10/10/2017

Mini Projet Java

Sujet n° 2 : Tri collectif

Thafsouth ADLI
Gaétan ROGER

INTRODUCTION

Dans le cadre du fin du module de la programmation orientée objet, nous avons réalisé un mini projet. Nous avons choisi le deuxième sujet qui consiste en la réalisation d'un simulateur de robots autonomes qui évoluent dans un environnement de type grille, contenant des resources qu'ils peuvent ramasser et déposer. Dans un premier temps, nous avons entamé la phase de conception du système de tri collectif avec le language UML (et l'outil Visual Paradigm). Dans un second temps, nous avons implémenté ce que nous avions conçu.

CONCEPTION

Diagramme de classe

La figure 1 représente le digramme de classe du noyau fonctionnel du tri collectif ainsi que les choix conceptuels que nous avons faits. Sur ce diagramme, certaines classes ne sont délibérément pas représentées, telles que la classe « Terminal » qui ne s'occupe que de l'affichage console du jeu.

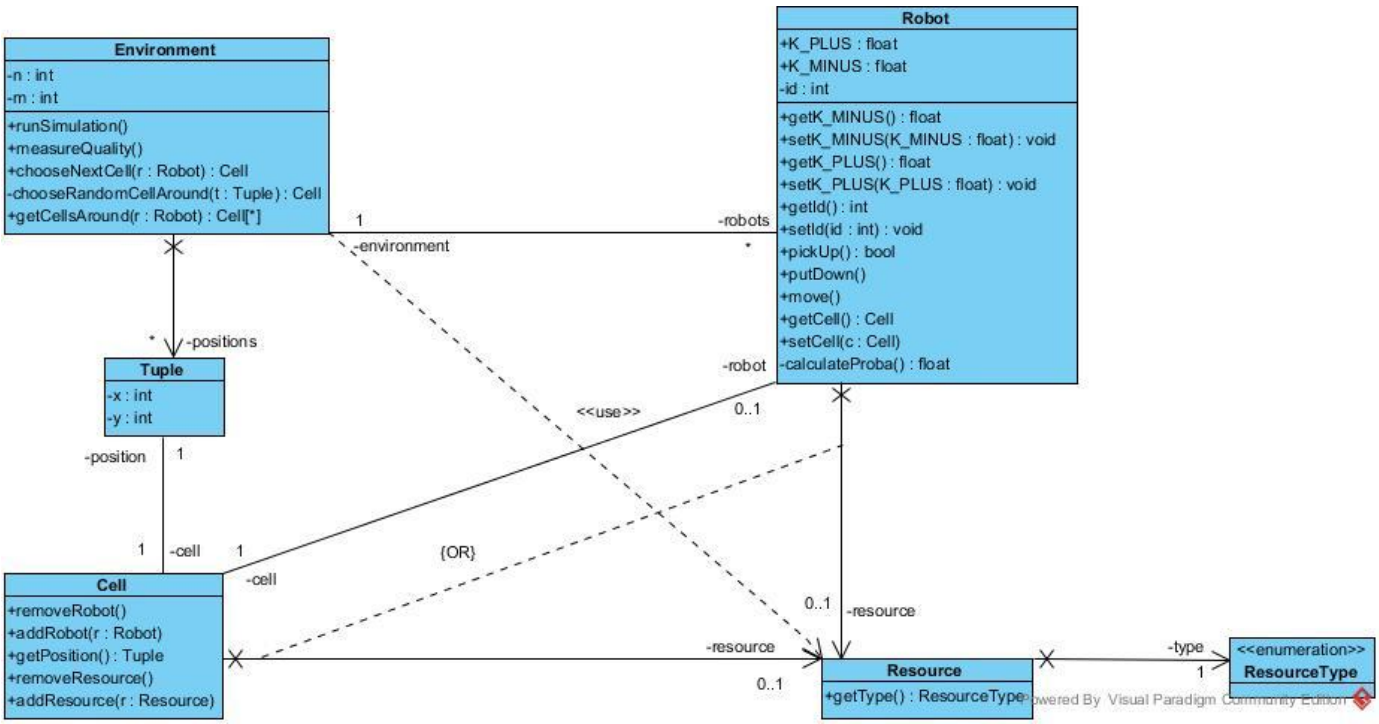

Figure 1 : Diagramme de classe

Diagrammes de séquences

Afin de bien définir le fonctionnement général du jeu, ainsi que le comportement de chaque élément du noyau fonctionnel, nous avons réalisé les diagrammes de séquences pour chacune des méthodes principales.

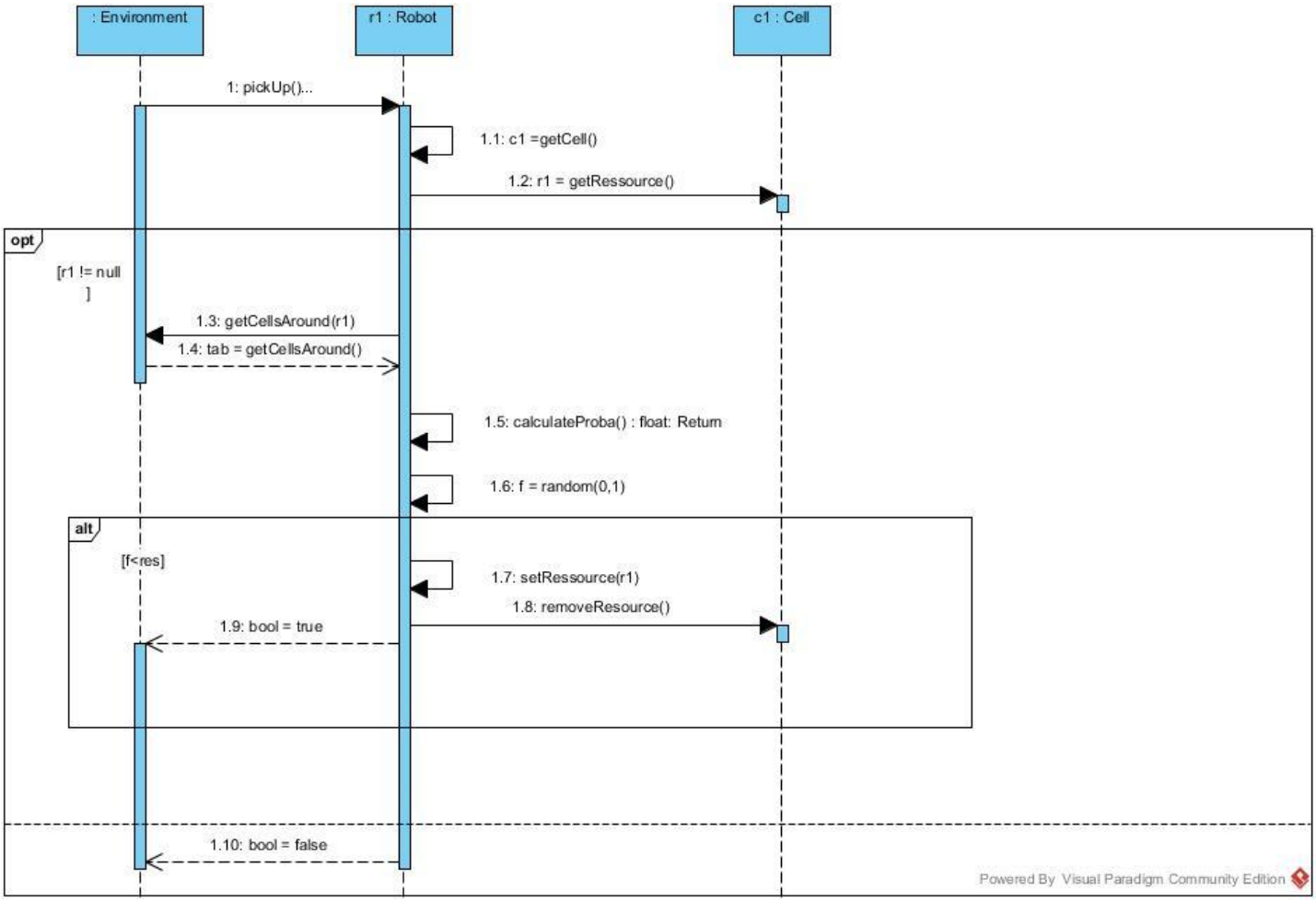

Figure 2 : Méthode *pickUp()*

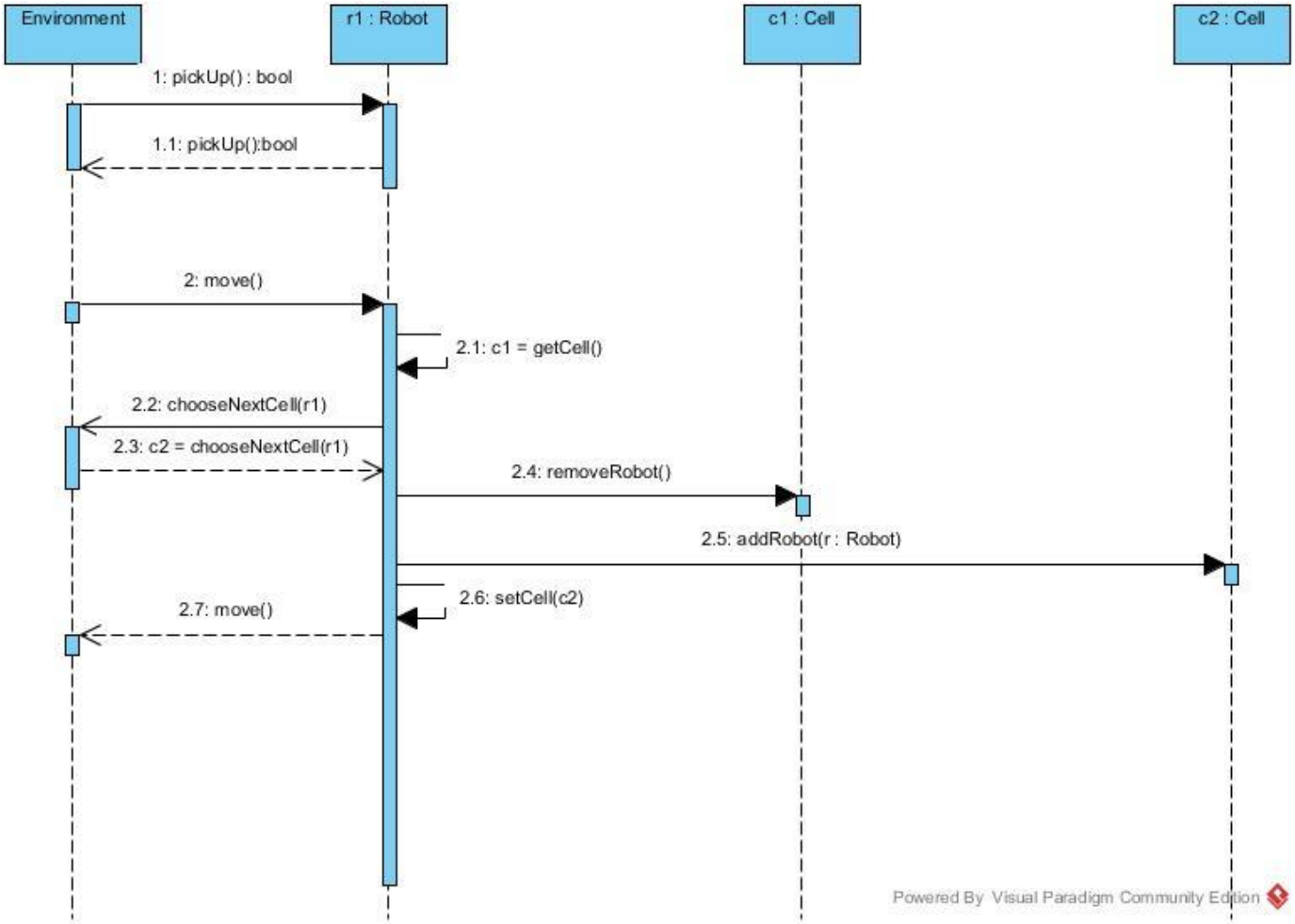

Figure 3 : Méthode *move()*

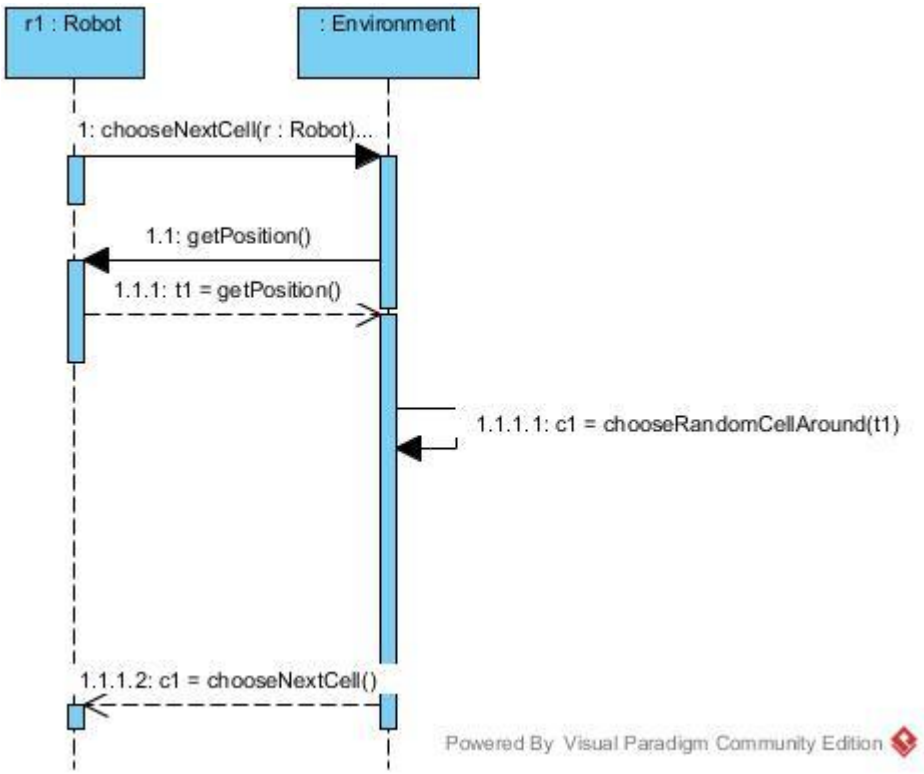

Figure 4 : Méthode *chooseNextCell()*

REALISATION

Commande d'exécution

Afin de lancer le jeu, il suffit d'exécuter la commande *java -jar robots.jar* dans le dossier *dist/1.X/.* Voici une liste de tous les arguments qui sont acceptés avec cette commande. Cette liste peut également être obtenue en exécutant la commande *java -jar robots.jar --help*

Arguments :

- *n=[1-9]+* → Largeur de la grille
- *m=[1-9]+* → Hauteur de la grille
- *robots=[1-9]+* →Nombre de robots sur la grille
- *resources=[1-9]+* → Nombre de ressources de chaque type sur la grille
- *cycles=[1-9]+* →Nombre de cycles de simulation
- *print=true|false* →Dessin la grille à chaque étape de la simulation ou non
- *steps=[1-9]+* → Nombre de tour de simulation avant chaque dessin
- *delay=[1-9]+* → Nombre de millisecondes entre chaque dessin
- *drawrobots=true|false* → Dessine ou non les robots
- *drawheldresources=true|false* → Dessine ou non les ressources que portent les robots

EN COURS D'EXECUTION

La figure ci-dessous représente l'affichage console de l'étape initiale de l'exécution de jeu par la commande : *java -jar robots.jar print=false drawrobots=false drawheldresources=false cycles=100000*

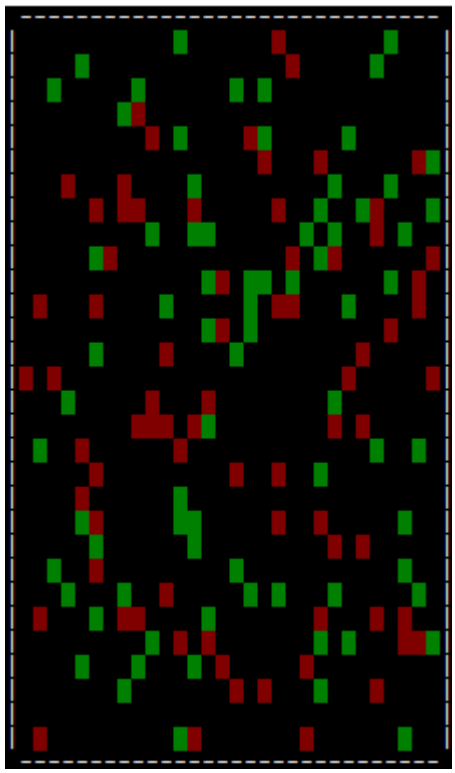

Figure 5 : Jeu en cours d'exécution

A la fin de l'exécution, on voit les tas de ressources qui sont formés.

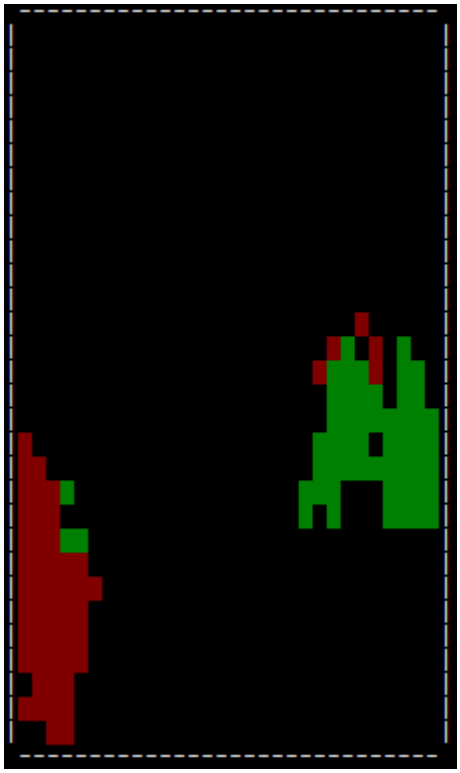

Figure 6 : Résultat final

CONCLUSION

Nous avons pu finir la conception des diagrammes sous Visual Paradigm et le code Java associé permettant de faire fonctionner le tri des ressources accompagné d'un fichier de log permettant de gérer les bugs. Le programme est complet et fonctionnel, Avec plus de temps, nous aurions pu réaliser une interface graphique au jeu et une IA dirigeant plus précisément les robots.