Polybasite

## Rapport d’Avancement

## Du 15 Janvier 2018



## STAMEGNA Clément

## MICHON Guillaume

## Sommaire

1. Niveau Conception *page.3 - 6*
2. Niveau Programmation *page.7*

# I. Niveau Conception

Rappel du github du groupe de projet :

<https://github.com/PolyDevTeam/Polybasite>

## Cahier des charges :

Etat : Terminé (Version 2 Modifications possibles)

Le cahier des charges version 1 a été envoyé le 04 / 12 / 2017, grâces aux remarques constructives avec l’enseignant il a été amélioré puis renvoyer en version 2 le 08/12/2017.

## Spécification :

Etat : Terminé (Version 5.0, Modification Possible)

Suites aux conseils de Yves Jehanno nous avons pu bien spécifiés toutes les règles de notre projet.

Voici les différentes évolutions depuis le 18 décembre :

* 24 décembre 2017

Révision et passage à la version 2.0.

Amélioration du diagramme des « Use Case » au niveau de l’IA.

Définitions des règles pour le don de puissance entre mineurs.

Début de la spécification des IA.

* 29 décembre 2017

Révision et passage à la version 3.0.

Spécifications sur la mort des mineurs.

Définitions des règles du jeu.

Amélioration du diagramme des « Use Case » au niveau des actions des mineurs.

Spécification sur la différenciation des camps des IA.

* 08 janvier 2018

Révision et passage à la version 4.0.

Spécification des commandes IA et communication avec le jeu.

* 15 janvier 2018

Révision et passage à la version 5.0.

Ajout de la spécification sur les entités « Trou Noir »

## Cahier de Conception :

Etat : Version 3.0 en cour de rédactions

Notre cahier de conception spécifie la conception de base de notre projet, il peut évoluer tout au long du projet.

Depuis le 18 Décembre 2018.

* 26 décembre 2017

Révision et passage à la version 2.0

Ajout des Descriptions des ressources logicielles.

Ajout de l’organisation de l’espace de travail.

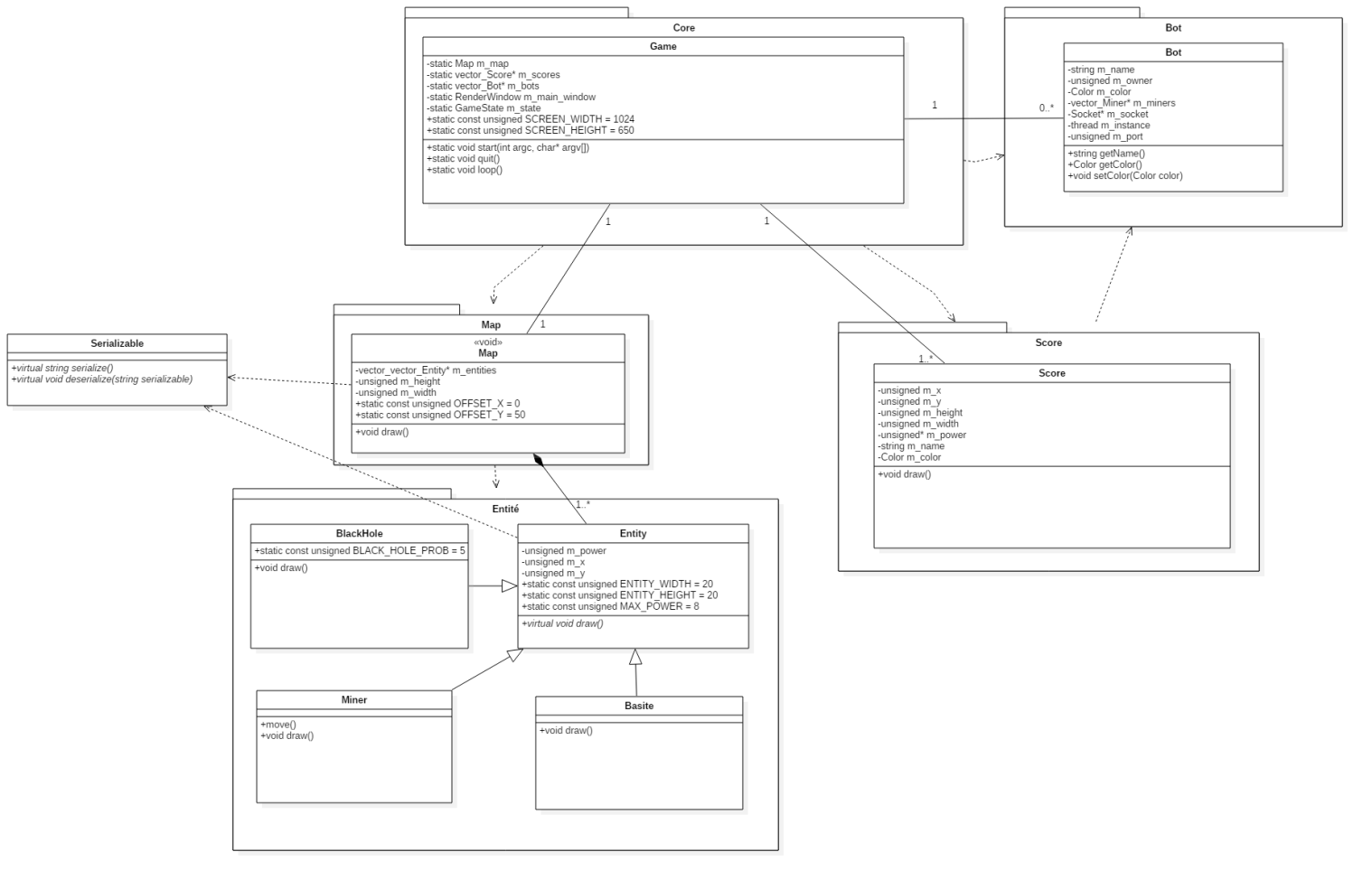
Explication de certaines méthodes via Pseudo Code notamment la méthode loop() de la classe « Game » et la méthode draw() de la classe « Map ».

Nous avons établie la Correspondance entre les Fonctionnalité et les différentes classes.

Amélioration du comportement en erreur attendu de notre jeu.

* 12 janvier 2017

Mise à jour du diagramme de Classe pour introduire l’entité « BlackHole » ainsi que les classes « Scores » et « Bots ».



Ajouts des différents attributs et méthodes pour ces classes.

Mise à jour de la classe principale « Game » pour l’ajout de « Bots » et de « Scores ».

Analyse statique des classe « Entity », « BlackHole », « Basite », « Miner », « Bot » et « Score ».

Il ne reste plus qu’a faire l’analyse dynamique de nos classes et éléments et encore finaliser l’analyse statique.

# II. Niveau Programmation

Rappel du github du groupe de projet :

<https://github.com/PolyDevTeam/Polybasite>

Depuis le 18 Décembre 2018.

Nous avons fini les classes de bases,

Notre classe principale « Game », la classe « Map » qui sert de zone de jeu ainsi que les classes entités « Miner » « Basite » « BlackHole »

Nous avons implémenté un affichage de scores permettant de suivre en temps réel l’évolution des différentes IA.

Pour pouvoir communiquer avec les IA nous avons implémenté la connections avec les IA via une classe « Bot » qui utilise les sockets SFML.

Pour tester ces communications nous avons programmés des clients de test qui serviront de base pour la partie connections de nos IA.

Il ne nous reste qu’a implémenter les éléments d’interactions entres nos entités, ainsi que les commandes que vont envoyer les IA au jeu pour contrôler leurs mineurs telle que spécifié dans nos cahiers.

Nous allons aussi faire des IA simplistes qui agiront aléatoirement.

L’équipe a crée des classes outils pour faciliter l’écriture de certaine partie du code telle que :

- La classe Socket : pour faciliter l’utilisation des socket sfml.

- Les classes Log et Loggable : pour faciliter la création de fichiers de logs.

- Les classes Line et RichText : pour améliorer l’écriture de phrase sur le jeu et les scores.