Ecole Publique d'Ingénieurs en 3 ans

Rapport

ANTS 3000

le 20 décembre 2020,
version 1.1

Loan ALOUACHE | Alexis HESLOUIN

www.ensicaen.fr

Table des matières

Choix de conception **3**

Diagramme de classe 3

Conception 4

Difficultés rencontrées **5**

Réalisations **5**

Fonctionnalités implémentées 5

Fonctionnalités manquantes 5

Répartition du travail 6

Aides extérieures 6

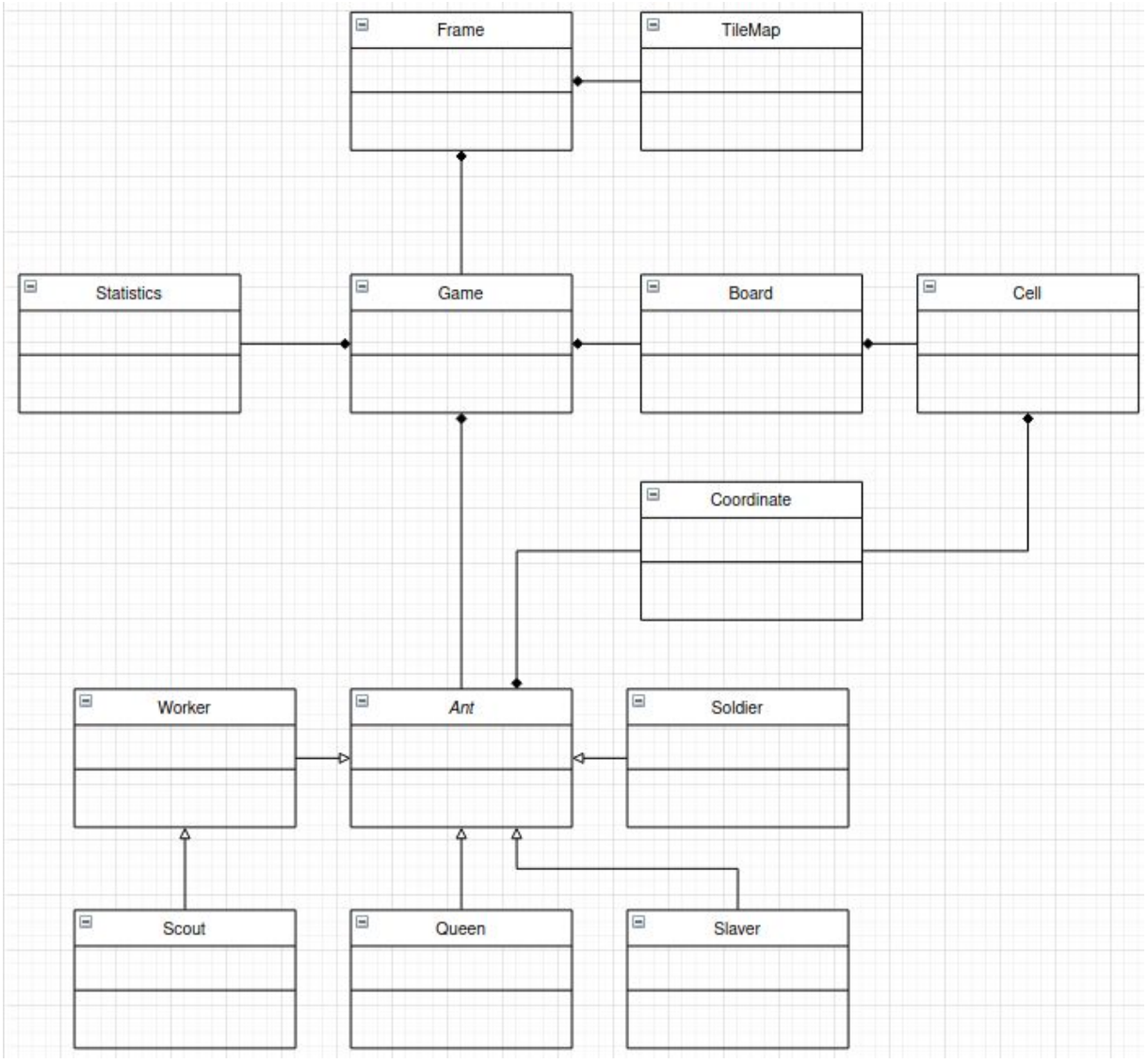
Limitations 6

ANTS 3000

https://gitlab.ecole.ensicaen.fr/lalouache/projet-cpp

1. Choix de conception

1.1. Diagramme de classe



1.2. Conception

La classe Ant possède une méthode abstraite “hourlyAction” que toutes les fourmis implémentent. Cette conception nous permet de tirer pleinement partie du polymorphisme.

Nous sommes conscients que notre architecture n’est pas la meilleure. En effet la classe Ant n’a pas à avoir accès à la classe Game. Cependant, nous apprenons à nos fourmis à ne pas faire n’importe quoi avec la classe Game, et ce dès leur plus jeune âge.

En voici la preuve:

La génération du plateau (obstacles et nourriture) suit les étapes suivantes:

- on essaie de placer un obstacle à X cases
- on sélectionne une case au hasard
- on regarde si la case est libre, de même pour ses 8 voisins
- si non, on choisit une autre case
- si oui, on place un obstacle sur la case et sur X-1 voisins
- si l’obstacle n’est pas placé au bout de plusieurs essais, on passe à l’obstacle suivant
- on réitère l’opération pour le reste des obstacles

En ce qui concerne les couleurs sur le plateau:

- rouge: slaver
- bleu: worker
- blanc: scout
- jaune: soldier
- vert: queen

Nous avons utilisé un tileset pour représenter au mieux le type de la cellule:

- une texture herbe pour les cases “vide”
- des champignons sur une case nourriture
- des rochers pour les obstacles
- une texture “sable” pour la colonie

La simulation commence avec l’ensemble du terrain grisé à l’exception de la colonie. La texture de la case est révélée une fois la case explorée. Sauf dans le cas des obstacles, si le scout explore une case alors tous les obstacles adjacents sont révélés.

2. Difficultés rencontrées

Difficulté	Résolution
La génération des obstacles à retarder l'avancement du projet. La génération du plateau était interminable.	Il s'est avéré que nous essayions de placer beaucoup trop d'obstacles alors qu'il ne restait plus de place disponible. En effet, nous considérions qu'un obstacle de plusieurs cases en tant qu'un seul obstacle et non comme un regroupement de X obstacles.

Le reste des difficultés provient plus d'un manque de connaissance et d'expérience sur le langage que d'un problème d'implémentation.

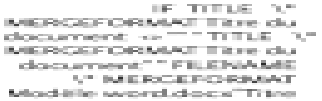
3. Réalisations

3.1. Fonctionnalités implémentées

- Apparition des fourmis grâce à la reine
- Déplacement des fourmis sur le plateau
- Fourmis éclaireuses découvrent le plateau
- Fourmis ouvrières se déplacent dans les zones découvertes
- Fourmis ouvrières récoltent de la nourriture
- Fourmis ouvrières rapportent la nourriture à la colonie
- Fourmis ouvrières déposent des phéromones
- Fourmis soldat tuent les fourmis esclavagistes
- Fourmis esclavagistes apparaissent aux 4 coins de la carte
- Fourmis esclavagistes volent nourriture/larves
- La reine ne fait pas apparaître de fourmis si attaquée la veille
- Génération aléatoires des obstacles de différentes tailles
- Génération aléatoire de nourriture
- Actualisation en temps réel du plateau
- Les fourmis meurent de faim / vieillesse (aka l'afrique)
- Statistiques

3.2. Fonctionnalités manquantes

- Cases phéromonées non suivies
- Chemin optimisé jusqu'à la colonie pour les esclavagistes



3.3. Répartition du travail

Loan	Alexis
Fourmis ouvrières	Génération du plateau
Fourmis reine	Fourmis Soldier
Gestion SFML	Fourmis Scout
Statistiques	Fourmis Slaver
Logique boucle de jeu	Gestion de la nourriture

Il est difficile de séparer complètement les tâches car nous avons tous les deux travaillé sur l'ensemble des fonctionnalités.

3.4. Aides extérieures

Maxence Morin nous a aidé à améliorer le temps de génération du plateau en nous conseillant de limiter les tentatives pour placer un obstacle (on essaie X fois et s'il n'est toujours pas placé, on passe à un autre).

Nous avons également utilisé la documentation SFML:
https://www.sfml-dev.org/documentation/2.5.1/

3.5. Limitations

La durée de vie des Slavers (et le manque d'optimisation du chemin) empêche de certaines des fonctionnalités implémentées d'être utilisées. En effet, les esclavagistes meurent avant d'atteindre la colonie et il n'y a donc aucun vol de larve. De plus, la présence des soldats et le faible nombre de slaver réduisent encore plus les chances de ces événements de se produire.

IF TITLE * MERGEFORMAT Titre du document <> "" TITLE * MERGEFORMAT Titre du document "" FILENAME * MERGEFORMAT Modèle word.docx"Titre du document // PAGE * MERGEFORMAT 5