assez puissant pour que le *client* s'exécute correctement.
- Le *serveur* doit pouvoir gérer toutes les connexions et tous les matches en instance ;
- Le système fonctionnera sur tous les systèmes compatibles avec la norme POSIX.

## 3.3 | Design et fonctionnement du système

Le système fonctionne selon le paradigme *client-serveur*. Lorsqu'un client essaie de se connecter au serveur, celui-ci crée un fil d'exécution spécifique au client. Le client se connecte et entre dans la partie gestion de l'application. Lorsqu'un match démarre pour le client, le serveur crée un nouveau processus pour le match séparé en trois fils d'exécution, un pour chaque client jouant le match et un pour la gestion du match. A la fin du match, ce processus est tué et les résultats du match sont renvoyés au processus principal. Le système sera programmé en C++, avec l'aide de la bibliothèque standard et d'une bibliothèque pour la partie graphique. Ce dernier point doit encore être discuté avec les clients et l'équipe technique.

### 3.3.1 | Composants

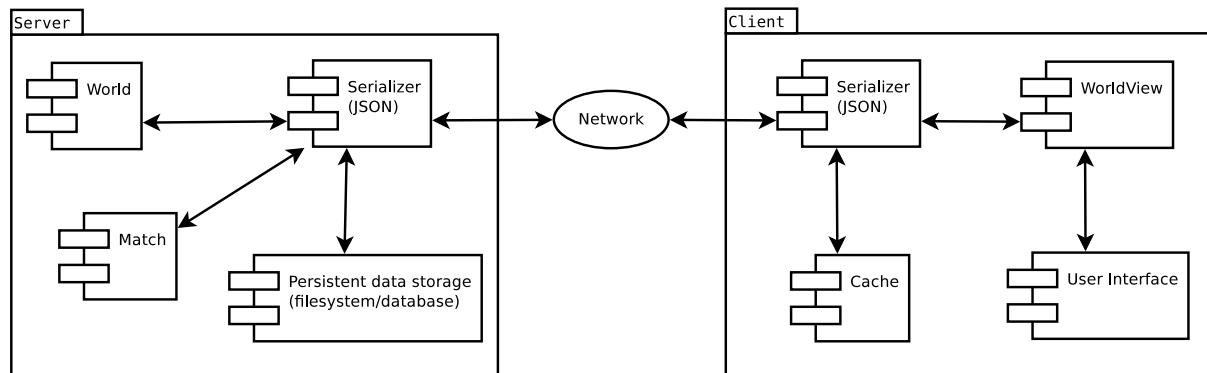


FIGURE 3.3 – Composants

Les différents use cases présentés antérieurement seront implémentés grâce aux composants décrits à la figure 3.3

### 3.3.2 | Envoi des messages

Pour l'envoi de données sur le réseau et la persistance des données, nous avons envisagé une seule et même solution : la sérialisation d'objets JSON (illustrée à la figure 3.4) (JavaScript Object Notation). A l'aide d'un outil de sérialisation/déserialisation JSON de notre cru, nous

allons transformer les objets à envoyer ou stocker en objets JSON. Ces objets JSON possèderont une méthode *toString* pour en faire une chaîne de caractères stockable dans un fichier texte ou envoyable sur le réseau.

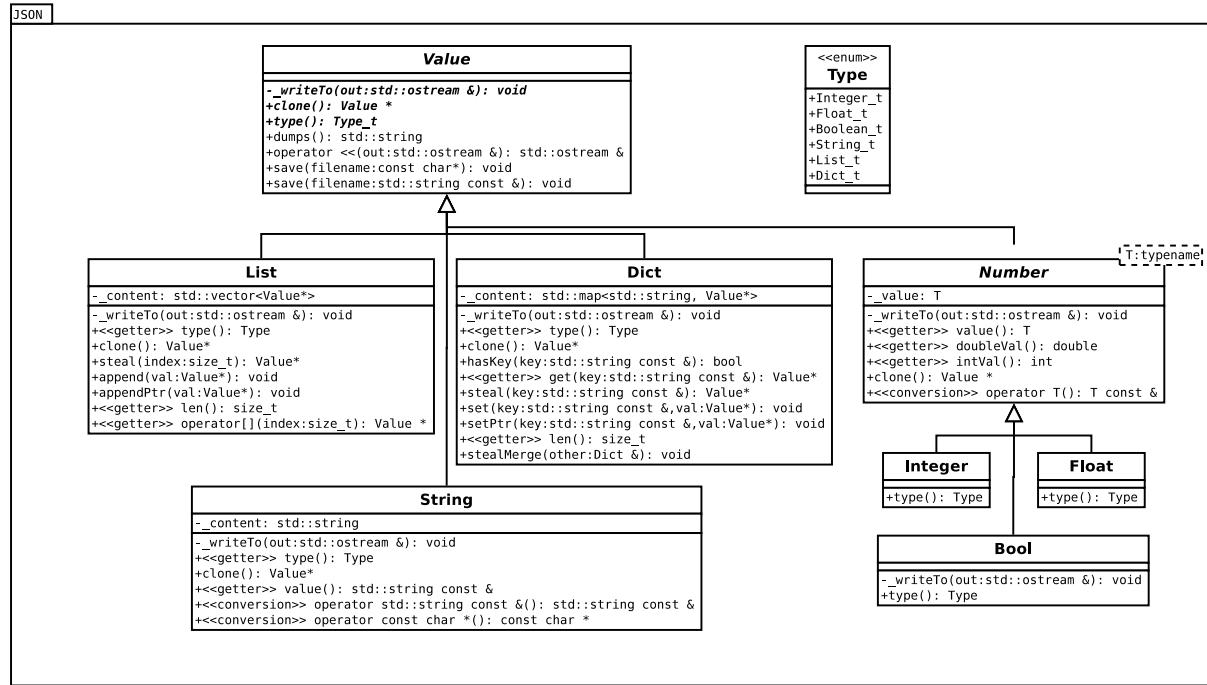


FIGURE 3.4 – Classes : outil de sérialisation JSON

### 3.3.3 | Phase de jeu

La figure 3.5 illustre les messages envoyés entre le *client* et le *serveur* lors d'une phase de jeu. Le serveur possède un processus dédié pour chaque match en cours. C'est de ce processus qu'il est question dans cette section. Le client indique d'abord qu'il est présent, ce qui permet au serveur de savoir si une résolution automatique doit avoir lieu. Ensuite le serveur envoie les informations du match au client. Dans le cas d'une résolution automatique, le serveur envoie les résultats du match. Quand le client a chargé le match, il indique qu'il est prêt et le jeu commence. Lors de chaque tour, le client envoie les coups joués par l'utilisateur et le serveur renvoie aux deux clients les mises à jour de l'état de la partie. Un message spécial du serveur lors de la mise à jour indique que la partie est terminée et le serveur envoie alors les résultats du match dans un autre message. Quand le client les a reçues, il le signifie au serveur qui ferme alors la connexion.

### 3.3.4 | Le gestionnaire de connexion

Le gestionnaire de connexion présente une queue d'entrée et une queue de sortie permettant d'envoyer ou recevoir des messages via des threads séparés. Chaque message contient un objet JSON et l'identifiant du pair qui l'a envoyé. Chaque pair a un ID unique.

Le BaseConnectionManager est la classe de base des gestionnaires de connexion. Sa responsabilité est de maintenir un ensemble de descripteurs de fichiers, d'y lire un message dès qu'il en arrive, et d'y écrire les messages sortants. Si le pair associé à un des FD ferme la connexion, le BaseConnectionManager le retire de la liste des FD gérés (il n'est donc plus disponible pour lecture) et émet un message dans la queue de sortie.

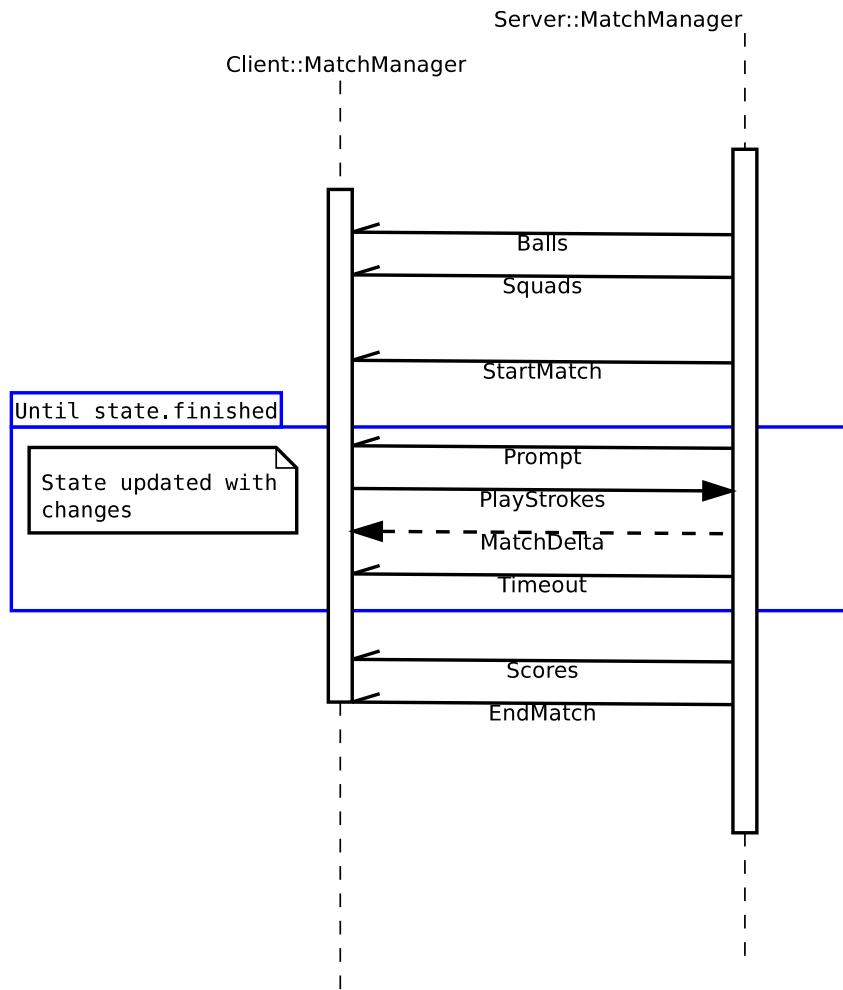


FIGURE 3.5 – Séquence : le jeu

Le ConnectionManager est comme BaseConnectionManager, mais accepte les connexion entrantes (appels système *bind* et *connect*). Un objet de cette classe est initialisé avec les queues d’entrée/sortie, ainsi qu’avec une adresse IP, un port et le nombre maximum de clients à accepter. Lors de la construction de l’objet, une socket en listen est créée. Lors de chaque nouvelle connection, un message est émis dans la queue d’entrée.

Un SubConnectionManager est un BaseConnectionManager qui emprunte des clients à un autre BaseConnectionManager (le parent). Lorsqu'il est détruit, il rend automatiquement les clients à son parent. Cette classe permet de créer des sous-connexions (par exemple lors d'un match) avec des queues d'entrée/ sortie séparées.

#### 3.3.4.1 Les gestionnaires de jeu

#### 3.3.4.2 Le gestionnaire de match

Le gestionnaire de match comprend les quatre balles, les deux équipes et les coups du tour actuel. Il gère tout ce qui concerne le match. Il s’occupe de l’échange des messages préliminaires et des résultats. Lors de chaque tour : côté client, il récolte les coups pour les deux équipes et puis les envoie au *serveur*, côté serveur, il résout les coups simultanément et renvoie le résultat au *client*.

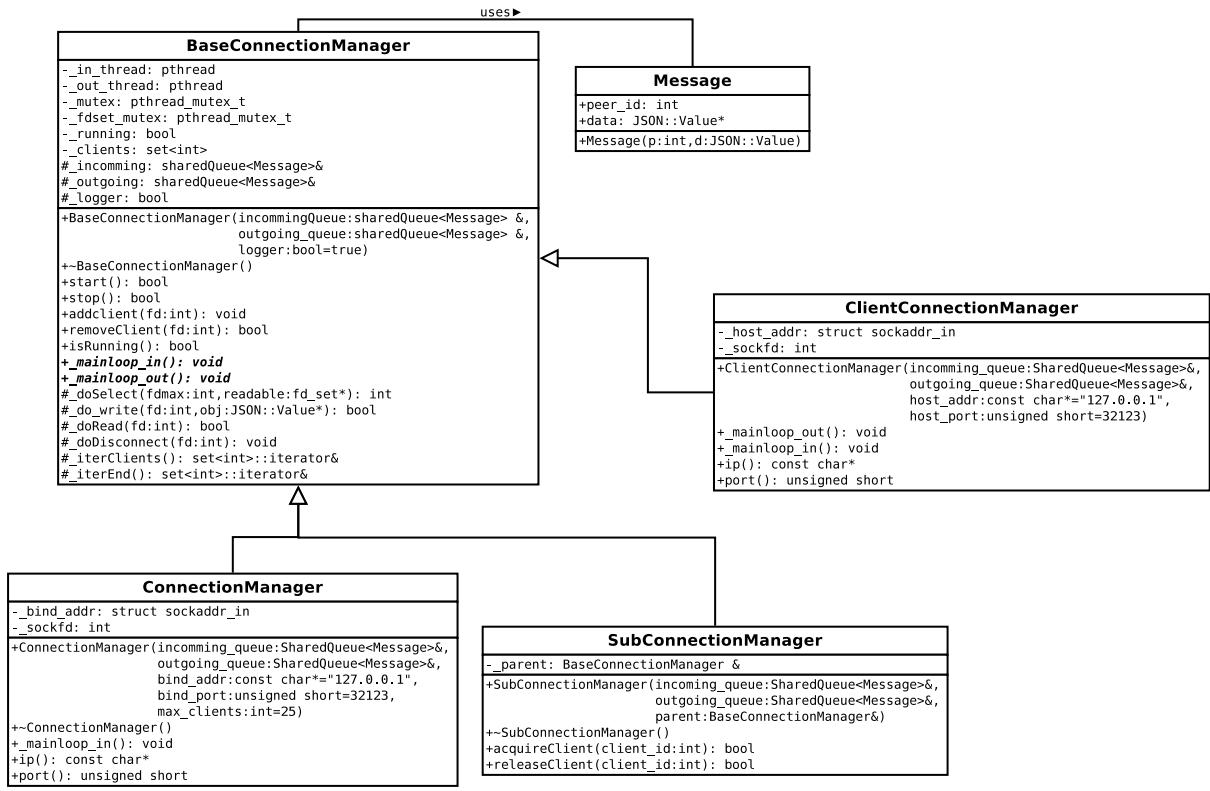


FIGURE 3.6 – Classes : Le gestionnaire de connexion

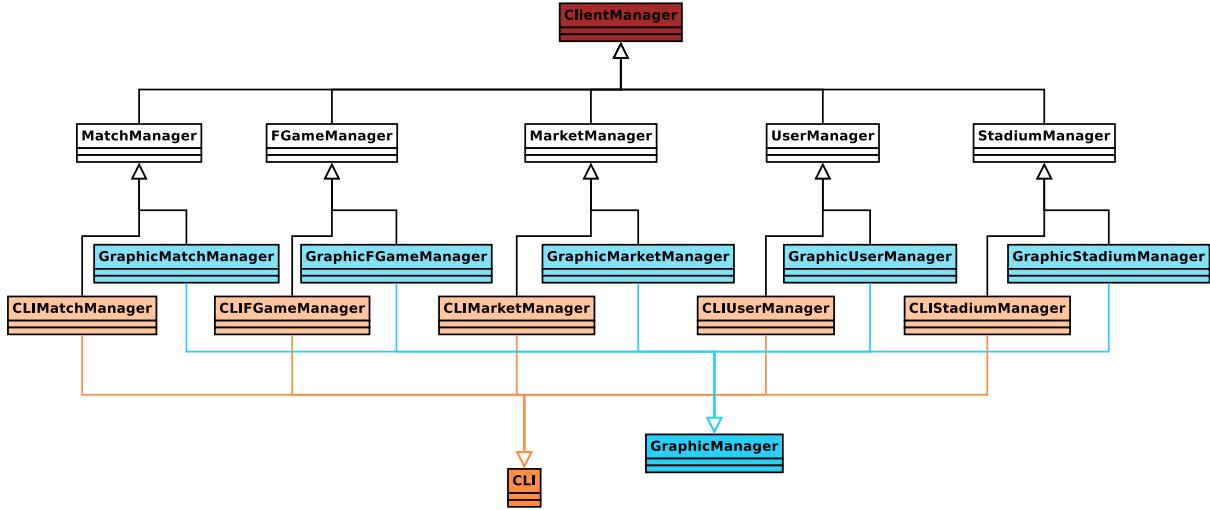


FIGURE 3.7 – Classes : Les gestionnaires de jeu du client

### 3.3.4.3 Les objets déplaçables du match

Tous les objets déplaçables dans un match (joueurs et balles) proviennent d'une même interface **Moveable**. Les **Moveable** possèdent une vitesse et une position. La vitesse correspond au nombre de cases parcourables en un tour et la position est un couple  $(x, y)$  dans un repère cartésien. Les directions sont des vecteurs constants représentant un déplacement d'1 pied dans une des directions possibles.

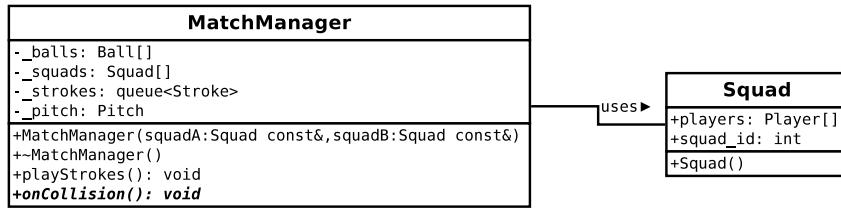


FIGURE 3.8 – Classes : le gestionnaire de match

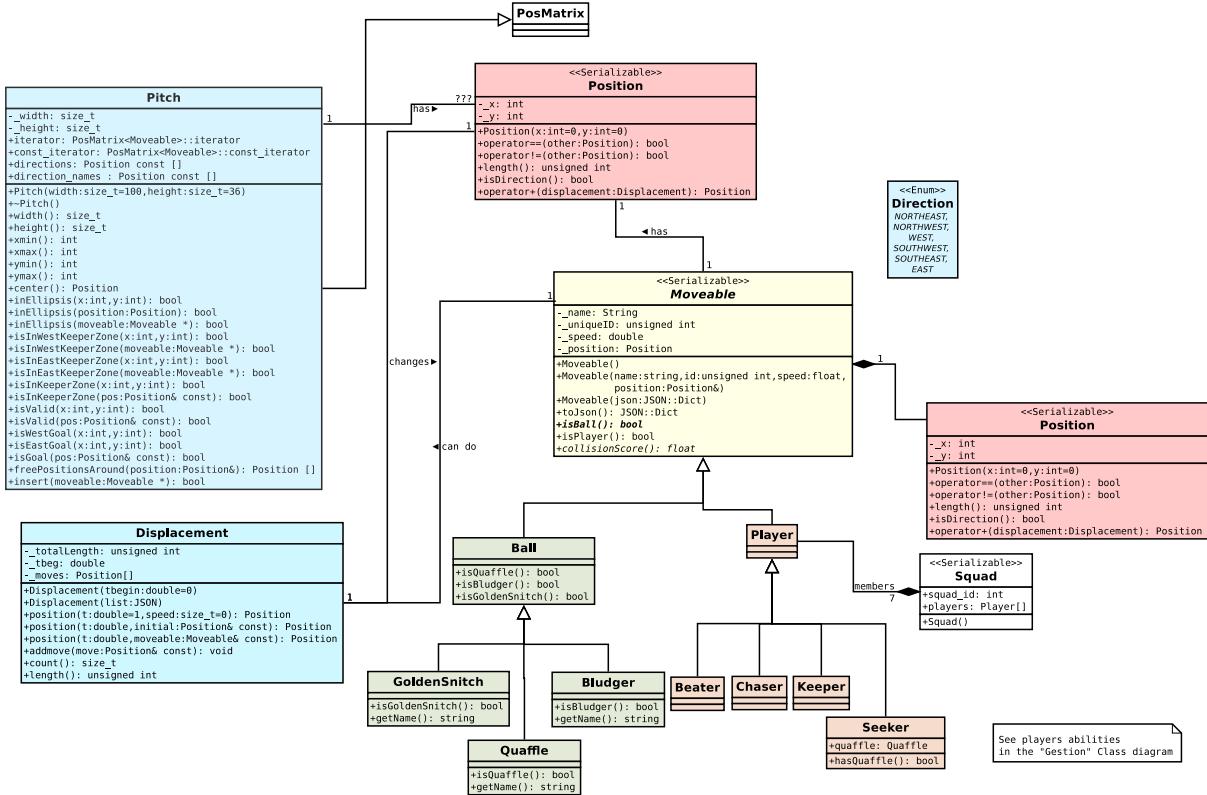


FIGURE 3.9 – Classes : les objets déplaçables du match

### 3.3.4.4 Les actions des joueurs

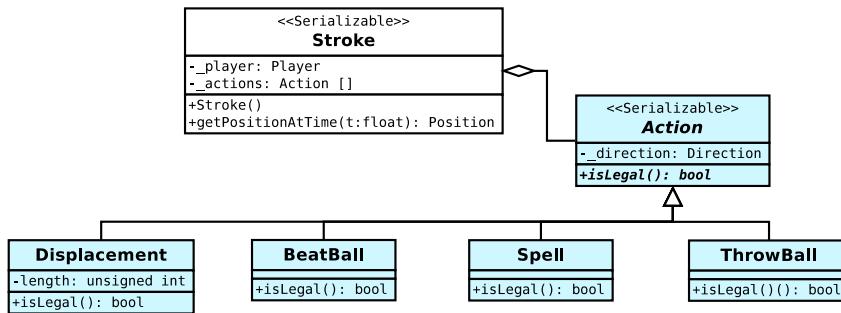


FIGURE 3.10 – Classes : les actions des joueurs

Les actions sont au nombre de quatre : déplacement, coup de batte, sort et lancer du Souaffle. On peut vérifier si une action est logique. Chaque action correspond à un coup qui est envoyé

au *serveur* et qui peut être décomposé pour permettre la "simultanéité" des événements.

### 3.3.5 | Phase de gestion

Un *manager* contrôle une *équipe*, c'est-à-dire des *membres* (*joueurs* et *coach*), des *installations* dans son *stade* et des *sponsors*. Chaque équipe a également des fonds, une réputation, un nombre matchs gagné et un nombre de matchs total. Les sponsors rapportent de l'argent à chaque match et nécessitent un minimum de réputation de l'équipe pour y être liée. Les installations coûtent de l'argent à chaque intervalle de temps et rapporte de l'argent lors de chaque match. Les membres ont un salaire, qui est déduit à chaque intervalle de temps des fonds de l'équipe et un prix d'achat sur le marché des transferts. Les joueurs possèdent une barre de vie qui diminue lors de blessures en match et une barre de mana qui diminue en lançant des sorts. Ils ont également des aptitudes qui varient selon leur position de jeu favorite. Les coachs possèdent des aptitudes qui permettent d'améliorer les aptitudes des joueurs.

#### 3.3.5.1 *Les championnats*

Le championnat se joue avec un système de "poule" de huit équipes. Chaque *sélection d'équipe* joue un match aller-retour contre toutes les autres équipes. Une victoire rapporte trois points à l'équipe victorieuse, un match nul un point pour les deux équipes et une défaite ne rapporte aucun point. Au terme du championnat, nous obtenons un classement général en fonction des points de toutes les équipes (cfr. classe Ranking). Si on se trouve face à un cas d'égalité, on départage les équipes en fonction du goal average.

Une fois que le championnat commence, le serveur envoie la grille de matches à toutes les équipes. Cette grille de matches est gérée par la classe Schedule qui connaît tous les matches à jouer et les matches joués, y compris leur score.

### 3.3.6 | Terrain

Un terrain de Quidditch est délimité par un ovale, et est constitué de cases hexagonales de 10 pieds<sup>1</sup>. Il fait 50 cases dans sa plus grande longueur, et 18 cases dans sa plus grande largeur, pour un total de 706 cases sur le terrain.

Chaque case ne peut accueillir qu'un seul joueur au maximum. Si deux joueurs arrivent au même tour sur la même case, une collision a lieu. Selon les *aptitudes* et les rôles des joueurs, différentes actions indirectes pourront avoir lieu.

#### 3.3.6.1 *Déplacement des joueurs*

Un tour de jeu est divisé en  $n$  intervalles isochroniques, avec  $n$  la plus grande vitesse parmi tous les objets déplaçables. La figure 3.13 illustre la procédure utilisée. Pour chaque instant, de  $t = \frac{1}{n}$  à  $t = 1$ , pour chaque déplacement à jouer<sup>2</sup>, on calcule la position de l'objet déplaçable. Si aucun autre déplaçable ne se trouve à la position calculée, on y place le déplaçable en cours de mouvement. Dans le cas contraire, une collision a lieu.

#### 3.3.6.2 *Collisions*

Chaque déplaçable a un score de collision, dépendant de ses caractéristiques et d'un jet de dé aléatoire. Lorsqu'une collision a lieu, on demande le score de collision de chacun des

---

1. 3.048 m

2. Les déplacements sont ordonnés par ordre de réception

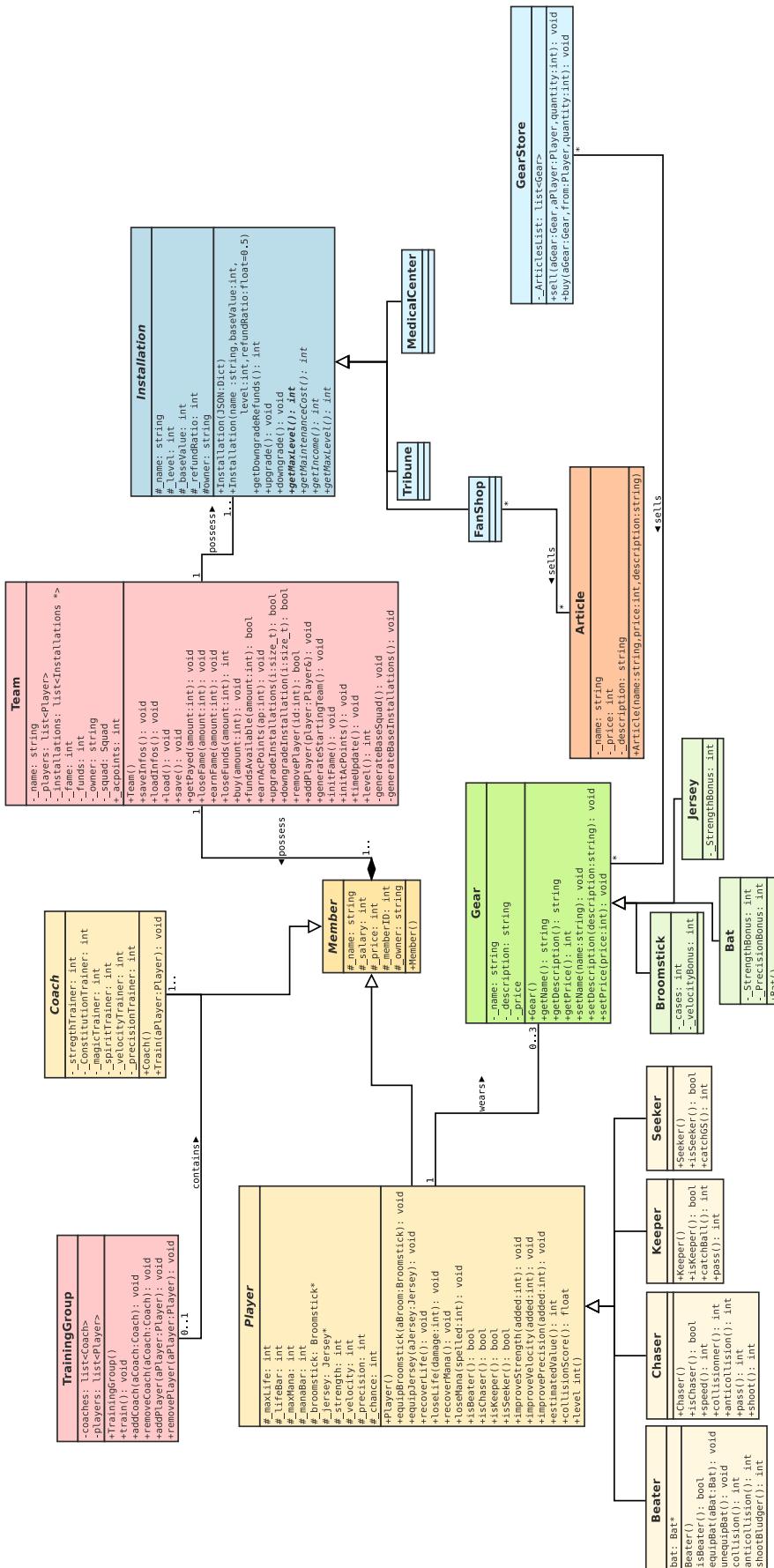


FIGURE 3.11 – Classes : la gestion de l'équipe

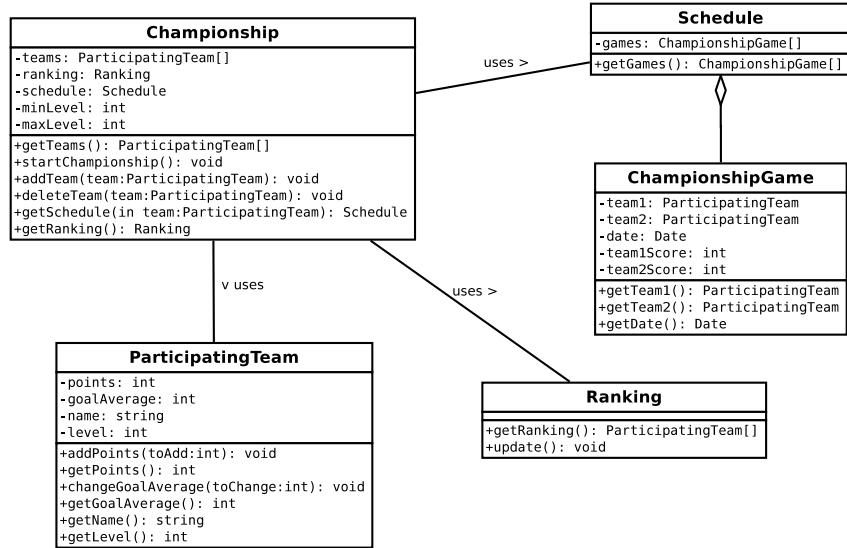


FIGURE 3.12 – Classes : les championnats

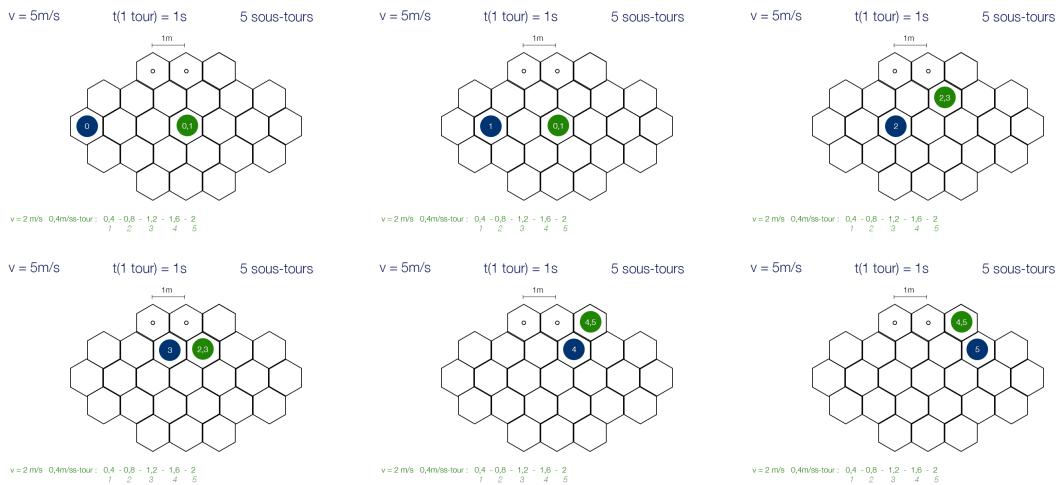


FIGURE 3.13 – Déplacements des joueurs sur le terrain

déplaçables. Celui ayant le plus grand remporte la case disputée. Plusieurs sous-cas peuvent alors se présenter :

#### Le premier joueur sur la case gagne

Le second prétendant à cette position voit son mouvement arrêté. Il est placé sur la dernière case qu'il a atteint et son déplacement est supprimé de la case

#### Le second arrivant gagne

Le premier joueur est placé sur une case libre adjacente, son mouvement est supprimé, et le second joueur lui vole la position.

**Un des deux déplaçable est attrapable par l'autre**

Le joueur attrape l'autre déplaçable.

## Annexe A

# À propos de la librairie graphique

Nous avons choisi d'utiliser la SFML (simple and fast multimedia library) dans l'optique de notre projet.

Cette librairie open-source a été codée en C++ spécialement pour le développement de jeux 2D, et permet l'utilisation de la programmation orientée objet, s'accordant donc avec l'optique de ce travail : apprendre le développement d'applications à des étudiants universitaires. De plus, le site web de la librairie<sup>1</sup> regorge de documentation et tutoriels.

Une alternative envisagée était SDL, fer de lance du développement de jeux 2D, mais l'utilisation intensive d'OpenGL par la SFML rend celle-ci souvent plus rapide dans son exécution<sup>2</sup>.

Enfin, pour ne rien gâcher, la SFML possède une communauté très active, et tout particulièrement la communauté francophone, le développeur étant lui-même Français.

---

1. <http://www.sfml-dev.org/>  
2. <http://en.sfml-dev.org/forums/index.php?topic=43.0>

## Annexe B

### Dans le futur...

Dans le cadre de ce projet, certaines fonctionnalités supplémentaires auraient pu être ajoutées. Malheureusement, le manque de temps nous aura contraint à les abandonner. Voici quelques-unes de ces idées :

- choix du thème graphique du jeu ;
- une API python (ou autre langage de script) ;
- un chat ou un système de mail entre utilisateurs ;
- un historique des résultats des matches et championnats ;
- sessions d'*entraînements* ;
- *sponsors* ;
- un système d'achat d'argent et de points d'action via Paypal.

## Annexe C

# Diagramme de classes général

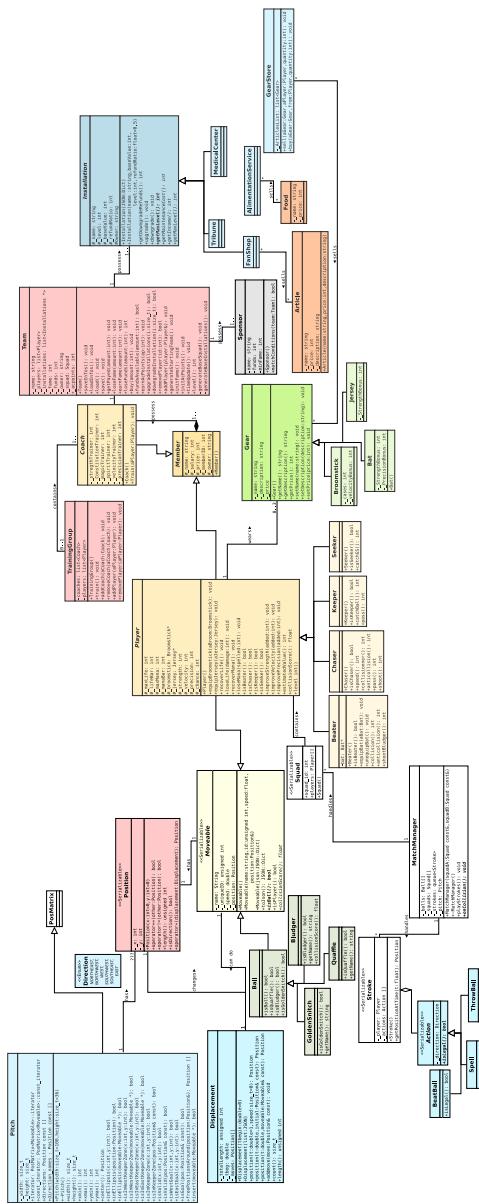


FIGURE C.1 – Classes : diagramme général

## Annexe D

# Diagramme de composants

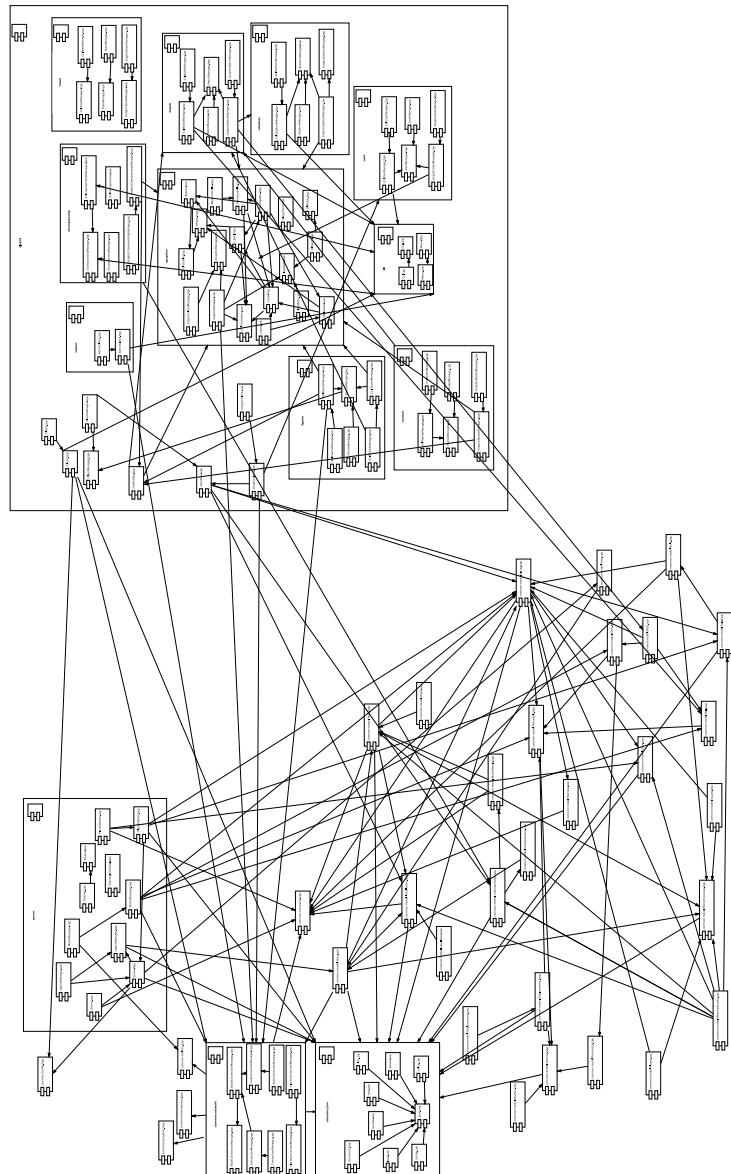


FIGURE D.1 – Diagramme de composants

# Table des figures

2.1	Cas d'utilisation : écran de démarrage . . . . .	8
2.2	Cas d'utilisation : Diagramme d'activité : écran de démarrage . . . . .	8
2.3	Cas d'utilisation : dashboard . . . . .	9
2.4	Cas d'utilisation : gestion d'équipe . . . . .	10
2.5	Cas d'utilisation : gestion du stade . . . . .	12
2.6	Cas d'utilisation : aller aux transferts . . . . .	14
2.7	Déroulement d'une enchère . . . . .	16
2.8	Cas d'utilisation : championnat . . . . .	17
2.9	Cas d'utilisation : matches amicaux . . . . .	19
2.10	Cas d'utilisation : diagramme d'activité : matches amicaux . . . . .	20
2.11	Cas d'utilisation : match de Quidditch . . . . .	22
3.1	Cas d'utilisation : Système - partie gestion . . . . .	27
3.2	Cas d'utilisation : Système - championnat . . . . .	31
3.3	Composants . . . . .	34
3.4	Classes : outil de sérialisation JSON . . . . .	35
3.5	Séquence : le jeu . . . . .	36
3.6	Classes : Le gestionnaire de connexion . . . . .	37
3.7	Classes : Les gestionnaires de jeu du client . . . . .	37
3.8	Classes : le gestionnaire de match . . . . .	38
3.9	Classes : les objets déplaçables du match . . . . .	38
3.10	Classes : les actions des joueurs . . . . .	38
3.11	Classes : la gestion de l'équipe . . . . .	40
3.12	Classes : les championnats . . . . .	41
3.13	Déplacements des joueurs sur le terrain . . . . .	41
C.1	Classes : diagramme général . . . . .	45
D.1	Diagramme de composants . . . . .	46