IMAC Defense

Projet JAVA / IMAC 2 2013

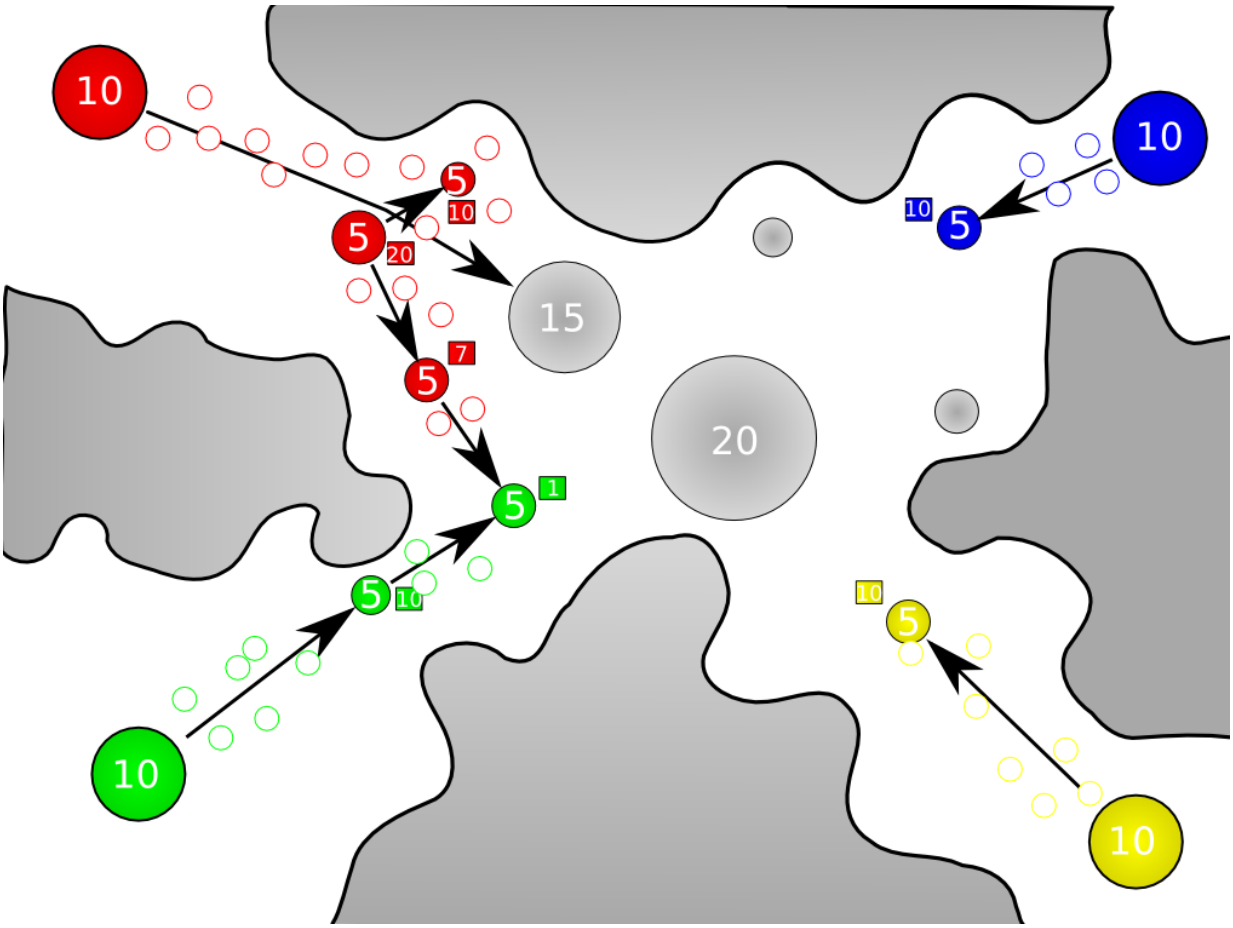
But

Le but de ce projet est de faire un jeu video de style Stratégie Temps Réel (un STR quoi !), mélange entre un Tower Defense et un Nanowars. Ce jeu aura les particularités suivantes :

1. il sera codé en JAVA (surprenant non ?)
2. il aura une magnifique Interface Homme Machine en Swing (ou autre)

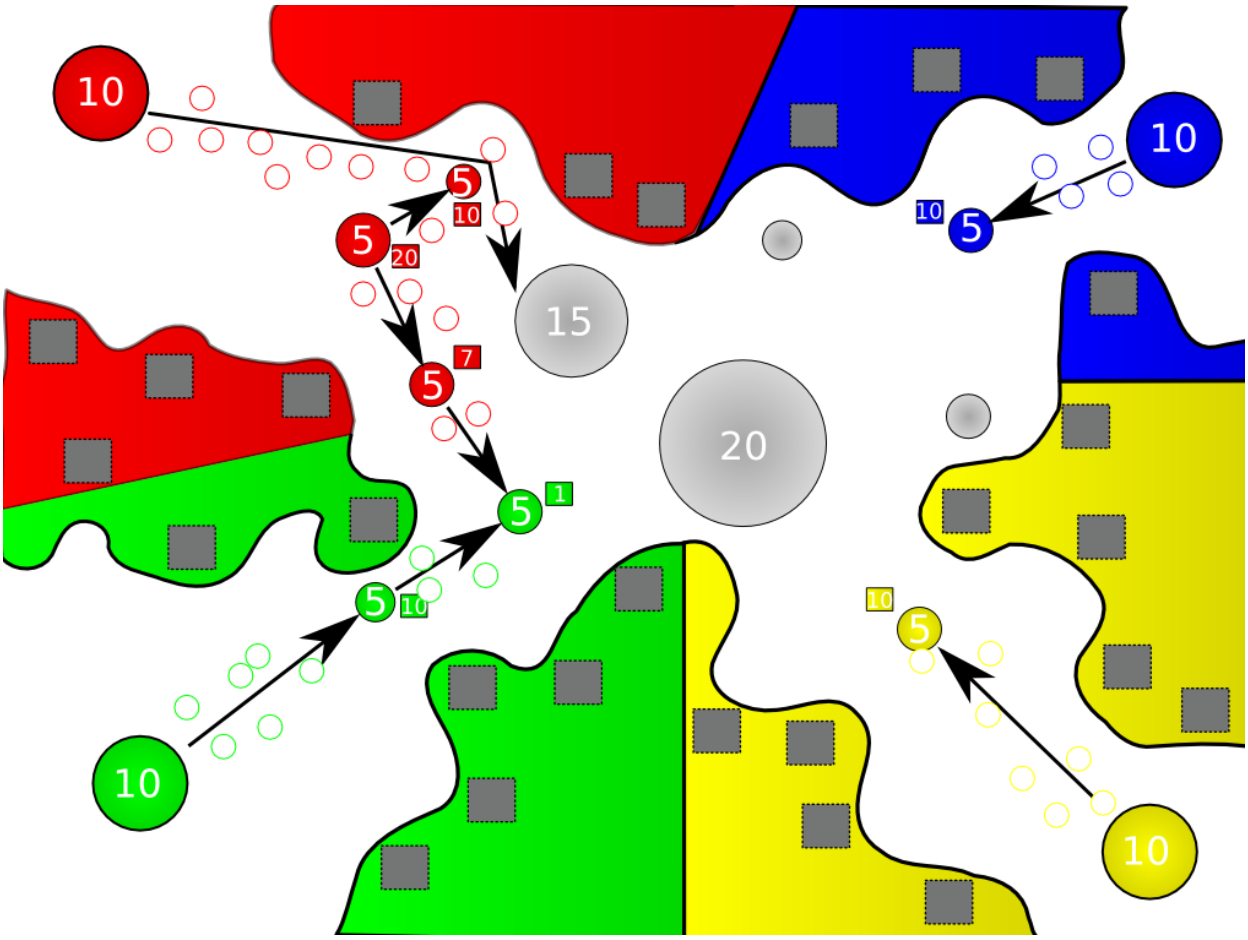
Règles du jeu

La scène du jeu est composée de plaines et de contreforts. Dans les plaines sont disposées des bases, qui sont soit neutres, soit attribuées a des joueurs. Chaque base envoie à intervalles réguliers des agents (dont le nombre est déterminé par sa capacité). Le rôle des agents est de capturer les bases qui ne sont pas sous le contrôle du joueur. Les agents ne peuvent se déplacer que dans les plaines.



Une base neutre est capturée des qu'un agent s'y introduit. Pour capturer une base ennemie, il faut tuer tous les agents ennemis situés à l'intérieur, en y envoyant les siens. Ainsi, pour capturer une base ennemie contenant 10 agents, ils faut y envoyer 11 agents : 10 pour tuer les dix agents ennemis, et un dernier pour prendre le contrôle de la base, qui n'a alors plus de defenses.

Chaque base est associée à une zone de construction, qui correspond a l'ensemble des contreforts se trouvant plus proche de cette base que de n'importe quelle autre. Dans cette zone de construction, le joueur peut placer des tours de défense, de différents types, afin d'empêcher les agents de parvenir a ses bases.



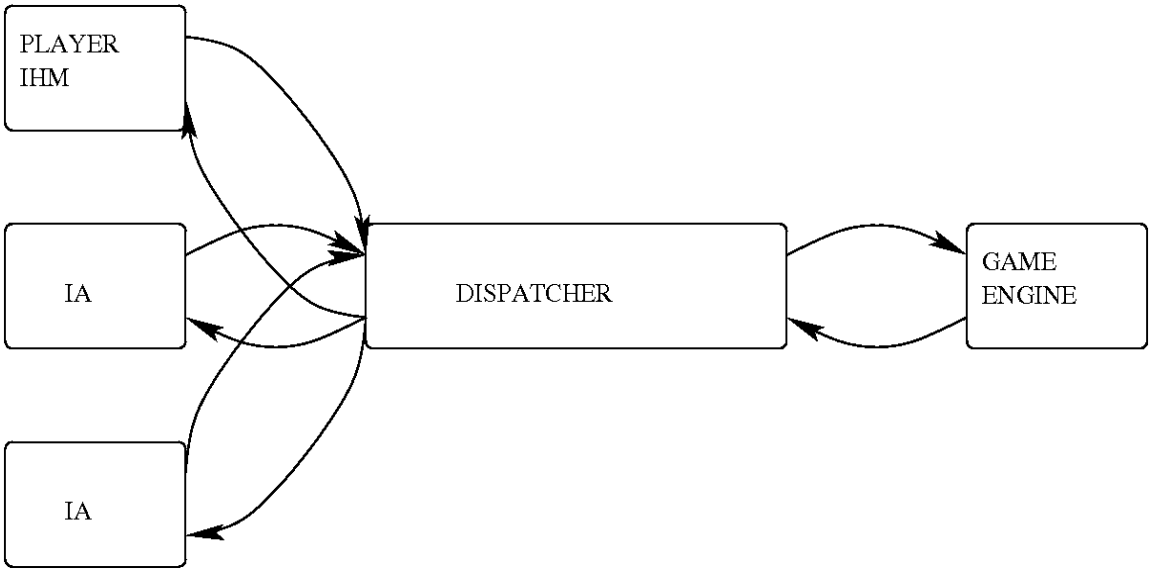
Tout agent tué par une tour rapporte un certain montant d'argent, proportionnel au temps de jeu écoulé. De même, les agents sont de plus en plus robustes, de façon exponentielle par rapport au temps de jeu écoulé. Grace a l'argent gagné, le joueur peut soit placer de nouvelles tours, soit améliorer les tours existentes (zone d'action, puissance, frequence de tir, ...).

Si une base change de propriétaire, le sort des tours de defense associées dépendra du choix du mode de jeu : soit elles seront detruites, soit elles seront alors possédées par le nouvel occupant, soit elles resteront la possession de l'ancien occupant.

Ce qui vous est demandé

Plusieurs parties composent ce jeu :

1. Le moteur du jeu, qui actualise le jeu, deplace les agents, évalue les dégats causés par les tirs des tours de défenses, calcule a qui appartient telle ou telle base, etc...
2. L'interface de joueur, qui permet de prendre connaissance du terrain, et de manipuler ses unités. Elle doit permettre aussi bien a un joueur qu'a une IA de jouer.
3. Le dispatcher, qui permet de gerer les ordres provenant des différents joueurs et de leur fournir les données d'actualisation du jeu.



Le moteur de jeu

Votre moteur de jeu devra être en mesure de gerer une partie, selon les regles ennoncées précedemment. C'est a lui également qu'incombe la responsabilité de verifier si les actions demandées par les joueurs sont valides (si il peut vraiment placer une tour la ou il a demander, si il peut construire tel type de tour, etc...).
C'est également au niveau du moteur de jeu que doit etre gerer le temps, avec ce que ca implique en terme de production de nouveaux agents, de reglage de leur nombre de points de vie et le montant que rapporte leur mort. Le détection de fin de partie et du vainqueur sont aussi des taches du moteur de jeu.
Le moteur de jeu est aussi en charge de pouvoir fournir toutes les informations relatives au terrain aux differents joueurs. Il doit leur fournir a intervalles réguliers la liste des modifications :

1. le deplacement des differents agents
2. les creations et modifications de tours de defense
3. les changement de propriétaires des bases
4. …

L'interface joueur

Vous devez concevoir une interface commune pour les IA et les IHM, et bien entendu concevoir les dites IA et IHM.

L'IHM sera codée au moyen de Swing, et permettra d'afficher la scène de jeu, ainsi que d'interagir avec ses unités, en cliquant dessus et en s'aidant de tout ce qu'il vous semble necessaire d'ajouter (menu, raccourcis clavier, icones, barre de taches).

Chaque joueur/IA tournera dans un thread séparé.

L'IA demandée doit juste etre capable d'effectuer les mêmes actions qu'un joueur humain. **IL NE VOUS EST PAS DEMANDER DE PASSER TROP DE TEMPS DESSUS !!!**

Le dispatcher

Le dispatcher a deux roles : il prend les actions de chaque joueur (humain ou IA) et les restitue au moteur de jeu, et il fourni a chaque joueur la liste des modifications et les messages que lui transmet le moteur de jeu.

L'interface de création de partie

On vous demande aussi de faire une IHM permettant de creer une nouvelle partie, de choisir la carte, les options (libre a vous de les définir), le nombre de joueurs, et dans le cas d'une partie en réseau, la visualisation des joueurs connectés.

Les types de tours de defense

Plusieurs types de tours de defense sont demandés :

- une tour "mitraillette", qui a une grande cadence de tir, mais qui fait peu de dégats
- une tour "gel", qui ralenti les agents touchés
- une tour "laser", dont la puissance augmente avec le temps passé a tirer sur une même cible
- une tour "médicale", qui soigne les agents du joueur passant a proximité
- une tour "bombe", dont les tirs font des dégats de zone

Vous devrez bien entendu gerer des ameliorations pertinentes pour ces differentes tours, et leur assigner un cout.

Les modifieurs d'agents

Plusieurs modifieurs d'agents sont demandés, qui apportent des propriétés supplémentaires aux agents :

- un mod "accélérateur" : la vitesse augmente progressivement, jusqu'a une vitesse max
- un mod "multiplicateur" : génère plusieurs agents avec moins de vie quand l'agent meurt
- un mod "régénérateur" : redonne de la vie à l'agent au cours du temps
- un mod "résistance" : diminue les dégats de certaines tours

Les agents peuvent avoir plusieurs modifieurs. L'attribution de modifieurs aux agents peut dépendre du type de partie: "aléatoire", "chaque joueur choisi un mod", "la capture d'une base spécifique applique un mod à chaque agent du joueur", etc...
Chaque modifieur augmente le montant délivré à la mort de l'agent associé.

Fonctionnalités a implanter

Pour resumer :

1. Le programme doit fournir une implementation temps réel des règles du jeu.
2. Les differents types de tours de defenses demandés doivent être disponibles.
3. Les differents types de modifieurs demandés doivent être disponibles.

4. Les joueurs peuvent être des intelligences artificielles.
5. Le moteur de jeu doit pouvoir charger une carte à partir d'un fichier texte, d'une image, aléatoirement ou un peu des trois.
6. Le moteur de jeu doit pouvoir vérifier la validité de la carte chargée.
7. Le moteur de jeu doit fournir une IHM de création de partie.
8. Le moteur de jeu doit pouvoir attribuer une base à chaque joueur connecté.
9. Le joueur peut être acteur ou spectateur.

Evaluation

Vous serez évalués sur le respect des contraintes détaillées et sur l'implémentation des fonctionnalités présentées dans le sujet.
Vous fournirez un rapport qui constituera une partie non-négligeable de l'évaluation dans lequel vous détaillerez les fonctionnalités implémentées. Le rapport devra contenir un nombre conséquent de capture d'écrans issus de votre application ainsi qu'un manuel précis pour la compilation et le lancement de vos programmes.

La date de rendu est le 30 mai a minuit.