

# Rapport de TP 4MMPOO : Simulation Orientée-Objet de systèmes multiagents

MAHIEU Lucas (SL\_PHELMA)  
DEVALON Hugues (SLE\_PHELMA)

10 novembre 2015

**Préambule** L'objectif de ce TP est de développer en Java des applications permettant de simuler de manière graphique des systèmes multiagents. Dans un premier temps, nous nous intéresserons à trois systèmes de type automate cellulaire : le jeu de la vie de Conway, un jeu de l'immigration, et le modèle de ségrégation de Schelling. Dans un second temps, nous nous intéresserons à la simulation d'un système de mouvement d'essaims auto-organisés : le modèle de Boids.

## 1 Automates Cellulaires

### 1.1 Jeu de la vie de Conway :

Nous présentons en figure 1 page 1 la représentation UML des classes permettant la Simulation de l'automate cellulaire du jeu de la vie de Conway. Nous avons explicité les attributs privés des classes pour que vous compreniez mieux nos choix de conception.

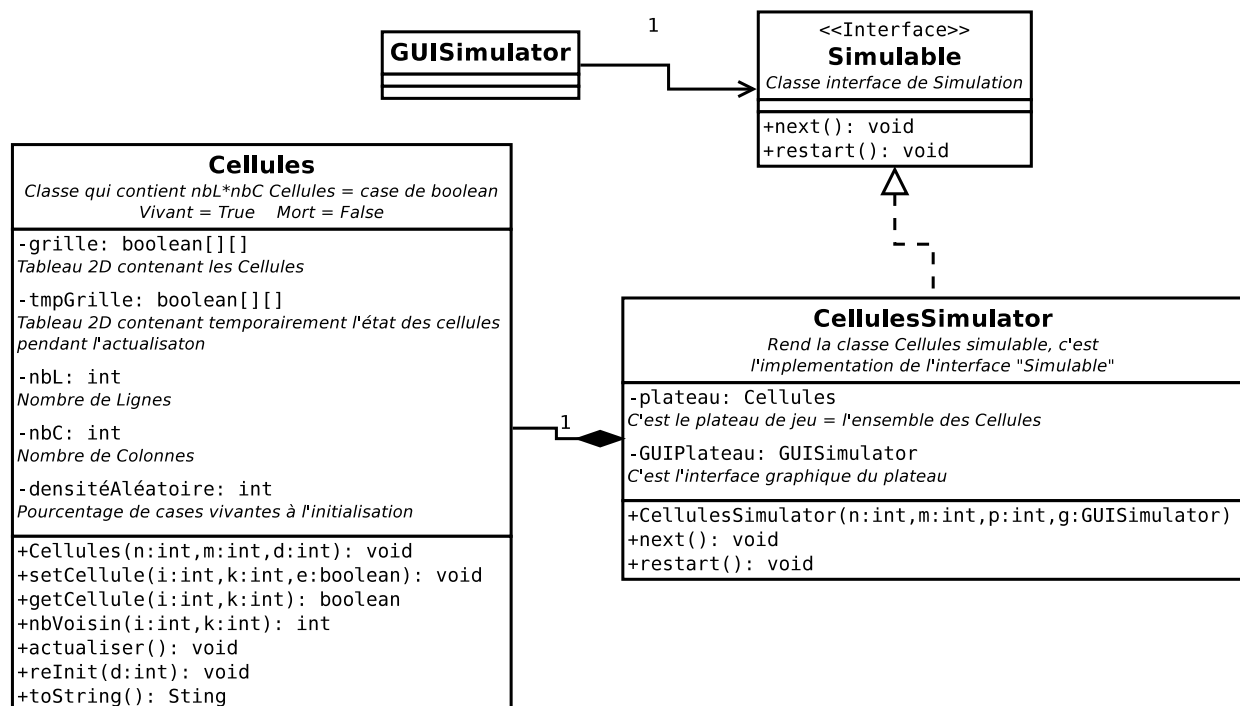


FIGURE 1 – Diagramme des classes du jeu de la vie de Conway

Comme vous pouvez le constater, nous avons utilisé des classes assez 'génériques' pour qu'elle soit facilement réutilisables pour les autres jeux. D'ailleurs on peut remarquer que notre conception est presque identique à celle proposée pour le jeu de balles. En effet, ici nous manipulons un tableau de *boolean* et non plus un tableau de *Point*. Ainsi pour l'adapter à d'autres jeux, il suffirait de modifier les booléens par des entiers, des couleurs, ou autre états plus complexes.

Pour tester ces classes, nous avons utilisé deux classes de tests : l'une pour faire une simulation textuelle, l'autre pour tester la simulation graphiquement.

## 1.2 Jeu de l'immigration :

Les seules différences avec le jeu de la vie sont

- Qu'une cellule peut prendre un nombre  $n_e$  d'états et non plus seulement 2.
- Les règles de changement d'états.

Du fait que notre conception est générique, nous avons réutilisé le jeu de la vie en y ajoutant une seule chose : le nombre d'états possibles pour une cellule. Ainsi nous pouvions supprimer l'attribut de densité utile à l'initialisation du jeu de la vie, et modifier la règle dans la méthode `void actualiser()`.

Nous avons fait le choix de créer une correspondance entre les *Entiers* qui représentent l'état d'une cellule et une *Couleur* qui sera affichée dans sa simulation graphique.

## 1.3 Le modèle de Schelling :

A TOI DE COMPLETER CETTE PARTIE LA POUR EXPLIQUER CE QUE TU AS UTILISEE ET PQ

# 2 Un modèle d'essaims : les *boids*

## 2.1 Conditions expérimentales

*Décrire les conditions permettant la reproductibilité des mesures : on demande la description de la machine et la méthode utilisée pour mesurer le temps.*

### 2.1.1 Description synthétique de la machine :

*indiquer ici le processeur et sa fréquence, la mémoire, le système d'exploitation. Préciser aussi si la machine était monopolisée pour un test, ou notamment si d'autres processus ou utilisateurs étaient en cours d'exécution.*

### 2.1.2 Méthode utilisée pour les mesures de temps :

*préciser ici comment les mesures de temps ont été effectuées (fonction appelée) et l'unité de temps ; en particulier, préciser comment les 5 exécutions pour chaque test ont été faites (par exemple si le même test est fait 5 fois de suite, ou si les tests sont alternés entre les mesures, ou exécutés en concurrence etc).*

## 2.2 Mesures expérimentales

*Compléter le tableau suivant par les temps d'exécution mesurés pour chacun des 6 benchmarks imposés (temps minimum, maximum et moyen sur 5 exécutions)*

	coût du patch	temps min	temps max	temps moyen
benchmark1				
benchmark2				
benchmark3				
benchmark4				
benchmark5				
benchmark6				

FIGURE 2 – Mesures des temps minimum, maximum et moyen de 5 exécutions pour les 6 benchmarks.

## 2.3 Analyse des résultats expérimentaux

*Donner une réponse justifiée à la question : les temps mesurés correspondent-ils à votre analyse théorique (nombre d'opérations et défauts de cache) ?*

### **3 Question : et si le coût d'un patch était sa taille en octets ? (1 point)**

*Préciser le principe de la résolution choisie (parmi celles vues en cours) ; donner les modifications à apporter (soit à vos équations, soit à votre programme, au choix) pour s'adapter à cette nouvelle fonction de coût.*