|  |
| --- |
|  |
| **ASD2 – Cahier des charges** |
| **ASD-Tower-Defense** |
|  |
| mercredi 2 décembre 2009 |
|  |

|  |
| --- |
| **logo_heig.gif logo_hes_so.jpg** |

Sommaire

[1. Objectifs 2](#_Toc247516046)

[2. Présentation de l’application 3](#_Toc247516047)

[3. Technologies utilisées 4](#_Toc247516048)

[*1.* *Langage* 4](#_Toc247516049)

[*2.* *Librairies externes* 4](#_Toc247516050)

[*3.* *Implémentation* 4](#_Toc247516051)

[4. Répartition des tâches 4](#_Toc247516052)

[5. Planification 5](#_Toc247516053)

# Objectifs

* Illustrer le concept de graphe de manière ludique et interactive.
* Acquérir de l’expérience dans la planification et l’accomplissement d’un projet conséquent.
* Utiliser et découvrir des librairies existantes implémentant le concept de graphe
* Apprendre à mettre en œuvre une interface graphique en Java.
* Séparer le travail en plusieurs niveaux d’abstraction pour faciliter l’élaboration et l’évolutivité de ce projet.
* Comprendre la nécessité d’utiliser des algorithmes complexes dans les applications informatiques.
* Mettre en œuvre un algorithme de recherche de chemin le plus court (ACPC).

# Présentation de l’application

Nous allons présenter notre concept de maillage dynamique sous la forme du célèbre jeu *Tower Defense*, très populaire sur Internet avec de nombreuses variantes tant en Flash ou autre que de *mod* intégré à des jeux commerciaux.

La variante que nous allons implémenter se comporte de la manière suivante : une zone de jeu très simple sera proposée à l’utilisateur avec une série de créatures allant d’un point de départ à un point d’arrivée par le chemin le plus court à travers la zone. L’utilisateur devra placer des obstacles (sous forme de tours) sur le chemin des créatures pour dévier leur trajectoire. Ces tours infligent des dégâts aux créatures tant qu’elles sont à portée. Elles ont donc deux fonctions : celle de bloquer le chemin et celle de blesser et tuer les créatures. Le but du jeu étant bien sûr d’établir une stratégie de placement des tours pour que les créatures ne rejoignent pas en vie le point d’arrivée, en leur laissant pourtant un chemin possible.

C’est dans le calcul des chemins possibles pour chaque créature qu’interviendra la structure de graphe avec les algorithmes associés. Nous allons en effet mailler la zone de jeu avec un graphe pondéré sur lequel nous allons appliquer des transformations pour schématiser la mise en place de tours. Sur cette base, chaque créature devra en fonction de sa place dans le graphe trouver le chemin le plus court vers le point d’arrivée en utilisant un algorithme ACPC.

Choix des tours

Départ

Arrivée

Dégats : 10

Cout  : 5

Dégats : 20

Cout  : 8

Dégats : 30

Cout  : 15

Légende

Créature

Chemin

optimal

Tour

*Schéma représentatif de l’application*

# Technologies utilisées

## *Langage*

Pour ce projet, nous utilisons le langage Java car c’est le langage le mieux adapté à nos capacités et à nos connaissance. D’autant plus, le concept d’interface graphique est facile à mettre en œuvre dans ce langage.

## *Librairies externes*

Nous utiliserons, si elle existe, d’une librairie implémentant le TDA graphe de manière performante et complète (librairie codée en Java).

Nous réaliserons l’interface graphique grâce à la librairie Swing intégrée à Java.

## *Implémentation*

Nous veillerons à respecter le design pattern MVC (*Model* – *View* – *Controller*) qui structure notre programme en 3 couches principales, nous permettant d’avoir un niveau d’abstraction confortable pour un travail collaboratif et un programme final évolutif.

Nous veillerons également à respecter les principes de la Programmation orientée objet qui nous ont été enseignés dans un cours parallèle.

# Répartition des tâches

Nous prévoyons de nous répartir les tâches de manière suivante entre chaque entité du groupe.

1. *Chef de projet*
   1. Rédactions, administrations
   2. Suivi des rendus (*deadlines*)
   3. Surveillance et coordination
   4. Développement
2. *Membre 1*
   1. Création de l’interface graphique
   2. Gestion de l’affichage
   3. Rendu graphique
   4. Interaction avec l’utilisateur
3. *Membre 2* 
   1. Algorithmique
   2. Implémentation des algorithmes de graphe
   3. Fournir les briques logicielles pour permettre la construction de la partie fonctionnelle de l’application

# Planification

Le projet se déroulera du 18 novembre 2009 au 15 janvier 2010, ce qui représente un total de 25 périodes en classe (par personne). Nous prévoyons également de passer un total d’environ au moins 25 périodes par personne en dehors des heures encadrées.

Au total, c’est environ 150 périodes de travail que nous allons planifier comme suit :

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **nov.09** | | **déc.09** | | | | | **janv.10** | | |
| **Taches** | **18** | **25** | **2** | **9** | **16** | **23** | **30** | **6** | **13** | **15** |
| Analyse préliminaire |  |  |  |  |  |  |  |  |  |  |
| Conception |  |  |  |  |  |  |  |  |  |  |
| Réalisation |  |  |  |  |  |  |  |  |  |  |
| Tests |  |  |  |  |  |  |  |  |  |  |
| Documentation |  |  |  |  |  |  |  |  |  |  |
| Planification |  |  |  |  |  |  |  |  |  |  |

5