# Projet d'année

Mamakis Alexis Ouriaghli Ismaël Dupuis Emma
Renard Simon Grimau Florent Stevens Quentin
Stoyanova Vasilena Benitez Alberdi Diego

21 décembre 2019

# Table des matières



## 1 Introduction

### 1.1 But du projet

L'objectif de ce projet est de créer un jeu d'artillerie et de stratégie multijoueur en réseau du nom de "Lombric à brac", où chaque utilisateur devra se connecter au jeu afin de pouvoir créer ou bien rejoindre une partie. Il aura aussi la possibilité d'être spectateur d'une partie en cours et pourra envoyer des messages depuis le jeu.

Avant de jouer les joueurs pourront nommer leurs lombrics ainsi que leur équipe, l'hôte de la partie pourra également s'occuper de quelques paramètres tels que le mode de jeu (match à mort, match à mort par équipe) ou également le nombre de lombrics par joueur (8 maximum).

Une fois la partie lancée chaque lombric sera dispersé aléatoirement sur le terrain, la partie pourra ensuite commencer. Les joueurs joueront chacun à leur tour et auront la possibilité de faire différentes actions, ils pourront déplacer leur lombric sélectionné ainsi qu'utiliser un outil leur permettant d'attaquer un autre lombric ou de se repositionner.

Pour qu'une partie se finisse, il faut qu'il ne reste que les lombrics d'une seule équipe sur le terrain, l'équipe en vie l'emporte donc, ou bien que tous les lombrics soient morts, mettant fin à la partie par une égalité.

### 1.2 Glossaire

### 1.3 Historique du document

| Version | Auteur | Date | Description |
|---|---|---|---|
| 0.1 | Simon R., Emma D. | 25/11/19 | Besoin utilisateur |
| 0.2 | Alexis M., Ismaël O., Emma D., Simon R., Florent G., Quentin S., Vasilena S., Diego B.A. | 26/11/19 | Création du squelette |
| 0.3 | Alexis M., Ismaël O. | 30/11/19 | Besoins du système |
| 0.4 | Florent G., Quentin S. | 08/12/19 | Modules |
| 0.5 | Diego B.A | 08/12/19 | But du projet |

## 2 Besoin utilisateur

### 2.1 Exigences fonctionnelles

Des diagrammes UML sont utilisés pour décrire les différentes exigences fonctionnelles de l'utilisateur. Ces exigences sont séparées en 3 diagrammes distincts. Ces 3 diagrammes représentent respectivement les interaction d'un utilisateur dans le menu principal, dans le salon d'attente et dans une partie.

### Menu principal

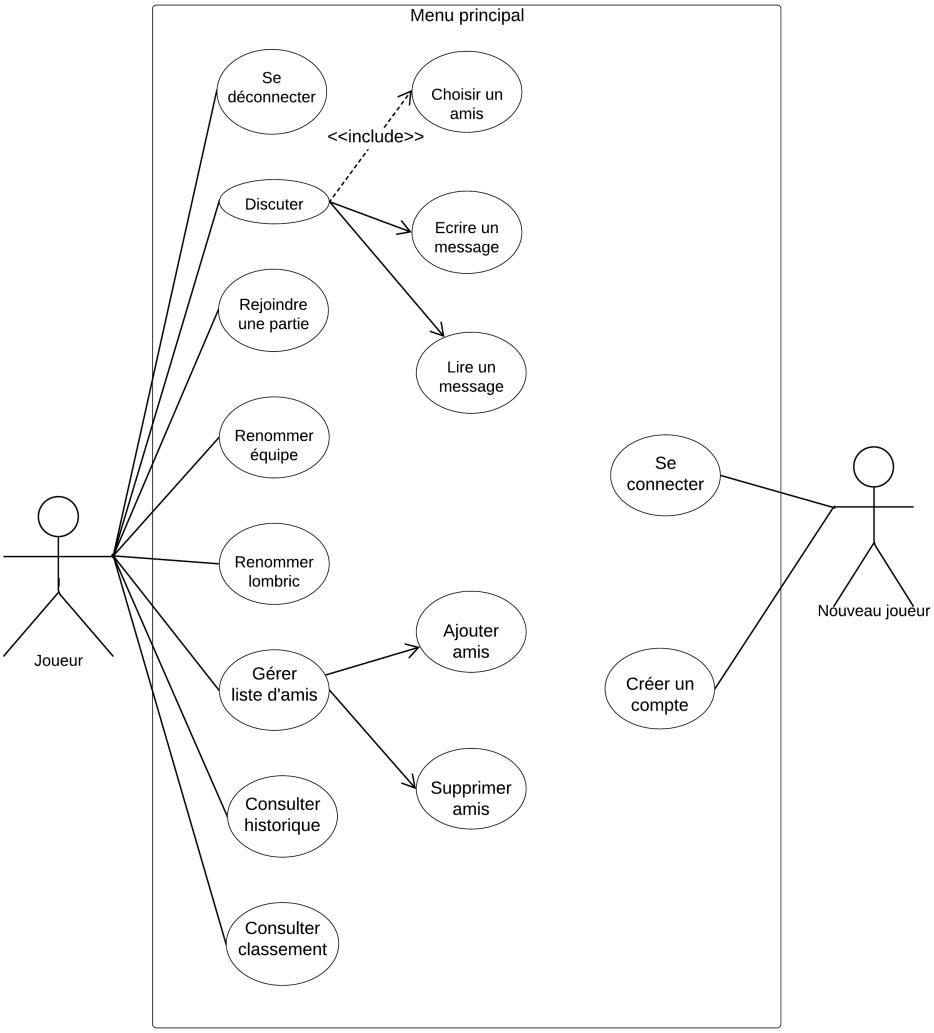


FIGURE 1 – Menu principal

Toutes les actions autre que se connecter et créer un compte impliquent que l'utilisateur est connecté.

| | Pré-conditions | Post-conditions | Cas général | Cas exceptionnels |
|---|---|---|---|---|
| Se connecter | — Avoir un compte<br>— Être déconnecté | Le joueur est connecté | L'utilisateur rentre un nom et un mot de passe corrects | — Le joueur n'est pas connecté au serveur<br>— Le nom ou le mot de passe est incorrect |
| Se déconnecter | Être connecté | L'utilisateur est déconnecté | L'utilisateur se déconnecte | L'utilisateur ferme la fenêtre sans se déconnecter |
| Créer un compte | L'utilisateur n'est pas connecté | Le joueur est connecté sur son nouveau compte | Le nom d'utilisateur n'existe pas encore | Le nom d'utilisateur est déjà pris |
| Ajouter amis | L'utilisateur n'a pas encore en ami la cible | L'utilisateur est ami avec la cible | Le nom cible est un nom d'utilisateur existant | Le nom de l'utilisateur cible n'existe pas |
| Supprimer un ami | L'utilisateur a choisi un de ses amis | L'utilisateur n'est plus ami avec la cible | L'ami est supprimé de la liste de l'utilisateur | Pas d'amis à supprimer |
| Renommer l'équipe | None | L'équipe a un nouveau nom | L'équipe de l'utilisateur est renommée | None |
| Renommer un lombric | — Un lombric a été choisi<br>— Un nom a été choisi | None | Le lombric choisi a été renommé | Le lombric n'avait pas encore été nommé |
| Consulter historique | Un joueur est sélectionné | None | L'historique des parties est affiché | Le joueur sélectionné n'existe pas |
| Consulter le classement | None | None | Le classement est affiché | None |
| Recevoir un message | None | None | Le message est affiché | None |
| Envoyer un message | — Un ami est choisi pour recevoir le message<br>— Un message est écrit | None | Le message est envoyé | None |

FIGURE 2 – Tableau du use case du menu principal

**Salon d'attente**

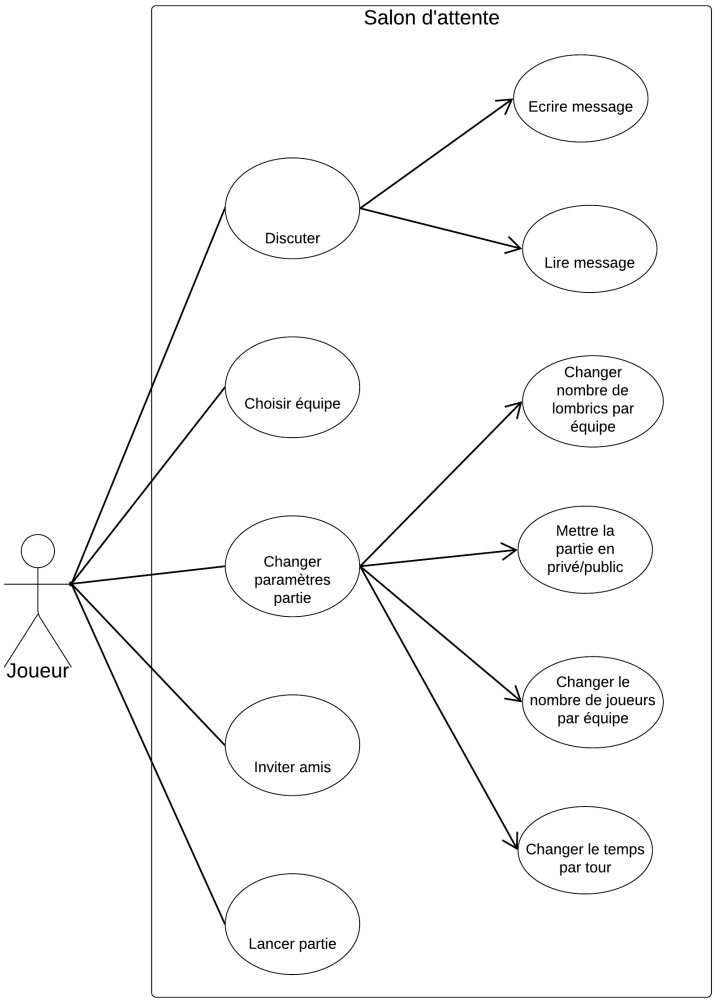


FIGURE 3 – Salon d'attente

Toutes les actions autre que discuter impliquent que la partie n'a pas encore été lancée.

| | Pré-conditions | Post-conditions | Cas général | Cas exceptionnels |
|---|---|---|---|---|
| Discuter | None | Un message est envoyé et peut être vu par les autres | Le joueur veut communiquer avec les autres joueurs | None |
| Choisir une équipe | Équipe pas encore formée | Équipe formée des lombrics choisis | Joueur choisi ses lombrics | None |
| Changer les paramètres d'une partie | None | La partie est configurée | Le joueur choisit les paramètres désirés | None |
| Inviter des amis | Avoir des amis | Amis à rejoint la salle d'attente | Le joueur veut jouer avec un ami | Le joueur ami n'existe pas |
| Lancer la partie | — Les paramètres de la partie sont configurés<br>— Le joueur à créé son équipe<br>— Au moins deux joueurs ont rejoint la partie | La partie est lancée | Le joueur configure la partie et veut commencer à jouer | Il n'y a pas assez de joueurs pour lancer une partie |

FIGURE 4 – Tableau du use case du salon d'attente

**En partie**

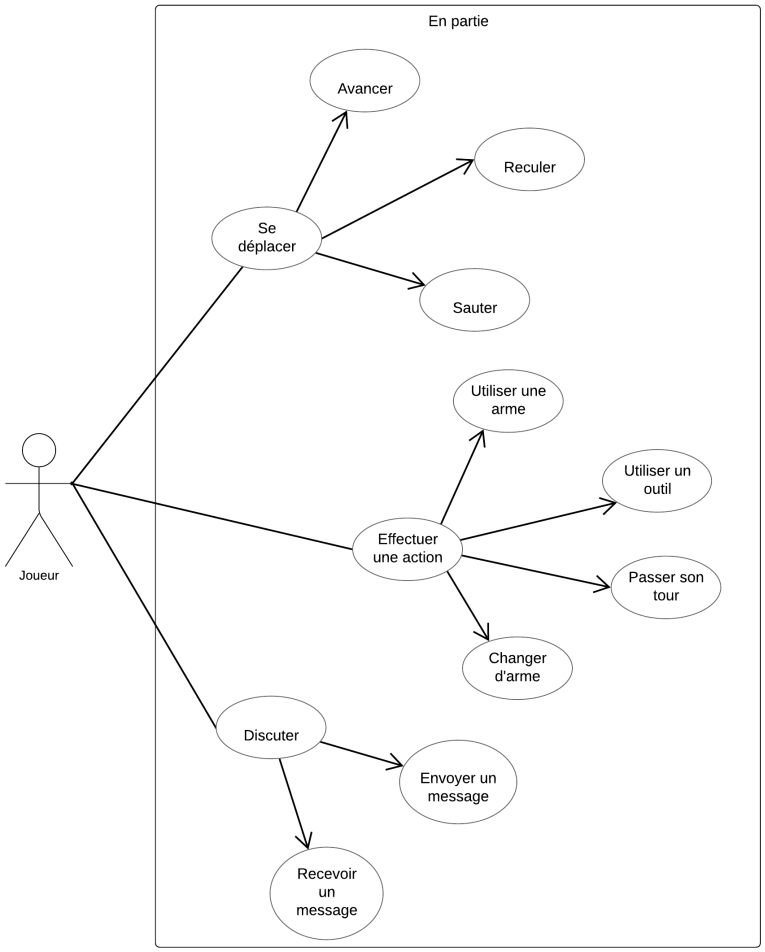


FIGURE 5 – En partie

Toutes les actions autre que discuter s'effectuent uniquement lors du tour du joueur.

| | Pré-conditions | Post-conditions | Cas général | Cas exceptionnels |
|---|---|---|---|---|
| Se déplacer | Déplacement choisit est accessible | — Le joueur se trouve à une autre position<br>— Le tour du joueur est fini | Joueur veut se déplacer sur le terrain | Si le joueur sort du terrain, il meurt |
| Utiliser une arme | — Avoir un lombric capable d'attaquer<br>— Avoir au moins une arme | — Perte de point de vie de l'adversaire attaqué<br>— Le tour du joueur est fini | Le joueur veut infliger des dégâts à l'adversaire | |
| Utiliser un outil | — Avoir au moins un outil<br>— c'est tout ? | Le tour du joueur est fini | L'outil est utilisé | None |
| Passer son tour | None | Le tour du joueur est fini | Joueur à fini son tour | None |
| Recevoir un message | None | None | Le message est affiché | None |
| Envoyer un message | — Un amis est choisi pour recevoir le message<br>— Un message est écrit | None | Le message est envoyé | None |

FIGURE 6 – Tableau du use case d'une partie en cours

## 2.2 Exigences non fonctionnelles

Le langage imposé pour ce projet est le c++. Le jeu doit au minimum tourner sur les ordinateurs de l'ULB (sous Linux).

En cas de perte de connexion d'un des joueurs, il faut faire en sorte que le cas soit traité étant donné que d'autres joueurs attendent ce-dernier.

### 2.3 Exigence du domaine

# 3 Besoin du système

## 3.1 Exigences fonctionnelles

#### Création de compte et gestion de compte

Au lancement du jeu,le programme propose à l'utilisateur de soit créer un compte ou de se connecter a un compte déjà existant, si l'utilisateur décide de créer un compte alors le programme envoie un formulaire d'inscription demandant le nom du nouveau compte (pseudo) ainsi que le mot de passe qui lui sera associé, une fois son compte créé l'utilisateur n'aura qu'a rentrer son pseudo et son mot de passe afin de se connecter.

#### Création de son équipe de lombrics

Lors de la première connexion, le nouveau joueur doit donner un nom à son équipe de huit lombrics.

#### Jouer une partie

**Création d'une partie et configuration de la partie :** Le joueur crée une partie en choisissant le mode de jeu qui lui convient ainsi que les différents paramètres modifiables.

**Gestion des invitations :** Le joueur peut aussi inviter des personnes faisant partie de sa liste d'amis afin de jouer ensemble.

**Trouver une partie :** Le joueur va rejoindre aléatoirement une partie qui est en attente de joueur afin que, si le nombre de joueur max est atteint, la partie se lance.

#### Gestion liste d'amis

Le joueur peut envoyer une demande d'ami à un autre joueur, si l'autre joueur l'accepte, ils deviennent amis et pourront dès lors s'envoyer des invitations. Le joueur peut consulter sa liste d'amis.

#### Gestion discussion des amis

Si deux joueurs sont amis, ils peuvent discuter ensemble.

#### Consultation de l'historique des parties d'un joueur

Un joueur peut consulter l'historique d'un autre joueur ou le sien ainsi que son classement dans chacune des parties effectuées.

**consultation du classement des joueurs**

Un joueur peut consulter une liste des différents joueurs classés selon une mesure ( par exemple, le nombre de victoire).

## 3.2 Exigences non fonctionnelles

**Besoin d'une connexion internet**

Comme le jeu se joue en réseau, une bonne connexion internet est requise afin de communiquer avec le serveur.

**Mise à jour de la base de donnée**

Le programme doit pouvoir enregistrer des (nouvelles) données comme le profil de l'utilisateur (son mot de passe, pseudo) , son nom d'équipe , sa liste d'amis ou encore les noms de ses lombrics.

**Gestion des erreurs**

Quand le programme rencontrera une erreur, il ne s'arrêtera pas brusquement et gérera l'erreur en question.

**Fluidité**

Le programme doit être optimisé afin que pendant une partie il n'y ai que peu de latence et que la connexion entre l'utilisateur et le serveur ne soit pas lente.

# 4 Design et fonctionnement du système

## 4.1 Diagramme de classe

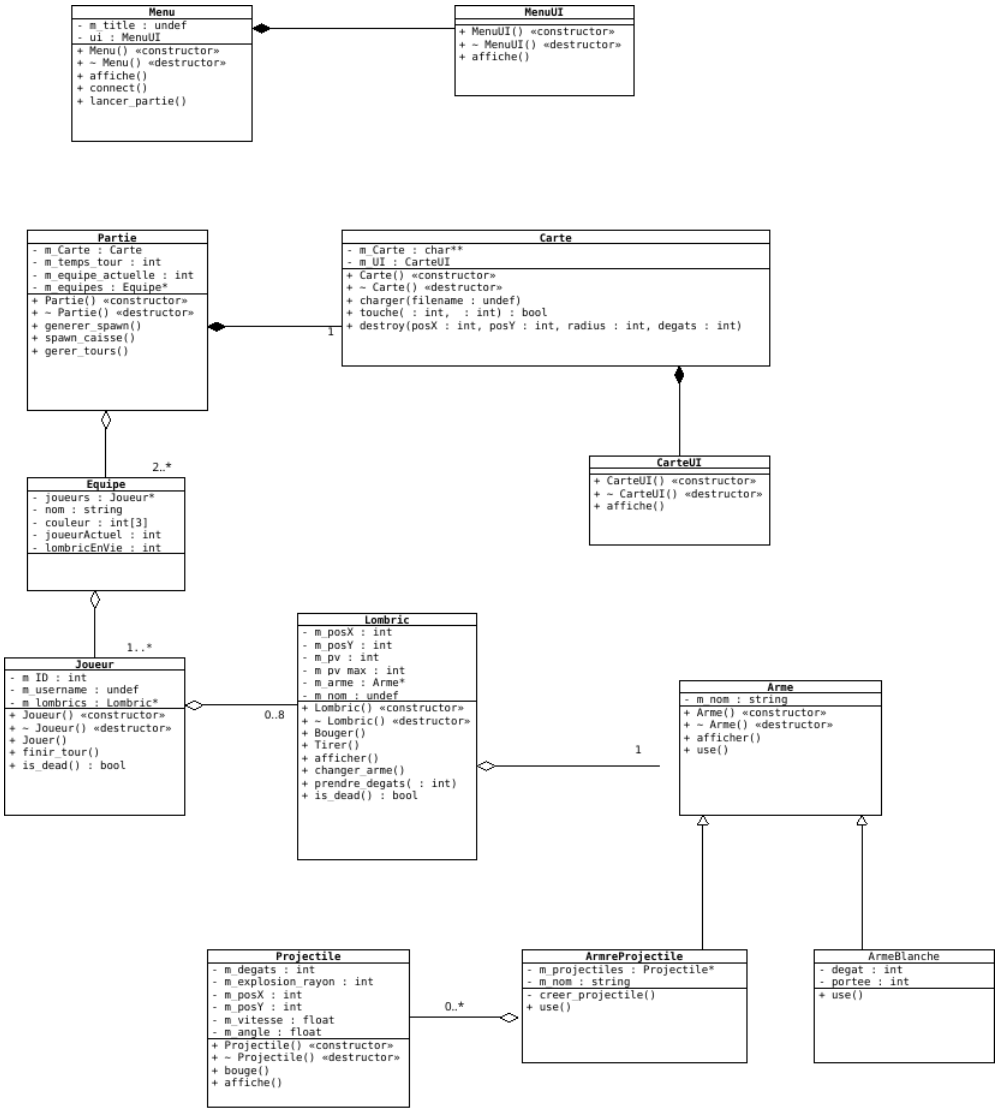


FIGURE 7 – Diagramme de classe

# 5 Modules

## 5.1 Module Client

### Module menu

Le module Menu gère l'affichage du menu, les interactions avec l'utilisateur (entrées clavier, clics de souris) et communique avec le serveur via le module communication.

### Module communication

Le module communication gère la communication avec le serveur, les envois et réceptions de données.

## 5.2 Module serveur

### Module Communication

Le module communication communique avec les différents clients, en envoyant et recevant des messages. Il s'occupe également de reconnecter le client en cas de déconnexion.

### Module "Salon"

Le module salon gère l'attente des joueurs avant la partie. Il permet de modifier les paramètres de la partie qui suit. Un salon contient des équipes, lesquelles contiennent des joueurs. Ces derniers possèdent leurs équipes de lombrics.

### Module Partie

Le module partie représente une partie jouée. Le module gère la carte, les interactions entre la carte et les joueurs le temps alloué à chaque joueur pour jouer son tour et l'apparition de bonus.

### Module Base De Donnée

Ce module gère les interactions avec la base donnée, notamment lorsqu'un joueur s'inscrit, ou qu'une partie se termine. Ce module permet aux autres modules de faire des requêtes pour trouver des joueurs/parties/..., ou d'écrire dans la base de donnée.