

Ant Legion : Le Projet de Simulation Multi-Agents

ARCHITECTURE LOGICIELLE ET QUALITE

Hadrien Wilbert & Anaël Courjaud | S2 ZZ2 F2 | 16/01/2023

(source : <http://www.dictionnaire-amoureux-des-fourmis.fr/JJ/jeux/Ant-legion/Ant-Legion-affiche.jp>)



Table des matières

I.	Choix du sujet, objectifs	2
II.	Cahier des charges / analyse des besoins.....	2
A.	Choix techniques.....	2
B.	Listing des fonctionnalités essentielles	2
C.	Fonctionnalités ambitieuses	2
III.	Logiques de fonctionnement des principales classes	3
A.	Principe du système de phéromone	3
B.	Diagramme d'activité classe fourmi	4
C.	Diagramme d'activité sous-classe soldate.....	4
D.	Diagramme d'activité sous-classe ouvrière	5
IV.	Planning	6

I. Choix du sujet, objectifs

Pour réaliser une SMA (simulation multi-agent), il faut avoir plusieurs agents avec des objectifs distincts et une capacité à interagir entre eux. La modélisation du fonctionnement d'une fourmilière nous a donc semblé adapté et rigolo. Nous allons principalement nous concentrer sur la politique extérieure de la fourmilière et tenter, avec des règles simples, de faire reproduire aux agents les comportements observés dans la Nature afin de rendre le système viable. Nous nous appuierons sur l'observation des résultats pour adapter notre modèle, si besoin.

II. Cahier des charges / analyse des besoins

A. CHOIX TECHNIQUES

Nous avons choisi de développer le logiciel en C++ sur Linux avec une compilation séparée, et nous utiliserons GitLab et Doxygen pour la collaboration et la documentation. En parallèle, Hadrien découvrira l'utilisation de Jenkins pour assurer l'intégration continue et Anaël approfondira de son côté l'implémentation de fichiers make et cmake afin de contrôler finement la compilation.

B. LISTING DES FONCTIONNALITES ESSENTIELLES

Notre projet comporte 3 points essentiels : La carte, les fourmis et les phéromones.

Les fourmis se baladent sur la carte à la recherche de nourriture et s'orientent grâce à diverses phéromones. Lorsqu'elles se déplacent, en fonction de l'état dans lequel elles sont, elles déposeront un type de phéromone particulier. Si elles suivent un chemin de phéromones, elles suivront toujours le chemin avec le plus haut niveau. En cas de rencontre entre deux fourmis, alors elles communiquent, par exemple pour identifier à quelle fourmilière elles appartiennent.

La carte est générée aléatoirement (Mersenne Twister évidemment), avec la fourmilière et les obstacles. De la nourriture est placée aléatoirement et à intervalles temporels réguliers afin de renouveler l'environnement. Les fourmis peuvent récolter une fraction de chaque pièce de nourriture pour la ramener à la fourmilière.

Chaque case comporte un niveau de phéromones (qui peuvent être de différents types) et qui décroît naturellement avec le temps. Les fourmis peuvent ajouter des phéromones sur la case où elles se trouvent.

C. FONCTIONNALITES AMBITIEUSES

Les fonctionnalités décrites dans la partie précédente représentent le minimum vital pour le fonctionnement du programme. Toutefois, nous souhaiterions ajouter d'autres fonctionnalités si le temps nous le permet.

Jusqu'à maintenant, la classe fourmi désignait implicitement des ouvrières. On voudrait désormais diversifier les sous-classes de fourmis en précisant la sous-classe « ouvrière » et ajoutant la sous-classe « soldates ». Si une ouvrière trouve une fourmi d'une autre fourmilière, elle retourne à la fourmilière en laissant une trainée de phéromones d'alerte. Une fois averties, les soldates remontent cette trainée et attaquent les fourmis ennemies qu'elles trouvent. Cela implique de faire en sorte que les fourmis puissent mourir.

Dans la continuité de notre volonté de rendre notre modèle plus réaliste, implémenter un système de faim, qui forcerait les fourmis à consommer régulièrement de la nourriture pour survivre nous semble intéressant. L'entraide est permise ! Une fourmi croisant une congénère affamée n'hésitera pas à lui transférer une partie de ses propres réserves de nourriture via son jabot social.

En raison des systèmes précédent, il serait utile d'ajouter une fonctionnalité de naissance en fonction de la réserve de nourriture disponible : Lorsque la réserve de nourriture dans la fourmilière atteint un certain niveau de remplissage, alors une nouvelle fourmi est créée. Ce sera, en fonction des besoins et de la population actuelle, une ouvrière ou une soldate. Une fourmilière qui ne réussit pas à récolter assez de nourriture au quotidien (pas assez d'ouvrière, agressions extérieures, environnement trop pauvre, manque d'efficacité etc...) n'arrivera pas à neutraliser son rapport naissance/décès et sera donc vouée à disparaître.

Pour limiter la croissance des fourmis, on pourrait ajouter le fait qu'elles puissent mourir de vieillesse ainsi qu'un système de climat qui change les paramètres de l'environnement.

III. Logiques de fonctionnement des principales classes

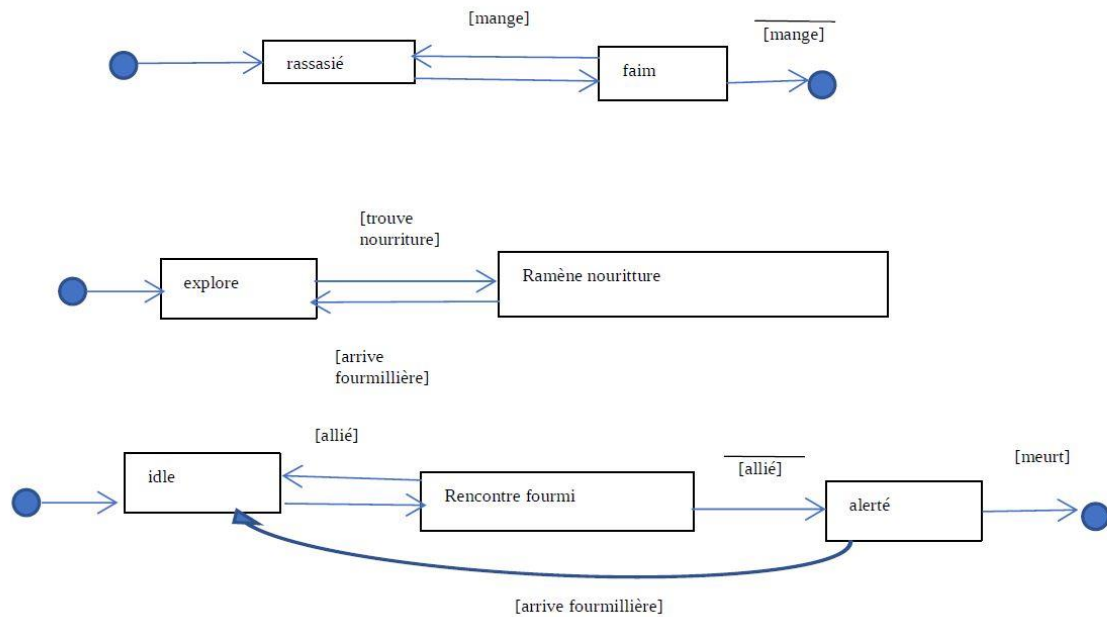
A. PRINCIPE DU SYSTEME DE PHEROMONE

Dans cette situation, on a 3 fourmis, 1,2,3, qui se déplacent sur le terrain. En se déplaçant, elles déposent des phéromones qui constituent un chemin vers la fourmilière. La fourmi 1 trouve de la nourriture elle la ramène à la fourmilière en suivant, sur chaque case, la direction indiquée par les phéromones de type « chemin vers la fourmilière », en prenant la direction la plus indiquée.

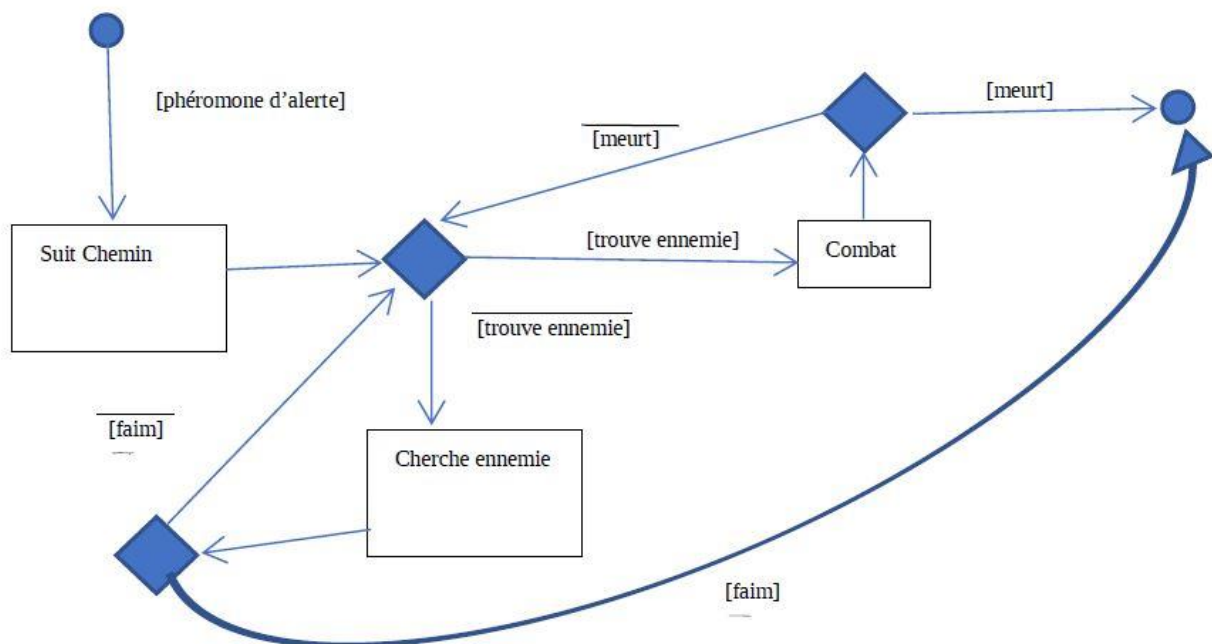
				* : fourmilière O : nourriture < : chemin vers la fourmilière 1 < : chemin vers la fourmilière 2 < : chemin vers la fourmilière 3 > : chemin vers la nourriture 1
			O v	
	v	> < v	^ v < <	
* >	> < < <	^ < <	< ^ ^	

Tableau 1 : exemple de fonctionnement des phéromones sur la carte

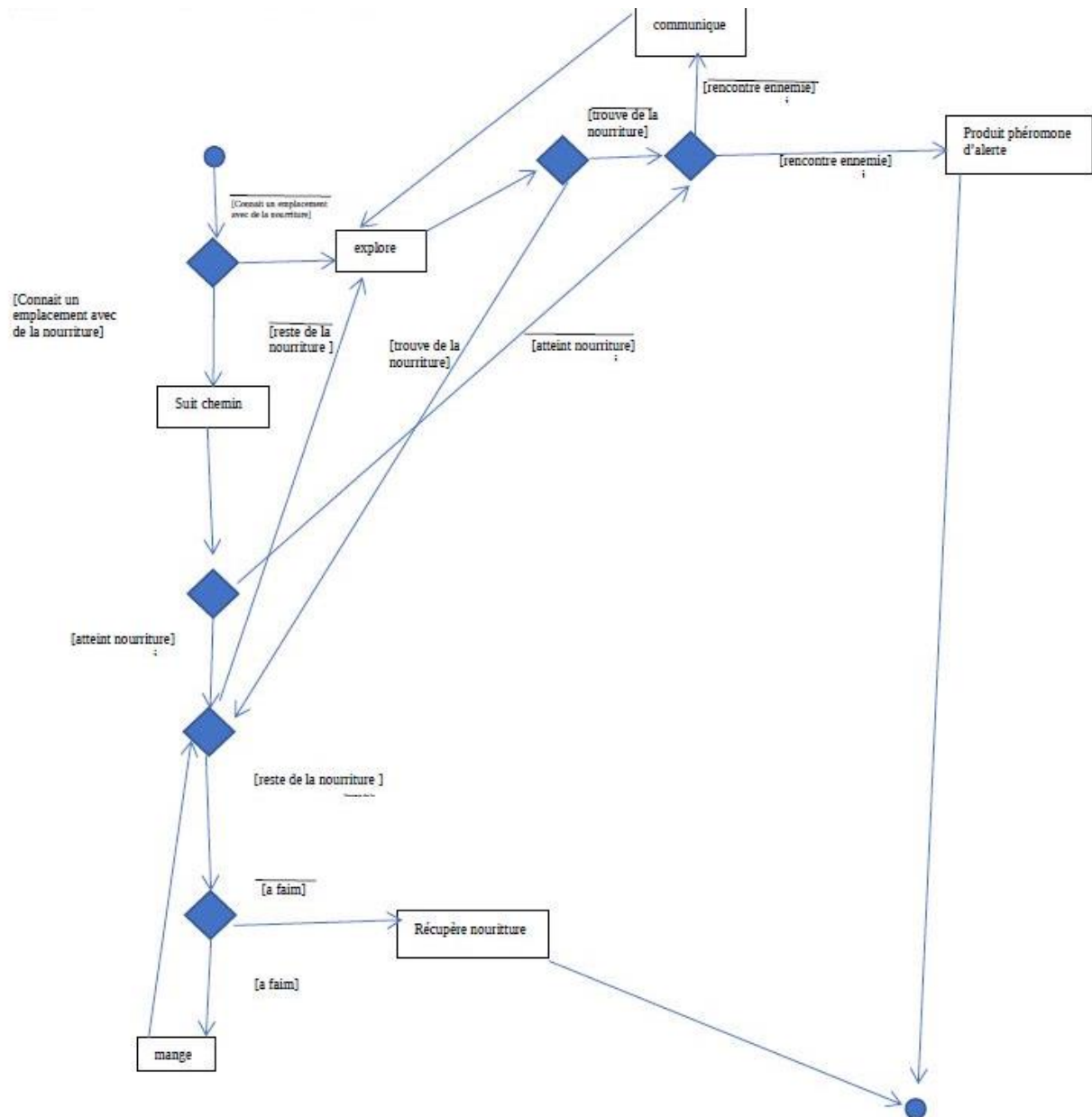
B. DIAGRAMME D'ACTIVITE CLASSE FOURMI



C. DIAGRAMME D'ACTIVITE SOUS-CLASSE SOLDAT



D. DIAGRAMME D'ACTIVITE SOUS-CLASSE OUVRIERE



IV. Planning

