SOFTWARE REQUIREMENTS DOCUMENT

Quidditch Manager 2014

Groupe 4 : Manon Legrand, Hélène Plisnier, Audry Delestree, David Bertha
INFOF209
Université Libre de Bruxelles
Version 1.0

Table des matières

1 Introduction **3**
- 1.1 But du projet . . . 3
- 1.2 Historique du document . . . 3
- 1.3 Glossaire . . . 4

2 Besoins de l'utilisateur **6**
- 2.1 Exigences fonctionnelles . . . 6
  - 2.1.1 Cas d'utilisation 1 : démarrer le jeu . . . 6
  - 2.1.2 Cas d'utilisation 2 : jouer un match . . . 8
  - 2.1.3 Cas d'utilisation 3 : gérer ses bâtiments . . . 12
  - 2.1.4 Cas d'utilisation 4 : gérer son équipe . . . 14
  - 2.1.5 Cas d'utilisation 5 : magasins . . . 16
  - 2.1.6 Cas d'utilisation 6 : améliorer ses joueurs . . . 18
  - 2.1.7 Cas d'utilisation 7 : le stade . . . 20
- 2.2 Exigences non fonctionnelles . . . 21
- 2.3 Exigences de domaine . . . 21

3 Besoins du système **22**
- 3.1 Exigences fonctionnelles . . . 23
  - 3.1.1 Cas d'utilisation 1 : Quidditch Manager 2014 . . . 23
- 3.2 Exigences non fonctionnelles . . . 27
  - 3.2.1 Le réseau . . . 27
  - 3.2.2 Performance . . . 27
  - 3.2.3 Sécurité . . . 27
  - 3.2.4 Environnement d'exécution . . . 27
- 3.3 Design et fonctionnement du système . . . 27
  - 3.3.1 Diagrammes de classes . . . 27
  - 3.3.2 Diagramme de séquence . . . 34

4 Index **35**

1 Introduction

1.1 But du projet

Le Quidditch manager est un jeu multi-joueurs de gestion et de stratégie, se jouant par réseau. Un manager (l'utilisateur) gère son équipe de joueurs, qu'il peut entrainer au travers des bâtiments construits sur la parcelle qui lui est attribuée. Certains de ces bâtiments permettent aussi de faire des achats et des ventes de balais ou de joueurs. Le but pour le manager est de développer son club aussi bien au niveau commercial que sportif. À travers la construction et l'évolution de ses bâtiments, le manager ouvre et augmente des possibilités de gestion du club, notamment la possibilité d'améliorer les caractéristiques de ses joueurs, ces caractéristiques étant déterminantes dans le déroulement d'un match. Les différents managers sont périodiquement invités à s'affronter par ces matchs s'inscrivant dans un championnat, étalonné dans le temps.

Ces matchs se déroulent dans un stade propre à un des managers et apportent une entrée d'argent dûe notamment à la victoire d'une équipe mais aussi au nombre de tickets vendus. Au cours du match, les managers en présence choisissent à chaque tour de jeu des déplacements pour leurs différents joueurs. Il existe quatre fonction possible pour un joueur lors d'un match. Un joueur peut être un attrapeur, un batteur, un gardien ou un poursuiveur. Chacun de ces rôles implique des actions possibles différentes au cours du match. Mais il est possible à chacun de se déplacer sur le terrain, celui-ci étant représenté par des hexagones (il y a donc 6 directions de déplacement possibles à partir d'une case). De plus, les joueurs se partagent le terrain avec les différentes balles du jeu, qu'ils devront manipuler selon les règles pour emmener leur équipe vers la victoire, celle-ci étant déterminée lorsque le vif d'Or est attrapé ou par abandon.

1.2 Historique du document

Numéro de version	Auteur	Date de la modification	Description des changements
0.1	David	13/12/13	Squelette du code LaTeX
0.2	Hélène	19/12/13	Première ébauche de la section 2.1
0.3	Manon	20/12/13	Section 2.1 finale
0.4	David	20/12/13	Sections 1, 2.2 et 2.3
0.5	Manon	20/12/13	Section 3.1
0.5	David	20/12/13	Section 3.2
0.6	Audry	20/12/13	Section 3.3
0.7	David	20/12/13	Glossaire et Index
1.0	Manon	20/12/13	Relecture et révision

1.3 Glossaire

agence de publicité lieu où le Manager peut faire augmenter la popularité de ses joueurs. 18

attrapeur joueur dont la fonction au sein de l'équipe en cours de match est d'attraper le Vif d'Or. 3, 10

batteur au nombre de deux par équipe au cours d'un match. Un batteur frappe sur les Cognards et les envoie sur les joueurs adverses pour les blesser/déstabiliser. 3

buvette lieu où les supporters de l'équipe en déplacement pour un match qui ne se rendent pas sur place peuvent assister à une retransmission du match. Les consommations sont payantes et les bénéfices sont reversés au Manager. 26

calendrier ensemble de rappels fixés dans le temps, le Manager est régulièrement rappelé à certaines actions (comme jouer un match) sur base de ce calendrier. 13, 20, 25

centre de recrutement lieu où le Manager peut acheter/vendre des joueurs. 16

centre d'entrainement lieu où le Manager peut faire s'entrainer ses joueurs. 18

club est un ensemble reprenant une équipe de Quidditch et son infrastructure. 3, 14

cognard au nombre de deux. Plus petite que le Souafle. Balle magique qui essaie de frapper les joueurs pour les faire tomber de leur balais. Ces balles sont cognées par les batteurs pour les diriger vers les joueurs adverses. 10

fanshop lieu où les supporters de l'équipe qui reçoit une équipe adverse pour un match peuvent acheter des produits dérivés. 26

gardien il n'y en a qu'un par équipe. Son rôle est d'empêcher les Poursuiveurs de l'équipe adverse de marquer des buts en interceptant leurs tirs. 3, 10

infirmerie lieu où les joueurs du Managers sont soignés. 33

joueur est un membre d'une équipe de Quidditch. 3, 6, 29

magasin de balais lieu où le Manager peut acheter/vendre des balais. 16

manager est l'utilisateur du programme, en charge de la gestion d'un club de Quidditch et de ses matchs. 3

match est un affrontement de deux équipes de Quidditch sur un terrain. 3, 8

parcelle espace alloué à l'utilisateur où se trouvent ses bâtiments et son terrain de Quidditch. 6

partie désigne la session de jeu que s'est créé l'utilisateur/Manager et qu'il peut gérer à sa guise. Il y a une partie par utilisateur. 6, 24

poursuiveur au nombre de 3 par équipe au cours d'un match, font des passes de Souafle entre eux et tentent de marquer des buts. Quand c'est l'équipe adverse qui a le souafle, ils essaient de le récupérer. 3, 10

Quidditch est un sport fictif issu de la saga Harry Potter créée par J. K. Rowling. Chaque équipe possède sept joueurs chevauchant des balais volants. L'objectif étant de marquer plus de points que l'adversaire en marquant un maximum de buts et en attrapant une balle magique, le Vif d'or. Un match peut durer des mois et comporte des risques mortels pour les joueurs. 3, 6

souafle balle inerte unique manipulée pendant un match par les Poursuiveurs à travers des passes et des tentatives de tirs dans l'un des cercles d'or pour marquer un but. 9

stade lieu sur l'espace alloué à un Manager où sont joués les matchs de Quidditch. 6, 20

terrain espace à l'intérieur du stade où se déplacent les joueurs et les balles au cours d'un match. 32

vif d'Or petite balle unique ayant sa propre volonté, qui se déplace aléatoirement et très rapidement dans les airs. Lorsqu'un Attrapeur attrape le Vif d'Or, la partie s'achève. 3, 10

2 Besoins de l'utilisateur

2.1 Exigences fonctionnelles

Les cas d'utilisation présentés ci-dessous décrivent les différentes actions proposées au client par le programme.

2.1.1 Cas d'utilisation 1 : démarrer le jeu

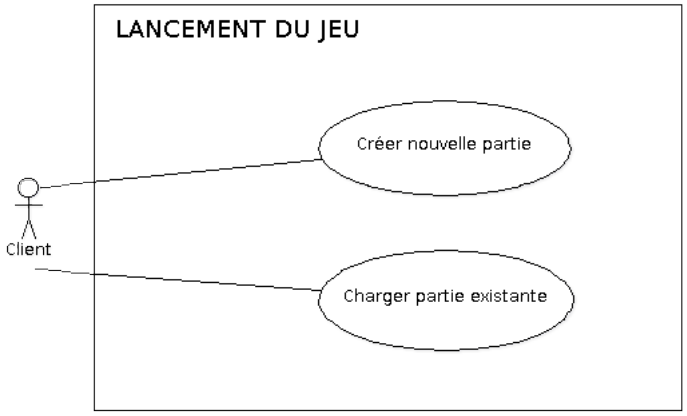

FIGURE 2.1: Cas d'utilisation 1 : Démarrer le jeu

L'utilisateur démarre le jeu. Soit il a déjà commencé une partie ; dans ce cas, l'utilisateur s'identifie et reprend sa partie là où il l'avait laissée.
Soit il ne possède pas encore de partie ; le programme lui demande alors quelques informations d'identification. Il se voit ensuite attribuer une équipe de 7 joueurs médiocres et une parcelle. Au commencement, ce dernier est presque désert, composé uniquement d'un stade de Quidditch et d'emplacements vides prévus pour accueillir les bâtiments.

Créer nouvelle partie

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Le client a choisi de créer une nouvelle partie.
- **Post-conditions** : Néant.
- **Cas général** : Le client veut créer une nouvelle partie. Il devra alors rentrer toute une série d'informations concernant cette partie nécessaire à l'identification de cette

partie et au bon fonctionnement du jeu (nom du manager, nom de l'équipe, mot de passe, etc.)
- **Cas exceptionnels** :
  - Un manager avec le nom indiqué par le client existe déjà. Ceci termine ce use case.

Charger partie existante

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Il faut qu'il existe déjà une partie à laquelle le client a accès.
- **Post-conditions** : Néant.
- **Cas général** : Le client souhaite continuer une partie déjà entamée et rentre le nom de la partie ainsi que le mot de passe pour récupérer les informations concernant cette partie.
- **Cas exceptionnels** :
  - Il n'y a pas de partie existante, ceci termine ce use case.
  - L'identification de la partie a échoué (mauvais nom ou mot de passe), ceci termine ce use case.

2.1.2 Cas d'utilisation 2 : jouer un match

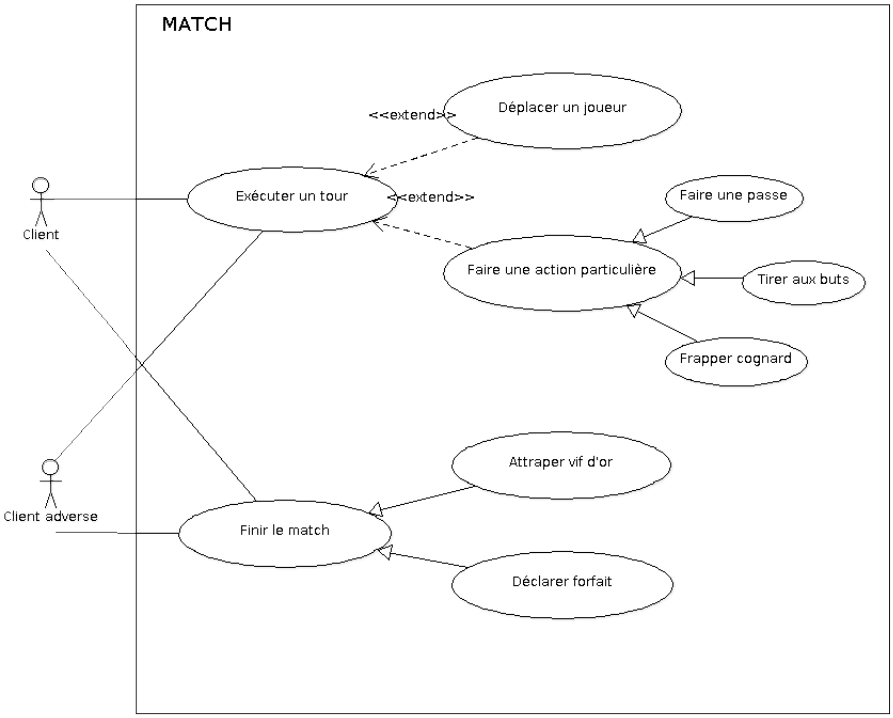

FIGURE 2.2: Cas d'utilisation 2 : Jouer un match

Tous les X temps, le Manager est invité à affronter un Manager adverse dans un match soit à domicile, soit à l'extérieur (cette information déterminera à qui reviendront les recettes générées par la vente de tickets). Avant de commencer le match proprement dit, le Manager doit choisir sept de ses joueurs pour former son équipe. Une fois le match terminé, le Manager prend connaissance du résultat du match (s'il a gagné ou perdu) et des recettes éventuellement réalisées.

Exécuter un tour

- **Relations avec d'autres cas d'utilisation** : Est étendu par Déplacer un joueur et Faire une action particulière
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.

- **Cas général** : Un match se déroule tour par tour. À chaque tour, les clients qui s'affrontent doivent indiquer pour chaque joueur ce qu'ils veulent faire.
- **Cas exceptionnels** :
  - L'adversaire s'est déconnecté et ne s'est pas reconnecté dans le temps imparti. Le match s'arrête.

Déplacer un joueur

- **Relations avec d'autres cas d'utilisation** : Etend Exécuter un tour
- **Pré-conditions** : Le joueur choisi pour faire un déplacement n'a pas été choisi pour une action particulière.
- **Post-conditions** : Néant.
- **Cas général** : Le client sélectionne un joueur et choisit où celui-ci doit se déplacer sur le terrain. Le client est informé de l'ensemble des destinations possibles pour le joueur sélectionné.
- **Cas exceptionnels** : Néant.

Faire une action particulière

- **Relations avec d'autres cas d'utilisation** : Etend Exécuter un tour. Généralise Faire une passe, Tirer aux buts et Frapper cognard
- **Pré-conditions** : Le joueur choisi pour faire une action n'a pas déjà été choisi pour faire une action.
- **Post-conditions** : Néant.
- **Cas général** : Le client choisit le joueur et l'action que celui-ci doit réaliser. Cette action dépend du poste du joueur.
- **Cas exceptionnels** : Néant.

Faire une passe

- **Relations avec d'autres cas d'utilisation** : Spécialise Faire une action particulière
- **Pré-conditions** : Le joueur qui va faire la passe doit tenir le souafle.
- **Post-conditions** : Que la passe soit réussie ou ratée, le joueur ne possèdera plus le souafle.
- **Cas général** : Le client choisit l'endroit de destination de la passe. Les endroits possibles dépendent des compétences du joueur qui a le souafle. Attraper le souafle suite à une passe ou l'intercepter (passe adverse) se fait de façon automatique par un poursuiveur se trouvant sur la trajectoire de la passe.
- **Cas exceptionnels** : Néant.

Tirer aux buts

- **Relations avec d'autres cas d'utilisation** : Spécialise Faire une action particulière

- **Pré-conditions** : Le joueur qui veut tirer doit tenir le souafle et se trouver dans l'alignement d'un des buts.
- **Post-conditions** : Qu'il y ait but ou non, le joueur ne tiendra plus le souafle.
- **Cas général** : Le client essaie de marquer un but en indiquant au poursuiveur qui a le souafle vers quel but il doit tirer. Intercepter le souafle se fait automatiquement par le gardien ou les poursuiveurs adverses s'ils se trouvent sur la trajectoire du souafle. Si le tir est réussi, l'équipe qui vient de marquer gagne 10 points.
- **Cas exceptionnels** : Néant.

Frapper cognard

- **Relations avec d'autres cas d'utilisation** : Spécialise Faire une action particulière
- **Pré-conditions** : Le joueur qui va essayer de frapper un cognard doit être un batteur et doit se situer près du cognard.
- **Post-conditions** : Néant.
- **Cas général** : Le client indique dans quelle direction le batteur doit frapper le cognard.
- **Cas exceptionnels** : Néant.

Finir le match

- **Relations avec d'autres cas d'utilisation** : Généralise Attraper vif d'Or et Déclarer forfait
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le match va se terminer, le score est figé et les résultats vont déterminer ce que va gagner ou perdre (argent, popularité) chaque manager.
- **Cas exceptionnels** :
  - Le match se finit parce que les deux clients qui s'affrontent se sont déconnectés et n'ont pas joué le match jusqu'au bout. Le match est considéré comme annulé.
  - Si les 7 joueurs d'une même équipe sont blessés et ne peuvent continuer à jouer, l'équipe adverse gagnera le match. Les joueurs blessés sont envoyés à l'infirmerie et sont bloqués jusqu'à leur rétablissement.

Attraper vif d'or

- **Relations avec d'autres cas d'utilisation** : Spécialise Finir le match
- **Pré-conditions** : Un des attrapeurs a repéré le vif d'Or.
- **Post-conditions** : Néant.
- **Cas général** : Un des attrapeurs a repéré le vif d'or, c'est-à-dire qu'en se déplaçant sur le terrain, il est arrivé suffisamment proche du vif d'or pour que sa précision permette de le repérer (et de le rendre visible pour son manager). L'attrapeur essaiera automatiquement de l'attraper (la réussite de l'action dépend des qualités de

l'attrapeur et de sa distance par rapport au vif d'or). S'il réussit à l'attraper, son équipe gagne 150 points et le match est terminé.
- **Cas exceptionnels** : Néant.

Déclarer forfait

- **Relations avec d'autres cas d'utilisation** : Spécialise Finir match
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Un des clients souhaite mettre fin au match en déclarant forfait (équivaut à une défaite 150-0, peu importe le score lorsqu'il déclare forfait).
- **Cas exceptionnels** :
  - Si un client se déconnecte au cours d'un match et ne se reconnecte pas dans un certain délai de temps, on considérera qu'il a déclaré forfait.

2.1.3 Cas d'utilisation 3 : gérer ses bâtiments

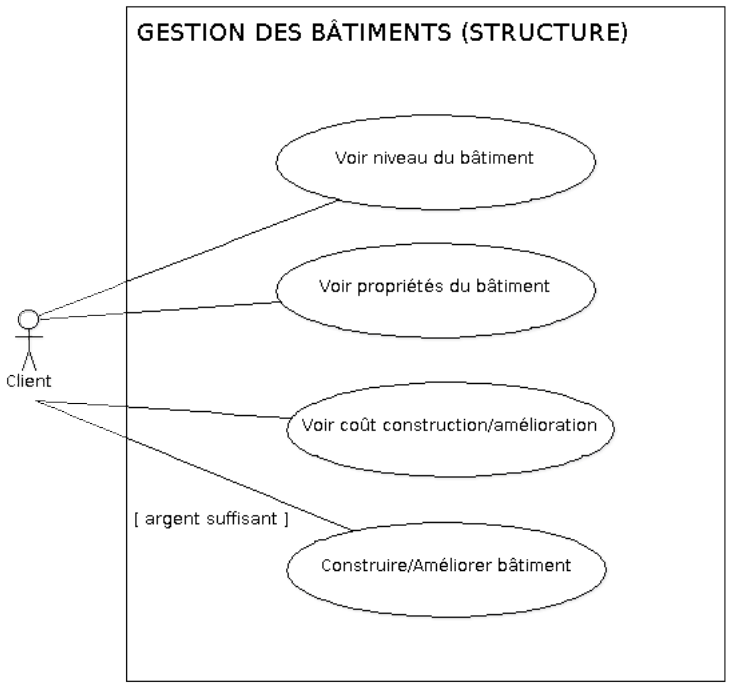

FIGURE 2.3: Cas d'utilisation 3 : Gérer ses bâtiments

Différents bâtiments sont disponibles à la construction pour le Manager, moyennant un certain coût et un temps d'attente (avant fin des travaux). Généralement : une fois construits, le Manager peut améliorer ses bâtiments et accéder aux informations les concernant. Certains bâtiments offrent des usages supplémentaires.

Voir niveau du bâtiment

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : On a sélectionné un bâtiment.
- **Post-conditions** : Néant.
- **Cas général** : Le client souhaite savoir quel est le niveau actuel d'un certain bâtiment.

- **Cas exceptionnels** : Néant.

Voir propriétés du bâtiment

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : On a sélectionné un bâtiment.
- **Post-conditions** : Néant.
- **Cas général** : Le client souhaite voir les propriétés d'un certain bâtiment (ces propriétés dépendent du niveau et du bâtiment).
- **Cas exceptionnels** :
  - Le bâtiment est de niveau 0, il n'a pas encore été construit et n'a donc actuellement pas de propriétés effectives.

Voir coût construction/amélioration

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : On a sélectionné un bâtiment.
- **Post-conditions** : Néant.
- **Cas général** : Le client souhaite voir quelle est la somme requise pour pouvoir lancer un projet de construction ou d'amélioration d'un certain bâtiment.
- **Cas exceptionnels** : Néant.

Construire/Améliorer bâtiment

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Le client doit posséder suffisamment d'argent pour payer le projet de construction ou d'amélioration. On a choisi le bâtiment qu'on veut améliorer.
- **Post-conditions** : Le projet sera terminé après un certain laps de temps déterminé par le niveau du bâtiment. La date de fin des travaux est sauvegardée dans un calendrier qui sera vérifié régulièrement.
- **Cas général** : Le client souhaite construire (si niveau actuel vaut 0) ou améliorer (si niveau actuel vaut 1 ou plus) un certain bâtiment et choisit donc de lancer un projet de construction/amélioration. Ce projet coûte de l'argent et prend un certain temps.
- **Cas exceptionnels** : Néant.

2.1.4 Cas d'utilisation 4 : gérer son équipe

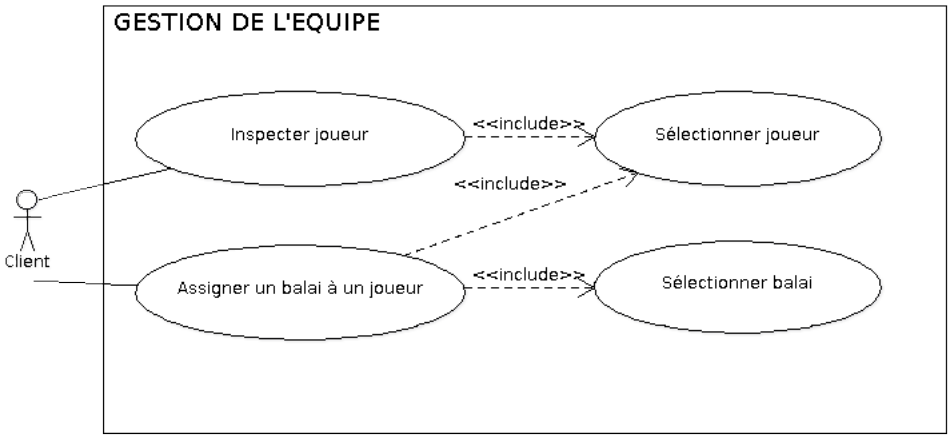

FIGURE 2.4: Cas d'utilisation 4 : gérer son équipe

Le manager a une vue sur tous les joueurs de son club ; leur description et leurs caractéristiques. Il lui est également donné la possibilité d'assigner un nouveau balai à tel ou tel joueur.

Inspecter joueur

- **Relations avec d'autres cas d'utilisation** : Inclut Sélectionner joueur
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le client souhaite se renseigner sur les capacités d'un de ses joueurs. Il pourra voir le niveau de chaque attribut de ses joueurs ainsi que leur popularité et leur balai.
- **Cas exceptionnels** : Néant.

Assigner un balai à un joueur

- **Relations avec d'autres cas d'utilisation** : Inclut Sélectionner joueur et Sélectionner balai
- **Pré-conditions** : Néant.
- **Post-conditions** : Le balai sélectionné sera assigné au joueur. L'ancien balai est à nouveau disponible pour une assignation.
- **Cas général** : Le client souhaite changer le balai d'un joueur. Les balais ont des bonus particuliers qui permettent d'augmenter certains attributs des joueurs. Le

joueur possèdera un nouveau balai. L'ancien balai ne sera plus assigné à un joueur mais reste disponible.
- **Cas exceptionnels** :
  - Le client peut retirer le balai d'un joueur sans lui en assigner un nouveau s'il ne sélectionne pas de balai à assigner.

Sélectionner joueur

- **Relations avec d'autres cas d'utilisation** : Est inclus dans Inspecter joueur et Assigner un balai à un joueur
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le client choisit le joueur qu'il veut gérer (inspecter ou changer le balai).
- **Cas exceptionnels** : Néant.

Sélectionner balai

- **Relations avec d'autres cas d'utilisation** : Est inclus dans Assigner un balai à un joueur
- **Pré-conditions** : Le balai doit être disponible, c'est-à-dire qu'il ne doit pas déjà être assigné à un joueur.
- **Post-conditions** : Néant.
- **Cas général** : Le client choisit le balai qu'il va assigner à un joueur. Le balai sera assigné et ne sera plus disponible.
- **Cas exceptionnels** : Néant.

2.1.5 Cas d'utilisation 5 : magasins

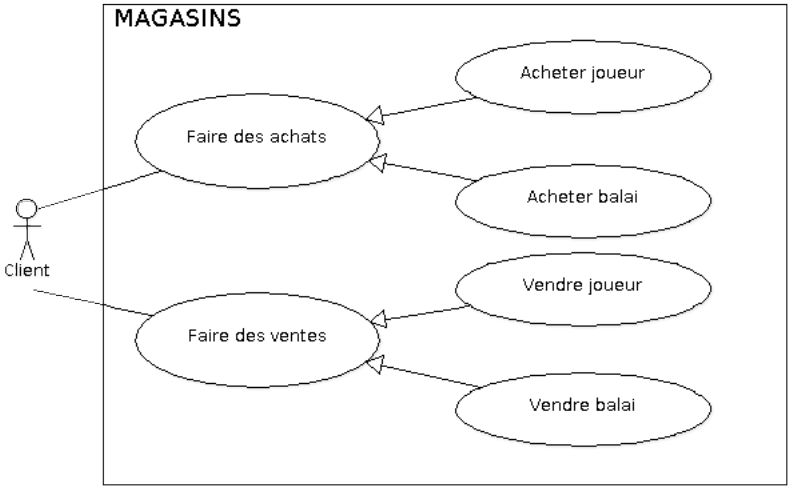

FIGURE 2.5: Cas d'utilisation 5 : Magasins

Dans le bâtiment « centre de recrutement » , le Manager pourra "s'offrir" de nouveaux joueurs, selon ses moyens. Il peut également vendre des joueurs qu'il possède. La valeur d'un joueur est déterminée par ses capacités. Dans le bâtiment « magasin de balais » , le Manager pourra acheter ou vendre des balais. Outre les balais de base qui permettent juste de voler, il y a aussi des balais qui octroient des bonus à certaines capacités des joueurs.

Faire des achats

- **Relations avec d'autres cas d'utilisation** : Généralise Acheter joueur et Acheter balai
- **Pré-conditions** : Le client doit posséder suffisamment d'argent pour payer ce qu'il veut acheter.
- **Post-conditions** : Le prix est retiré du solde du manager.
- **Cas général** : Le client souhaite acheter un nouveau joueur pour renforcer son équipe ou un nouveau balai pour renforcer un joueur.
- **Cas exceptionnels** : Néant.

Faire des ventes

- **Relations avec d'autres cas d'utilisation** : Généralise Vendre joueur et Vendre balai.
- **Pré-conditions** : Néant.

- **Post-conditions** : Le client reçoit l'argent de la vente.
- **Cas général** : Le client souhaite vendre un joueur ou un balai s'il n'en a plus besoin ou si ses finances vont mal.
- **Cas exceptionnels** : Néant.

Acheter joueur

- **Relations avec d'autres cas d'utilisation** : Spécialise Faire des achats
- **Pré-conditions** : Le client a encore de la place pour un joueur dans son équipe.
- **Post-conditions** : Néant.
- **Cas général** : Le client choisit le joueur qu'il veut acheter selon les capacités de ce dernier (ce qui l'intéresse), et selon le prix de ce joueur. Les joueurs achetés sont équipés d'un balai médiocre qui n'a pas de valeur financière.
- **Cas exceptionnels** : Néant.

Acheter balai

- **Relations avec d'autres cas d'utilisation** : Spécialise Faire des achats
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le client choisit le balai qu'il veut acheter en fonction des caractéristiques de celui-ci et de son prix.
- **Cas exceptionnels** :
  - Des balais médiocres sans bonus sont disponibles gratuitement.

Vendre joueur

- **Relations avec d'autres cas d'utilisation** : Spécialise Faire des ventes
- **Pré-conditions** : Le joueur que le client souhaite vendre doit être disponible (pas bloqué par une blessure, un entraînement ou une tournée de promotion).
- **Post-conditions** : Néant.
- **Cas général** : Le client choisit le joueur qu'il veut vendre. Le joueur ne sera plus dans l'équipe mais sera disponible à l'achat. Le client peut vendre un joueur équipé d'un balai, ce qui augmente un peu la valeur du joueur.
- **Cas exceptionnels** : Néant.
  - Le client n'a que 7 joueurs dans son équipe et en vend un. Il ne lui reste pas suffisamment de joueurs pour avoir une équipe complète, il recevra donc un joueur médiocre avec des attributs et un balai de base.

Vendre balai

- **Relations avec d'autres cas d'utilisation** : Spécialise Faire des ventes
- **Pré-conditions** : Le balai doit être disponible, c'est-à-dire qu'il ne doit pas être assigné à un joueur.

- **Post-conditions** : Le balai ne sera plus dans l'inventaire de balais du client, mais sera disponible à l'achat.
- **Cas général** : Le client désire vendre un balai.
- **Cas exceptionnels** : Néant.

2.1.6 Cas d'utilisation 6 : améliorer ses joueurs

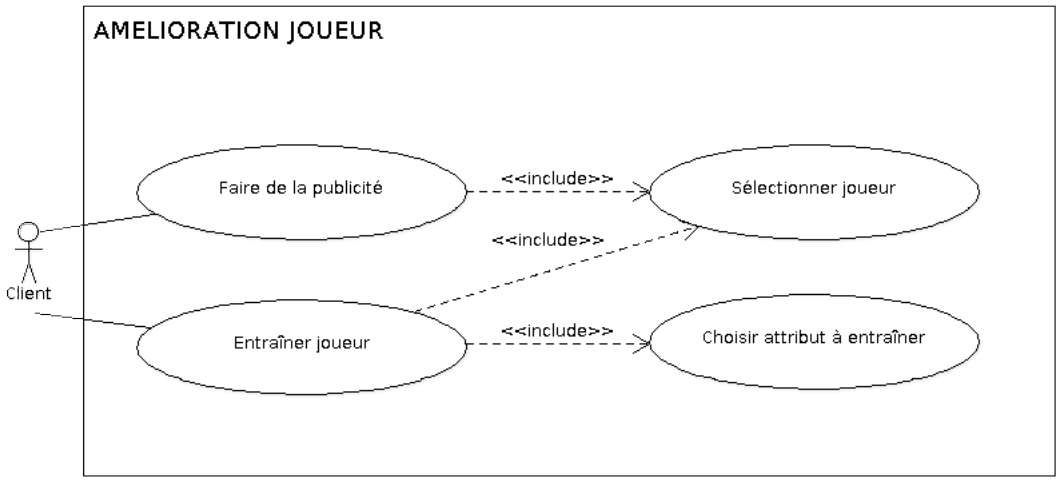

FIGURE 2.6: Cas d'utilisation 6 : Amélioration joueur

Lorsqu'il voudra augmenter les capacités de l'un de ses joueurs, le Manager l'enverra au « centre d'entrainement » . Une séance d'entraînement a un certain coût. Ce coût varie en fonction du niveau d'entraînement déjà atteint par le joueur et en fonction du niveau du bâtiment. Le manager peut également améliorer la popularité d'un joueur en l'envoyant à l'« agence de publicité » . Encore une fois, le Manager doit payer pour bénéficier de ces services. Ces actions rendent le joueur indisponible jusqu'à la fin du temps requis pour réaliser l'entraînement ou la publicité.

Faire de la publicité

- **Relations avec d'autres cas d'utilisation** : Inclut Sélectionner joueur
- **Pré-conditions** : Le client dispose de suffisamment d'argent pour payer la publicité.
- **Post-conditions** : Le joueur est bloqué jusqu'à une certaine date sauvegardée dans un calendrier. Ce calendrier sera vérifié régulièrement.

- **Cas général** : Le client souhaite augmenter la popularité d'un joueur. Ceci est possible grâce au bâtiment « Agence de publicité » dont le niveau détermine la durée. En payant une certaine somme, le client peut donc augmenter la popularité du joueur choisi. Le joueur est indisponible jusqu'à la fin de la publicité.
- **Cas exceptionnels** : Néant.

Entraîner joueur

- **Relations avec d'autres cas d'utilisation** : Inclut Sélectionner joueur et Choisir un attribut à entraîner.
- **Pré-conditions** : Le client dispose de suffisamment d'argent pour payer l'entraînement.
- **Post-conditions** : Le joueur est bloqué jusqu'à une certaine date sauvegardée dans un calendrier. Ce calendrier sera vérifié régulièrement.
- **Cas général** : Le client souhaite augmenter un attribut d'un joueur. Le niveau du bâtiment concerné par cette action détermine la durée d'indisponibilité du joueur. En payant une certaine somme, le client peut donc augmenter une caractéristique du joueur choisi. En attendant la fin de l'entraînement, le joueur est donc indisponible.
- **Cas exceptionnels** : Néant.

Sélectionner joueur

- **Relations avec d'autres cas d'utilisation** : Est inclus dans Faire de la publicité et Entraîner joueur
- **Pré-conditions** : Le joueur sélectionné doit être disponible; c'est-à-dire qu'il ne doit pas être blessé (car il est en pris en charge par l'infirmerie), ni en train de faire un autre entraînement ou en campagne de promotion.
- **Post-conditions** : Le joueur sur lequel on va faire l'amélioration sera indisponible jusqu'à la fin de l'entraînement ou de la publicité.
- **Cas général** : Le client choisit de quel joueur il veut augmenter un attribut ou la popularité.
- **Cas exceptionnels** : Néant.

Choisir attribut à entraîner

- **Relations avec d'autres cas d'utilisation** : Est inclus dans Entraîner joueur
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le client choisit quel attribut il souhaite augmenter. Il ne peut entraîner qu'un attribut à la fois.
- **Cas exceptionnels** : Néant.

2.1.7 Cas d'utilisation 7 : le stade

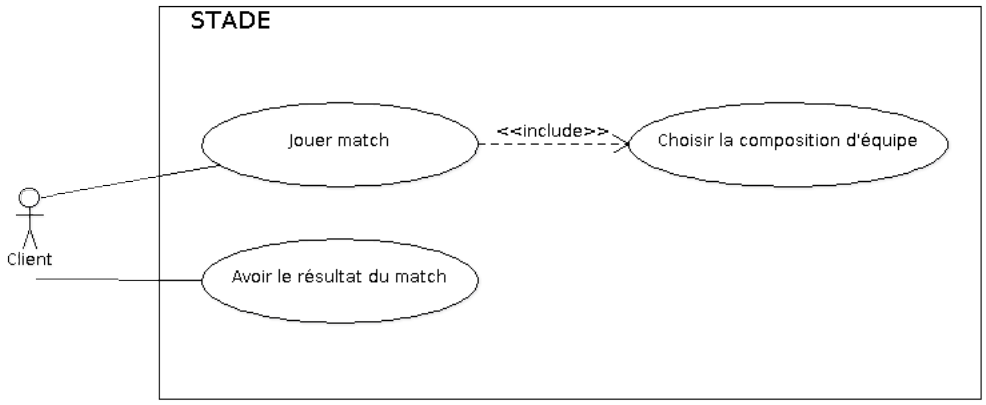

FIGURE 2.7: Cas d'utilisation 7 : Stade

C'est via le stade que le Manager peut lancer un match lorsque c'est un jour de match. Il devra alors choisir les joueurs qui participeront à la rencontre. Une fois le match terminé, il pourra voir les résultats de ce dernier, ainsi que ce qu'il a gagné, ou perdu (argent et popularité).

Jouer match

- **Relations avec d'autres cas d'utilisation** : Inclut Choisir la composition d'équipe
- **Pré-conditions** : C'est un jour de match et l'adversaire est également connecté.
- **Post-conditions** : Néant.
- **Cas général** : Pour savoir si c'est un jour de match, on vérifie en checkant le calendrier des matches. S'il y a bien un match de prévu, le client est connecté en même temps que son adversaire du jour. Ils doivent donc jouer le match dont la date a été prédéterminée.
- **Cas exceptionnels** :
  - L'adversaire ne se connecte jamais. Le client jouera contre l'équipe de son adversaire qui suivra une stratégie automatique minimaliste.

Choisir la composition d'équipe

- **Relations avec d'autres cas d'utilisation** : Est inclus dans Jouer match

- **Pré-conditions** : Les joueurs choisis doivent être disponibles et être équipés d'un balai.
- **Post-conditions** : Néant.
- **Cas général** : Le client choisit les 7 joueurs qui seront sur le terrain pour le match qu'il doit jouer.
- **Cas exceptionnels** : Néant.

Avoir le résultat du match

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Le match doit être terminé.
- **Post-conditions** : Néant.
- **Cas général** : Après le match, le client peut voir le résultat du match et ce qui s'en suit ; l'argent gagné et l'effet sur la popularité des joueurs.
- **Cas exceptionnels** : Néant.

2.2 Exigences non fonctionnelles

Pour le confort de l'utilisateur, il faudra une interface graphique très intuitive, les menus étant attachés à des bâtiments, leur sélection par la souris ouvrira ces menus. De même, la gestion d'un match devra être aisée, malgré le nombre d'actions à choisir sur un tour. Le plaisir de l'utilisateur dépendra du rythme de la succession de ces tours.

2.3 Exigences de domaine

L'expérience immersive de l'utilisateur doit être connectée à l'univers fixé, à savoir la saga d'Harry Potter. Cela signifie, d'une part, qu'il ne faut pas contrevenir aux droits d'auteur portant sur les oeuvres qui décrivent cet univers, et d'autre part, qu'il faut y ressembler suffisament pour que l'utilisateur qui connait l'oeuvre y retrouve les éléments composant cet univers.

3 Besoins du système

3.1 Exigences fonctionnelles

3.1.1 Cas d'utilisation 1 : Quidditch Manager 2014

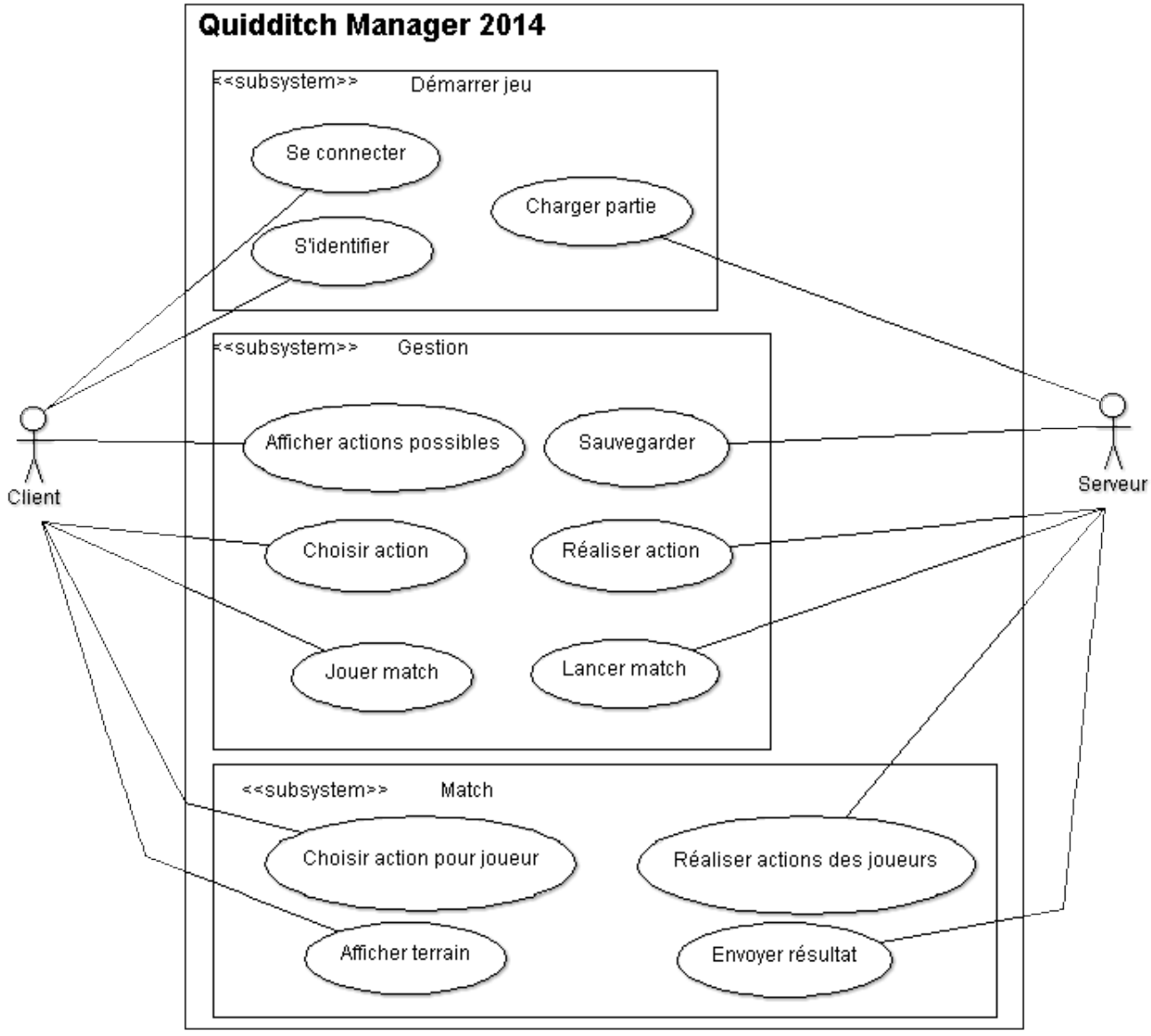

FIGURE 3.1: Cas d'utilisation 1 : Quidditch Manager 2014

Ce diagramme représente le fonctionnement de base du jeu avec la répartition des "tâches" au niveau serveur et client.

Se connecter

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Néant.
- **Post-conditions** : Client et serveur sont reliés par le réseau.
- **Cas général** : Le client se connecte au serveur du jeu pour pouvoir jouer.
- **Cas exceptionnels** : Néant.

S'identifier

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le client crée une nouvelle partie ou en charge une existante pour s'identifier auprès du serveur. Si le client charge une partie existante, le serveur va récupérer le fichier de sauvegarde approprié, sinon, il va en créer un nouveau.
- **Cas exceptionnels** : Néant.

Charger partie

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le serveur va charger les informations de la partie à laquelle va accéder le client.
- **Cas exceptionnels** : Néant.

Sauvegarder

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le serveur sauvegarde les informations de la partie. L'état actuel de la partie du client est sauvegardée dans un fichier de sauvegarde.
- **Cas exceptionnels** : Néant.

Afficher actions possibles

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le client affiche tout ce qu'il est possible de faire d'un point de vue gestion.

- **Cas exceptionnels** : Néant.

Choisir action

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le client a choisi une action de gestion à effectuer (amélioration bâtiment, entraînement, publicité, achat, vente, gestion d'équipe). Il doit maintenant attendre que le serveur exécute cette action.
- **Cas exceptionnels** : Néant.

Réaliser action

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le client a choisi de faire une action de gestion (amélioration bâtiment, entraînement, publicité, achat, vente, gestion d'équipe). Le serveur va se charger de faire les modifications liées à cette action.
- **Cas exceptionnels** : Néant.

Jouer match

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : La date et le moment de la journée correspondent à un match prévu dans le championnat.
- **Post-conditions** : Ce match est retiré du calendrier des matches.
- **Cas général** : On va quitter la partie gestion pour lancer le match. Avant de pouvoir commencer le match, le client doit choisir la liste des joueurs qui seront sur le terrain.
- **Cas exceptionnels** : Néant.

Lancer match

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Au moins un des clients qui doivent s'affronter lors de ce match est connecté.
- **Post-conditions** : Néant.
- **Cas général** : Le serveur va maintenant gérer le match et l'aspect gestion ne sera plus accessible tant que le match ne sera pas terminé.
- **Cas exceptionnels** :
  - Un des clients ne s'est pas connecté pour le match, le serveur prendra sa place pour déplacer ses joueurs et affronter le client connecté.

- Aucun client ne s'est connecté pour le match, le match est considéré comme annulé.

Choisir action pour joueur

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Le joueur sélectionné est encore libre de faire une action.
- **Post-conditions** : Néant.
- **Cas général** : Le client choisit quoi faire pour un joueur. Cela peut-être un déplacement et/ou une action particulière comme passer le souafle ou tirer aux buts. Le client doit faire cela pour chacun de ses joueurs ou indiquer au serveur qu'il a fini de choisir ses actions (s'il souhaite par exemple que des joueurs ne fassent rien).
- **Cas exceptionnels** : Néant.

Réaliser action des joueurs

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : A chaque tour, le serveur attend de recevoir les actions que les clients ont choisies pour chacun de leurs joueurs. Une fois l'ensemble de ces actions reçues, le serveur les exécute puis envoie la nouvelle situation aux clients (pour qu'ils affichent le terrain et redemandent les actions à exécuter).
- **Cas exceptionnels** :
  - Si un des clients ne s'est pas connecté pour le match, le serveur n'attendra pas d'actions de la part de ce client. Ses joueurs suivront une stratégie minimaliste prise en charge par le serveur.

Afficher terrain

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Néant.
- **Post-conditions** : Néant.
- **Cas général** : Le client reçoit les informations relatives au terrain de la part du serveur et l'affiche pour permettre au client de facilement visualiser la situation actuelle du jeu.
- **Cas exceptionnels** : Néant.

Envoyer résultat

- **Relations avec d'autres cas d'utilisation** : Néant.
- **Pré-conditions** : Le match est terminé.
- **Post-conditions** : Le client dispose des informations à afficher après un match.
- **Cas général** : Le match est terminé. Le serveur va calculer combien le client a gagné (en fonction des bâtiments fanshop (match à domicile) ou buvette (match

en extérieur)). En fonction de la situation de fin de match, victoire ou défaite, le serveur va également déterminer combien d'argent et de popularité le manager a gagné ou perdu. Toutes ses informations sont envoyées au client.
- **Cas exceptionnels** : Néant.

3.2 Exigences non fonctionnelles

3.2.1 Le réseau

Au niveau du réseau, le problème de la synchronisation n'est pas préoccupant, car le client ne sera qu'une interface d'affichage et d'interaction. Pour ce qui est de la quantité d'informations échangées, elles ne devraient pas être très importantes pour autant, les éléments à afficher et leur diversité étant relativement assez réduits.

Il faudra veiller à ce que l'interaction avec l'utilisateur soit transparente au niveau du serveur, à travers une classe de message générique, que le programme client instanciera de la même manière, qu'il tourne à travers une interface graphique ou à travers le terminal.

3.2.2 Performance

Pour ce qui est de la performance, la localisation de tous les calculs sur la machine serveur implique une centralisation de la puissance de calcul. D'un autre côté, le client ne nécessitera que très peu de puissance de son côté pour pouvoir jouer.

3.2.3 Sécurité

La sécurité d'une partie sera assurée par l'identification : seul un utilisateur identifié peut charger une partie, et la sienne uniquement.

3.2.4 Environnement d'exécution

L'environement d'exécution étant fixé (les machines des salles informatiques du NO), le code se limitera à l'utilisation des librairies C++ standards. Le système d'exploitation visé est GNU/Linux.

3.3 Design et fonctionnement du système

3.3.1 Diagrammes de classes

Nous allons ici présenter les différentes classes et leurs relations.

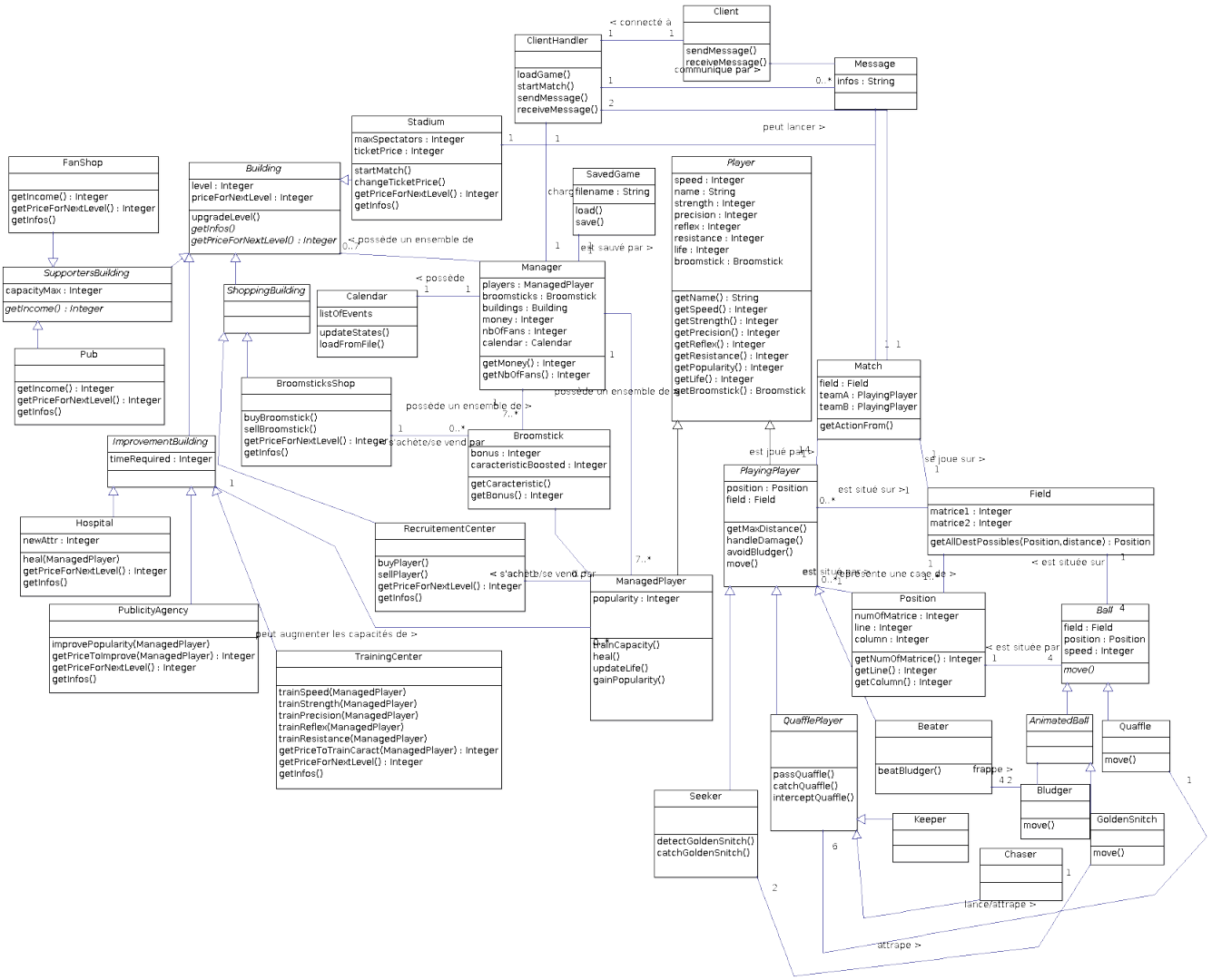

FIGURE 3.2: Diagramme de classe général

Les joueurs

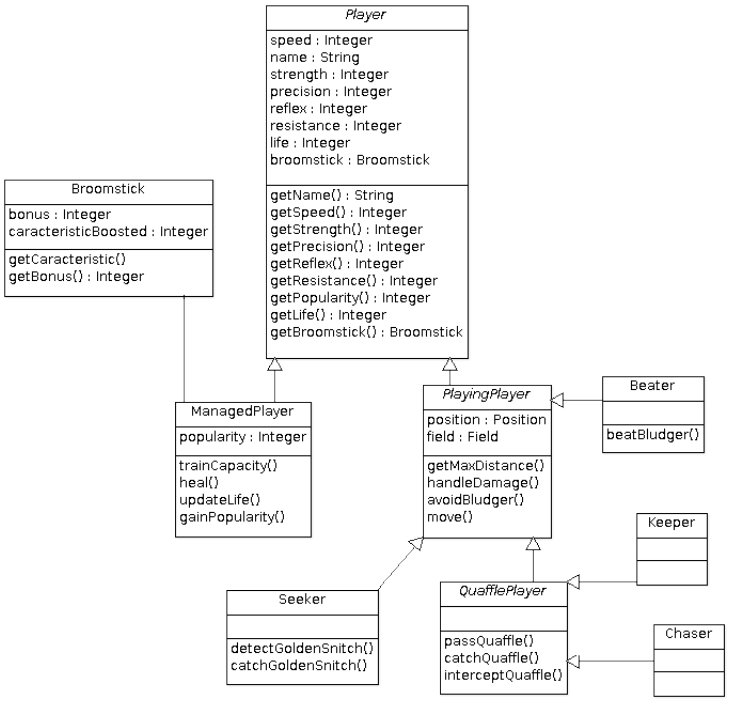

FIGURE 3.3: Diagramme des classes joueurs

Une classe abstraite réprésente le joueur avec ses caractéristiques de base. Dans un soucis de découplage, nous différencions deux classes spécialisées à partir de cette base. Le joueur au niveau de la gestion est distingué du joueur au niveau du match. En effet, ils nécessitent chacun des méthodes fort différentes : d'un côté des méthodes d'amélioration d'attributs, de l'autre des méthodes d'actions spécifiques à un match. Les attributs du joueur de match, ce dernier pouvant se construire depuis un joueur au niveau de la gestion, ne sont pas une simple transposition des caractéristiques du joueur en gestion. En effet, intervient notamment le calcul du bonus lié au balai.

Concernant le réseau

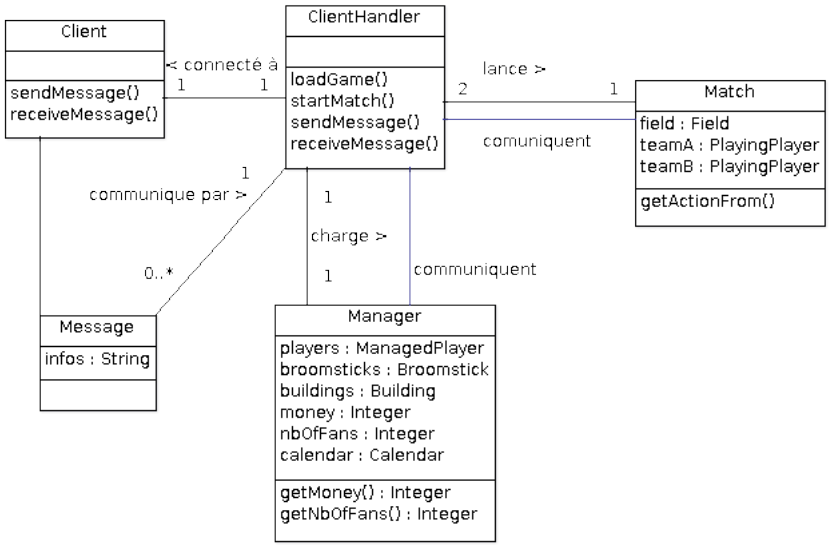

FIGURE 3.4: Diagramme des classes impliquées dans le réseau

La relation avec le client est gérée par une classe principale, qui va assurer la communication à travers une classe de messages (un *struct* puisque la base réseau sera écrite en C). C'est cette classe principale qui va gérer les alternances entre le côté gestion et le côté match. Le match nécessite une connexion entre deux clients, elle est donc très différente de la gestion à ce niveau-là aussi.

Concernant la gestion générale

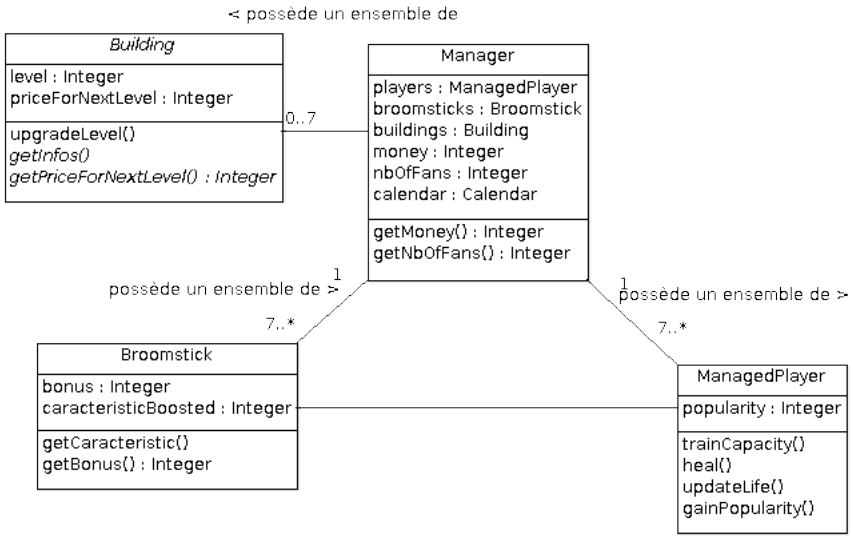

FIGURE 3.5: Diagramme des classes impliquées dans la partie gestion

Le manager est la classe dédiée à la partie gestion. Elle représente une partie du jeu, composée d'un ensemble de joueurs, de balais, de bâtiments, d'un calendrier et d'un trésor. Elle donne accès aux bâtiments qui chacun donne accès aux possibilités d'améliorations, d'achats et de vente.

Concernant le match

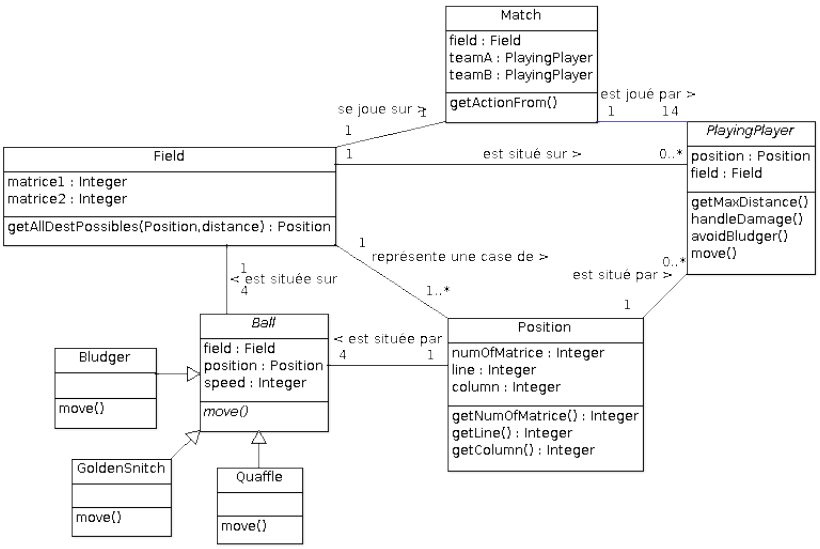

FIGURE 3.6: Diagramme des classes impliquées dans les matchs

Le match relie deux clients différents dans un affrontement sur un terrain où se déplacent à la fois des joueurs et des balles, les joueurs pouvant faire se déplacer les balles. La classe *Position* regroupe les informations de position d'un joueur ou d'une balle sur le terrain.

Les bâtiments

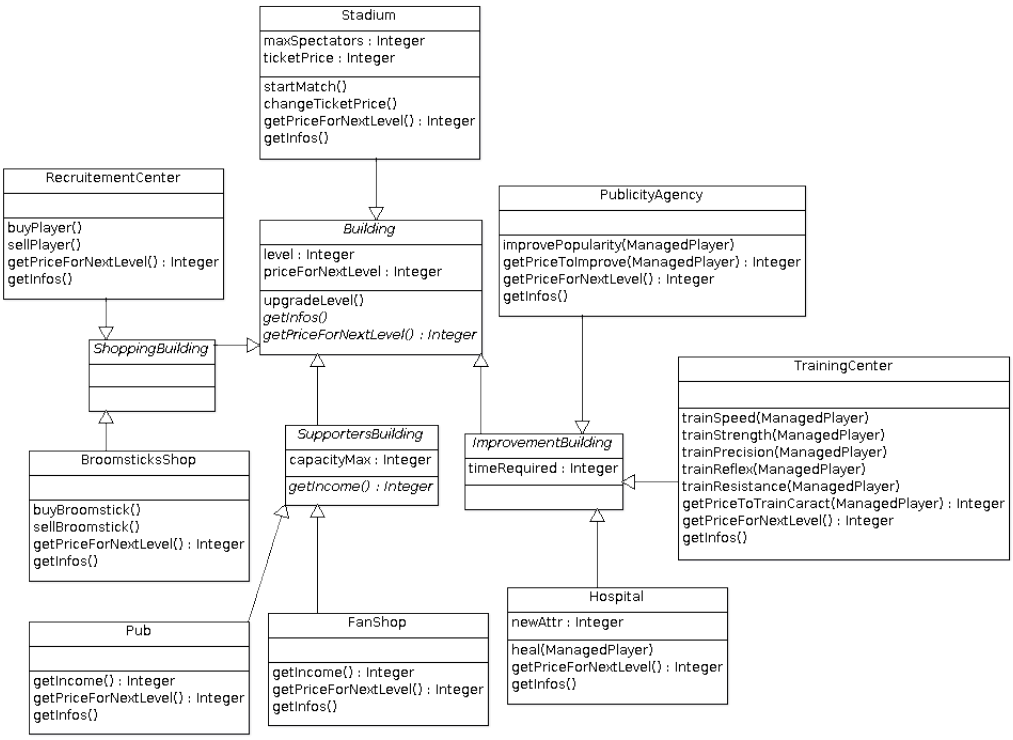

FIGURE 3.7: Diagramme des classes de bâtiments

Une classe abstraite représente tout bâtiment. Cette classe est dérivée dans d'autres classes abstraites qui regroupent les types de bâtiments selon leur rôle principal (achat/vente, augmentation d'une capacité), plus le stade à partir duquel un nouveau match peut démarrer. Par exemple, l'infirmerie fait partie des bâtiments améliorant les joueurs puisqu'elle soigne leurs blessures. Les attributs communs sont le niveau (chaque bâtiment à un niveau, le niveau 0 signifiant « pas encore construit ») ainsi que le prix pour passer au niveau suivant.

Concernant les fichiers

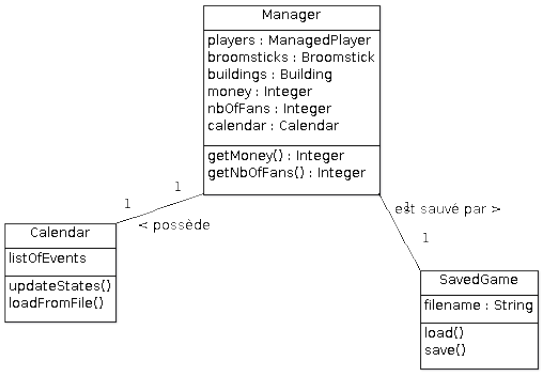

FIGURE 3.8: Diagramme des classes impliquées dans l'enregistrement

L'état d'une partie aussi bien que les actions bloquantes ou planifiées doivent être enregistrées pour perdurer entre deux connexions. Pour cela, nous utilisons deux classes qui vont permettre de charger des données depuis le disque ou de les y sauvegarder.

3.3.2 Diagramme de séquence

Voici un diagramme de séquence qui illustre la gestion des différents rôles du serveur au travers de différents processus.

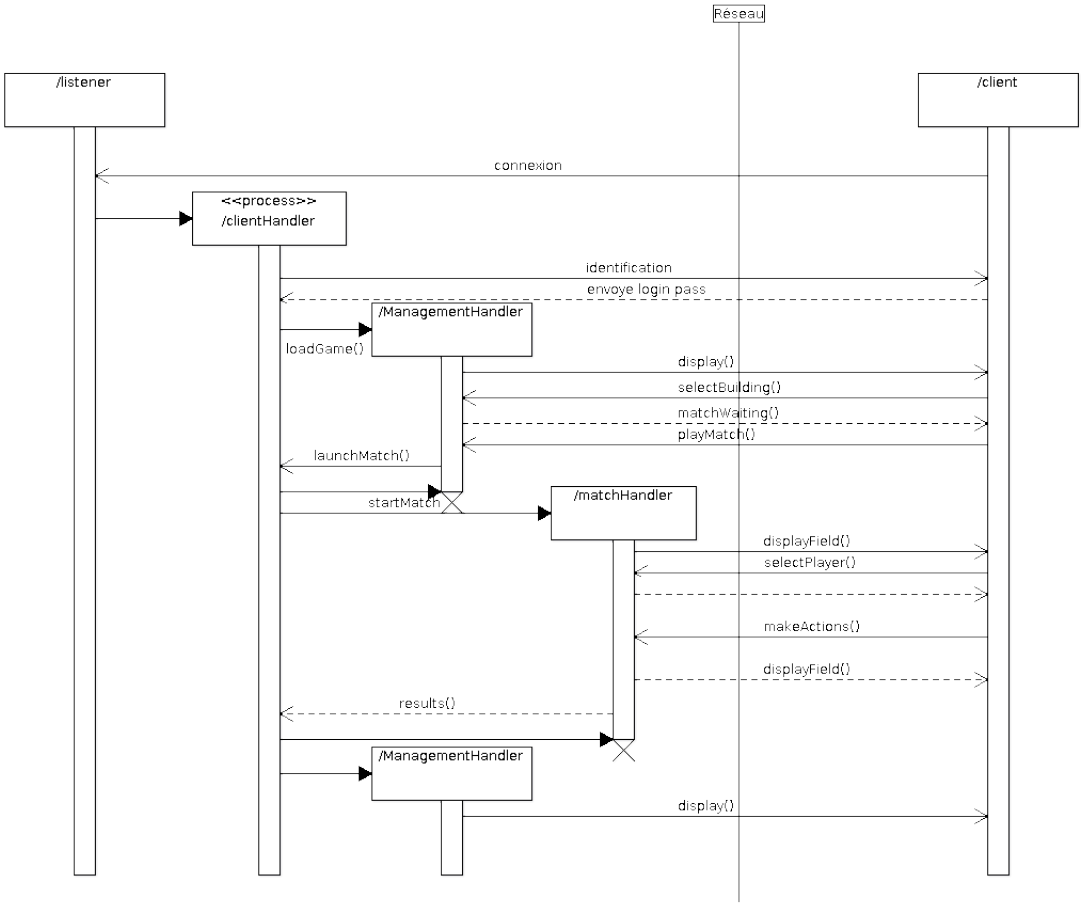

FIGURE 3.9: Diagramme des classes impliquées dans le réseau

Le côté gestion est bien dissocié du côté match, et le *clientHandler* assure la communication avec le client. C'est le *clientHandler* du manager qui reçoit le match dans son stade qui va créer l'instance en charge du match.

4 Index

bâtiment, 31, 33
    Agence de publicité, 18, 19
    Buvette, 26
    Centre d'entraînement, 18
    Centre de Recrutement, 16
    FanShop, 26
    infirmerie, 33
    Magasin de balais, 16
    Stade, 6, 20, 33
balai, 15–17, 31
balle, 32
    cognard, 10
    souafle, 9, 26
    Vif d'Or, 10

calendrier, 13, 20, 25, 31
club, 14

joueur, 17, 21, 29, 31
    attrapeur, 3, 10
    batteur, 3, 10
    gardien, 3, 10
    poursuiveur, 3, 10

manager, 3, 7, 8, 10, 12, 14, 18, 35
match, 8, 25, 29

parcelle, 3, 6
partie, 6, 24

Quidditch, 3, 6

terrain, 9, 10, 21, 26, 32
trésor, 31