Projet Logiciel Transversal

Jeu au tour par tour multijoueur :

« Guerre des gangs »

Romain Loury
Matthieu Debarge

Sommaire

Introduction

Ce rapport traite du projet « Guerre des gangs ». Il s'agit d'un jeu vidéo développé dans un objectif d'apprentissage de la programmation logicielle et de ses principes. Un travail de conception, de programmation, d'optimisation et de service réseau sera mis en place au sein de ce projet. Celui-ci revêt un caractère transversal dans la mesure où il sera la mise en application de savoirs provenant de différentes matières telles que le génie logiciel, l'algorithmique, la programmation parallèle. Le projet est construit sur un total de 112 heures de travaux pratiques encadrés, et se conclut par un produit fini, ayant des fonctionnalités avancés, à la fois robuste et évolutif.

1. Objectif

1.1.Présentation générale

L'objectif du projet « Guerre des gangs » est la réalisation d'un jeu de stratégie pouvant se jouer seul contre une intelligence artificielle ou à plusieurs. Le jeu se déroule sur une carte Il s'agit d'un jeu au tour par tour, l'état du jeu n'est modifié qu'à la fin d'un tour après que chaque joueur ait renseigné son coup. Le principe est simple, conquérir les territoires des gangs adverses en déplaçant ses recrues sur une carte.

1.2.Règles du jeu

La carte est composée d'un ensemble de territoires (une trentaine) et de 6 quartiers généraux. Chacun des joueurs (physique ou intelligence artificielle) possède un quartier général, le but du jeu est de conquérir les QG de ses adversaires. Chaque joueur commence avec un certain nombre de recrues dans son QG. Pour conquérir d'autres territoires, le joueur peut envoyer une partie de ses recrues sur des territoires adverses, il s'agit d'une attaque. Lorsqu'un joueur envoie un nombre de recrues supérieur au nombre de recrues présent sur un territoire alors il obtient ce territoire. En revanche si un joueur attaque un territoire adverse ayant plus de recrues, le joueur attaquant se voit perdre toutes les recrues qu'il a engagées dans l'attaque et n'obtient pas le territoire qu'il souhaitait, le territoire attaqué conserve toutes ses recrues, ou une partie en fonction du nombre de recrues l'ayant attaqué. En cas d'égalité le territoire gagnant est attribué aléatoirement. A chaque tour tous les joueurs possèdent 3 déplacements d'un territoire vers un autre. Tout l'intérêt du jeu et de la stratégie est d'essayer d'anticiper les coups des adversaires, pour savoir où placer ses recrues. Après chaque tour, chaque joueur se voit attribuer une certaine valeur d'argent lui permettant d'acheter de nouvelles recrues. Plus le joueur possède de territoires, et de QG, plus il obtient d'argent. Pour ajouter de la stratégie au jeu, il existe des cartes, qui peuvent aussi être achetées par le joueur. Les cartes sont plus ou moins puissante en fonction de leur prix, il existe trois catégories de cartes de puissance différentes, le joueur peut en acheter une seule à chaque tour et ne connait pas ses effets lorsqu'il l'achète, il connait uniquement la puissance de la carte. Les cartes ont des effets sur le jeu, voici une liste non exhaustive d'effets introduits

par application d'une carte : réduction de l'effectif de recrues adverse d'un certain coefficient, ajout d'un certain nombre de recrues à un secteur, annulation des attaques des ennemies sur un ou plusieurs territoires, et d'autres. Le jeu s'arrête lorsque tous les QG sont conquis par un joueur. Lorsqu'un joueur n'a plus de QG la partie est perdue pour lui, même s'il lui reste un territoire qui n'est pas un QG. Il y a 9 cartes de 3 types différents, A, B, ou C

Type A: -$nombre\ de\ recrues\ des\ secteurs\ adjacents \times 0.25$
-$nombre\ de\ recrues\ des\ secteurs\ adjacents + 10$
-$Annulation\ des\ attaques\ provenant\ des\ secteurs\ adjacents$

Type B:-$nombre\ de\ recrues\ des\ secteurs\ adjacents \times 0.50$
-$nombre\ d'alliés\ de\ l'attaque\ \times 2$
-$Annulation\ des\ cartes\ ennemies\ joués\ dans\ les\ secteurs\ adjacents$

Type C:-$nombre\ de\ recrues\ du\ secteur\ attaqué\ \times 0.50$
-$nombre\ d'alliés\ d'une\ attaque\ + 10$
-$Annulation\ des\ cartes\ jouées\ dans\ le\ secteur\ attaqué$

Plusieurs carte de même type et ayant les mêmes effets peuvent être possédés et joués par différent gangs sur la carte

1.3. Conception logiciel

2. Description et conception des états

2.1.Description des états

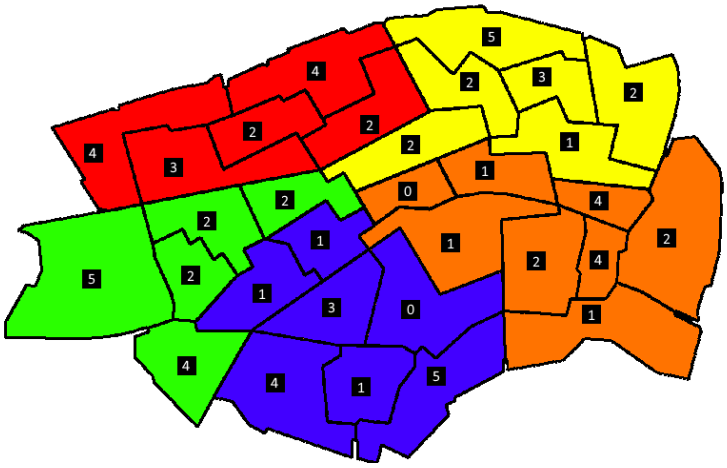
Un état du jeu est défini par l'état de l'ensemble de ses éléments. Tous les éléments sont fixes, la carte ne se déplace pas au cours du jeu, ni les autres éléments. Au cours du jeu, certains attributs évoluent et d'autres sont fixes. Voici la liste des éléments du jeu.

-Elément **« Gang »** : Les gangs sont au nombre de 5 et sont définis par 5 attributs, l'argent disponible gagné au fil des tours, le nombre de recrues correspondant à la somme de tous les recrues sur l'ensemble des secteurs, les cartes possédées, le nom du gang et un booléen définissant si le Gang est encore en vie.

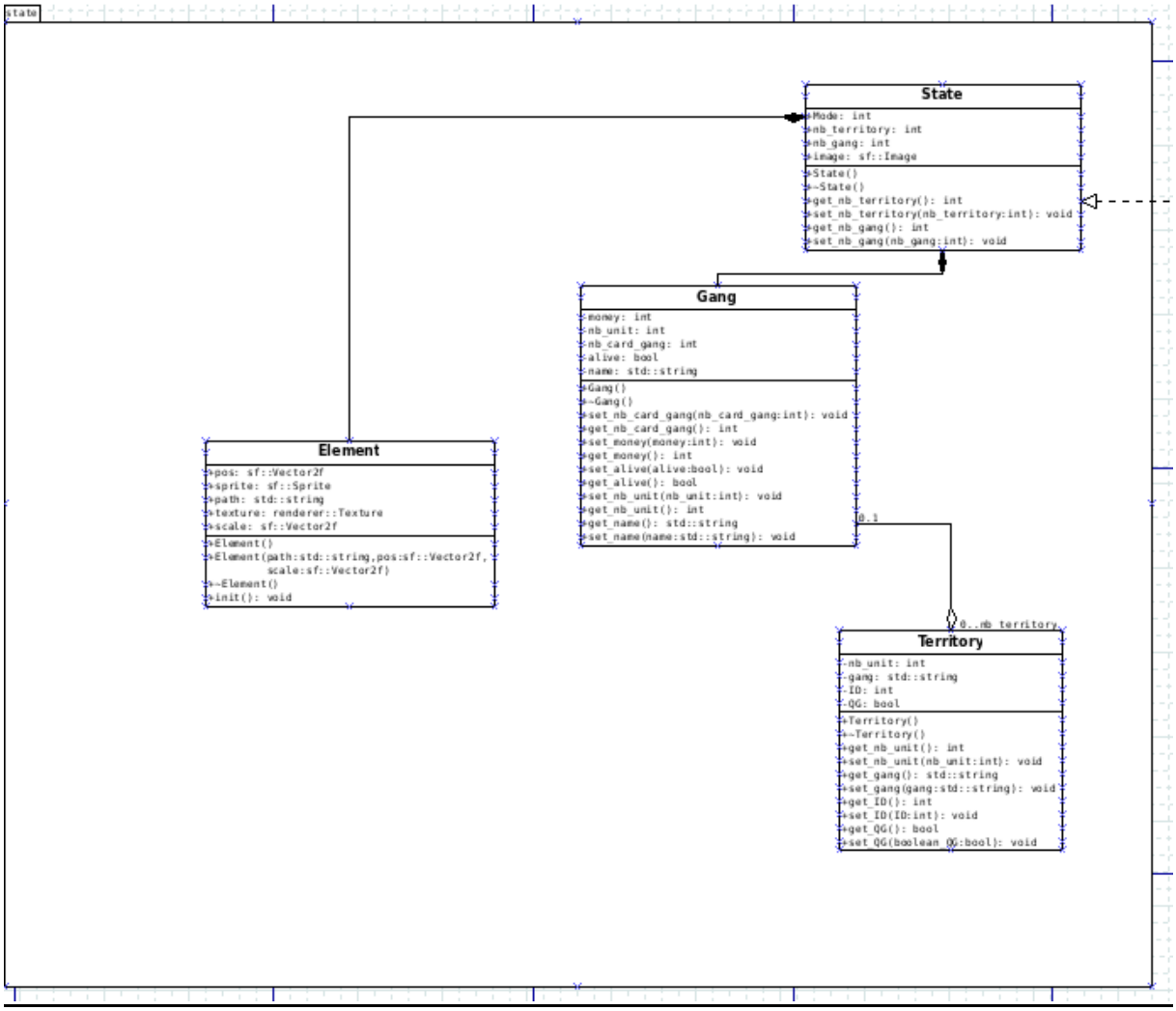
-Elément **« Element »** : Cette classe permet de créer tous les élément qui composent ma fenêtre ainsi que leurs paramètres.

-Elément **« Secteur »** : Il y a 31 secteurs sur la carte chacun possède un nombre de recrues présent sur celui-ci. Le secteur est défini par 4 attributs, le nombre de recrues qu'il possède, le nom du gang auquel il appartient à un instant du jeu, son identifiant permettant de le distinguer des autres secteurs sur la carte. Enfin un booléen permettant de savoir s'il s'agit d'un QG. Les QG sont des secteurs qui possèdent la particularité de voir leur nombre de recrues augmenter à la fin de chaque tour en fonction du nombre de secteurs et de recrues que possède le gang correspondant.

.



Carte représentant les 31 secteurs et leur nombre de recrues lors d'un état du jeu



Graphe d'état state

3. Rendu: Stratégie et conception

3.1.Stratégie de rendu d'un état

Dans un état particulier, le joueur doit pouvoir être informé de l'ensemble des variables de cet état. Le rendu graphique permet d'afficher les variables d'un état faisant ainsi le lien entre les données et l'affichage. Nous avons décidé de séparer le rendu en deux modes d'affichage différents. Le premier mode permet d'afficher la carte et son contenu, une minimap, etc…. La seconde fenêtre se lancera par appui sur la touche B du joueur. Dans ce mode, on affiche un menu boutique permettant au joueur d'acheter de nouvelles cartes ou de nouvelles recrues. On découpe nos fenêtres en plusieurs vues .

3.1.1. Les views

Les views permettent de divisé l'écran et donc de séparer un affichage. Elles sont générer à partir de la classe View.

-Classe « **View** » :elle permet grâce à sa fonction init, d'initialiser toutes les vues soit :

-Une vue pour la map

-Une vue pour la plage commune

-Une vue pour la boutique

-Une vue pour le menu

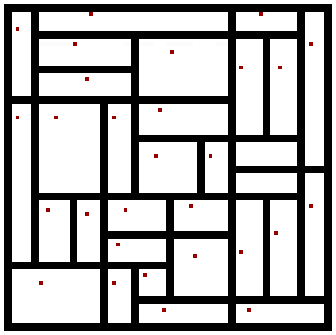
-Une vue pour la minimap

3.1.2. Affichage de la map

L'élément central du jeu est la carte, parce qu'elle permet au joueur de définir ses actions, sa stratégie. Pour créer la carte, nous utilisons le logiciel paint.net qui nous permet d'éditer une image. L'image au format PNG est ensuite traitée par la classe Tile qui est ensuite utilisé par la classe TileMap qui permet de dessiner la carte.

-Classe « **Tile** » : avec un argument « sf ::Image », créer un pointeur sur un tableau à deux dimensions correspondant aux numéros des tuiles associées à chaque pixel de l'image. Les tuiles sont positionnée suivant le placement des frontières. Il nous faut des frontières de deux pixels soit deux tuiles pour créer celle-ci. Une autre fonction de Tile renverra à partir de la même image, la position des nombres permettant d'afficher le nombre de recrues possédées par chaque secteur ainsi que les tours. Le fait d'utiliser une fonction traitant une image permettra de modifier la carte sans avoir à modifier tout le code source et de façon plus simple et rapide.

-Classe « **TileMap** » : est un élément généré à partir du tableau de tuile. Cette fonction permet de spécifier la texture de la map . Cet élément est drawable afin de pouvoir l'afficher via la librairie SFML .



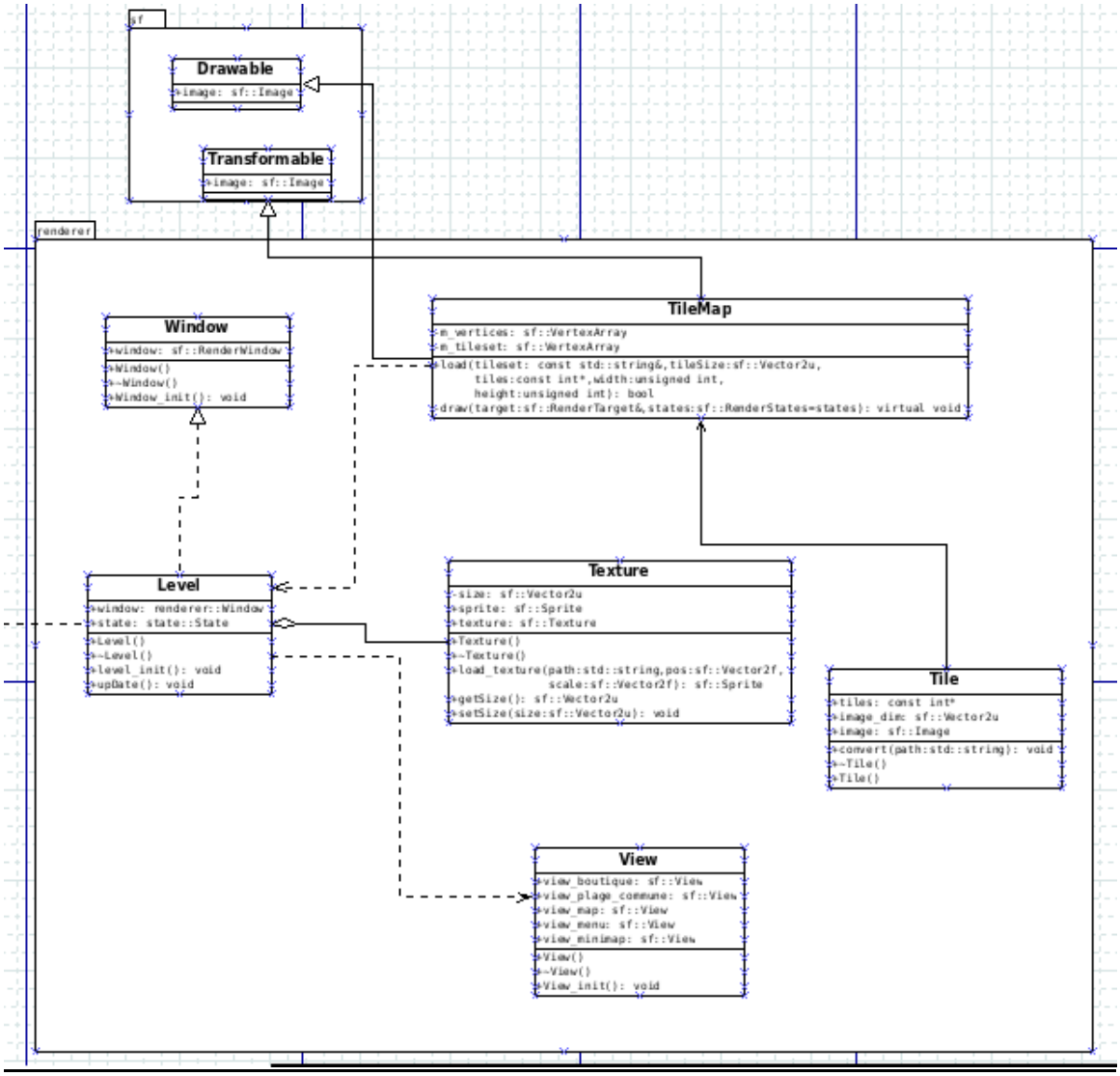
Carte du niveau envoyée par paint

3.1.3. Affichage des élements

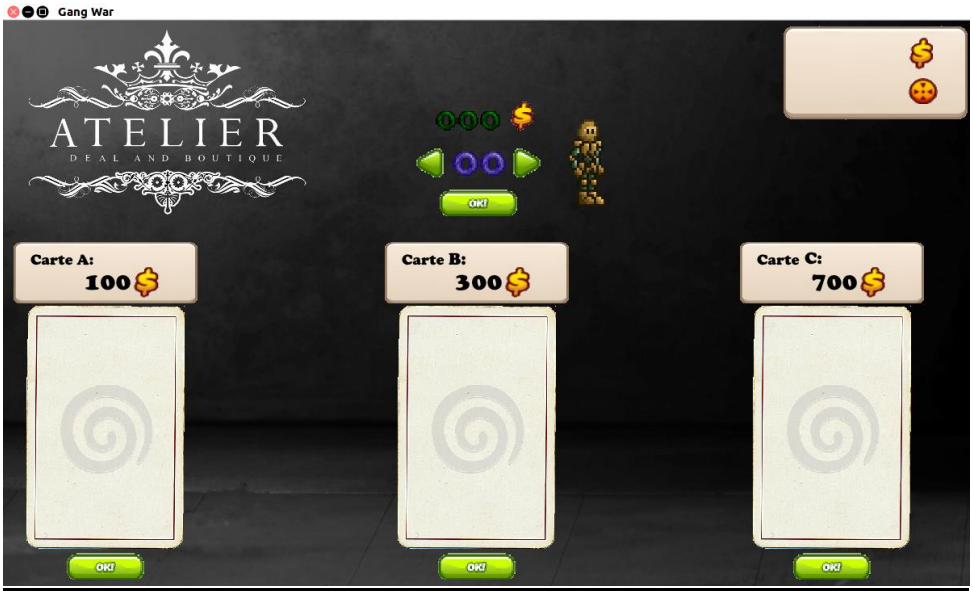
Pour afficher les éléments, on vient les initialiser dans la classe Level afin d'avoir accès à leur état par sa liaison avec la classe State . On vient aussi instancier un fenêtre qui permettra le rendu dans cette classe .

-Classe « **Window** » : cette classe permet de créer la fenêtre dans laquelle on va rendre tout nos vues qui contiennent nos éléments. Elle est initialisé par sa fonction init dans la classe Level. C'est son attribut window qui subira la fonction draw() permettant le rendu avec SFML.

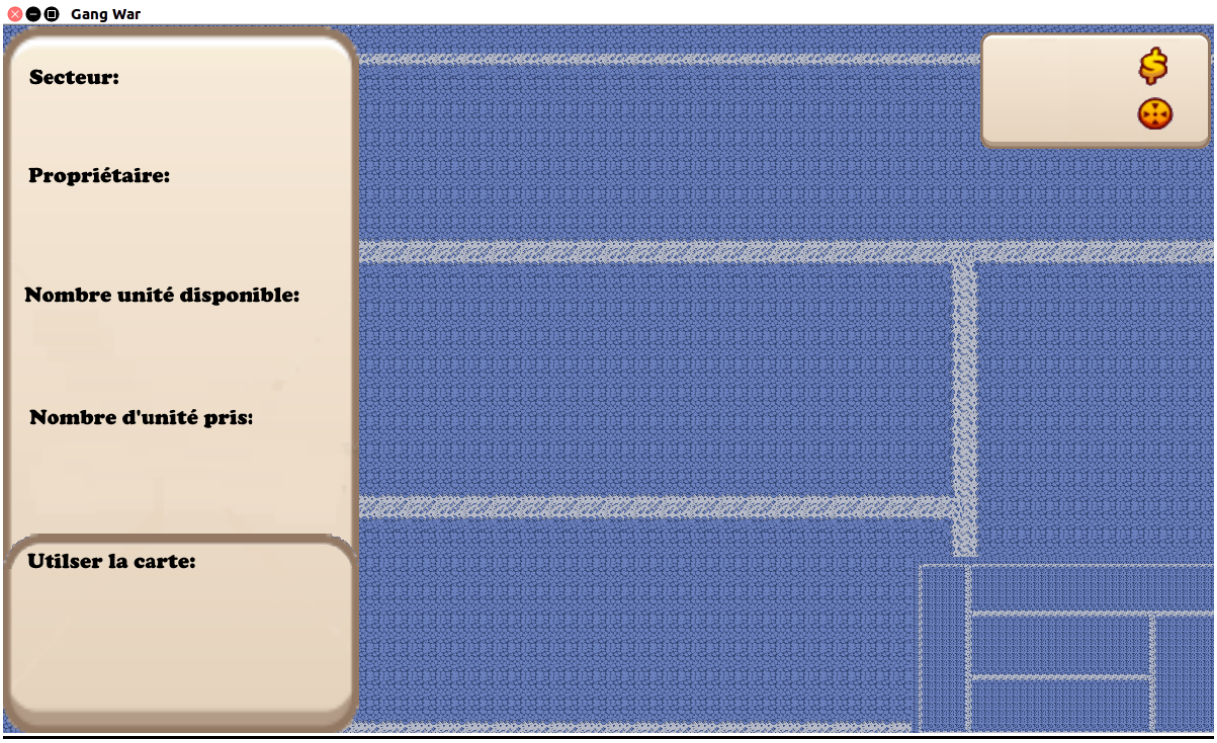
-Classe « **Level** » : Cette classe est la classe maitresse du package et permet d'initialiser tous les éléments, de les définir dans la bonne vue et enfin de les afficher dans la fenêtre. Elle regroupe toute les classes vues précédemment et fait le lien avec la classe State.



Le dia du package renderer



Rendu de la boutique(Mode=0)



Rendu de la Carte (Mode=1)