Robotique

La migration hivernale des sauvageons

2019-2020

Amélie DELAIN & Léa HARLAY
Groupe 3 - Alternant

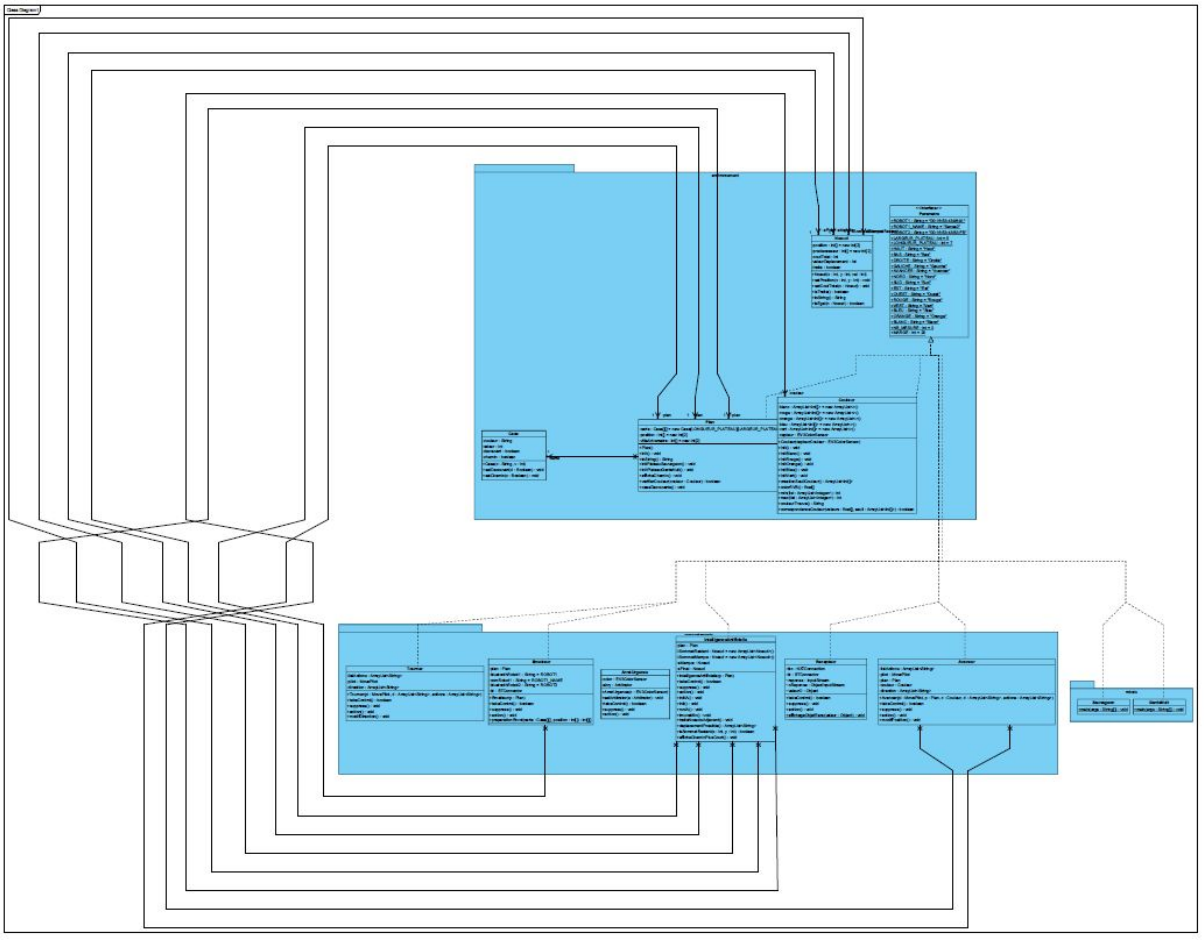
Javadoc

À la racine du projet, le dossier nommé “Javadoc” contient la Javadoc du programme.

Diagrammes UML

Diagramme de classes

Pour générer le diagramme de classes de notre code source, nous avons utilisé le logiciel Visual Paradigme. Il se trouve également à la racine du projet.

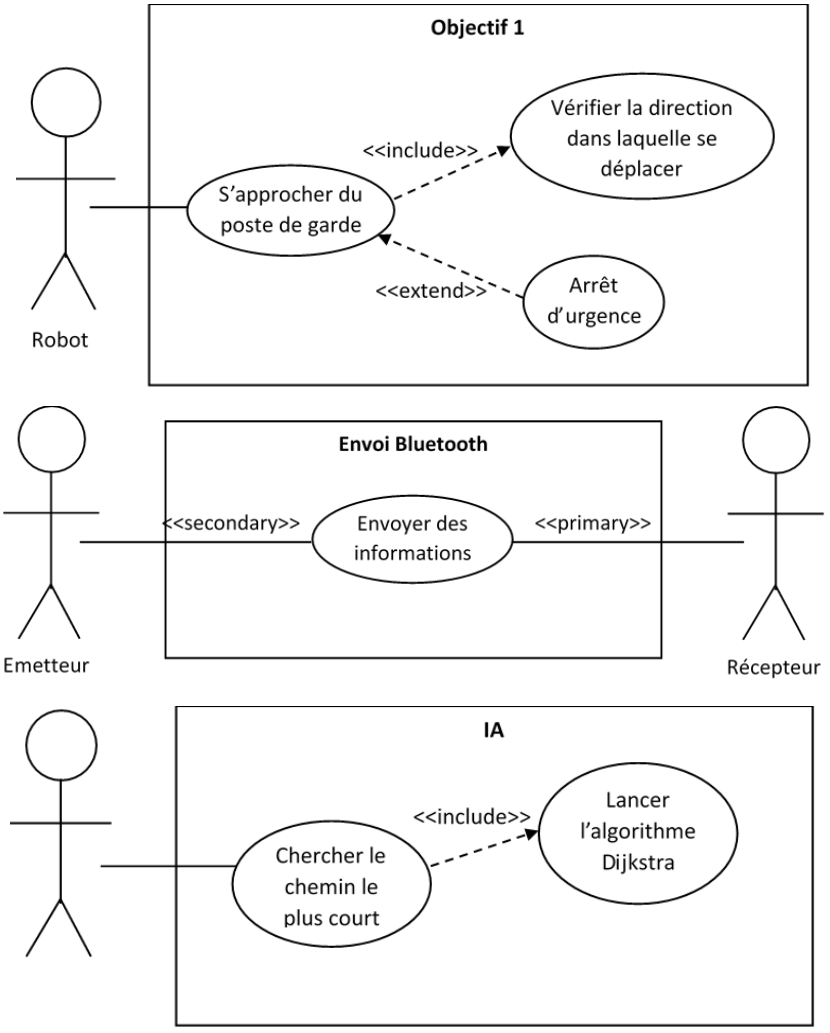


Nous avons choisi de décomposer le projet en trois packages :

- “comportement” : pour les comportements du robot permettant sa gestion,
- “environnement” : pour l’environnement (le plateau, les constantes, etc.) qui regroupe les modalités de l’activité et dans lequel les robots évoluent,
- “robot” : pour les classes principales permettant de faire fonctionner les robots.

L’utilisation de comportements nous permet de pouvoir interrompre ce qu’est en train de faire le robot. Pour cela, nous avons utilisé l’abritrator vu en cours.

Diagramme de cas d'utilisation



Guide de lancement du programme

1. Initialisation des robots

Après avoir allumé les robots, il est nécessaire de renseigner les informations Bluetooth de chacun robot dans l'interface "Parametre" situé dans le package "environnement".

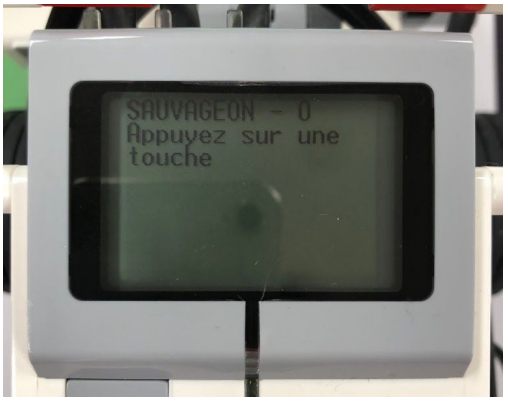
Exemple :

```
public interface Parametre {
    // Bluetooth
    final String ROBOT1 = "00:16:53:43:96:91";
    final String ROBOT1_NAME = "Sansa2";
    final String ROBOT2 = "00:16:53:43:B3:FB";
```

Ensuite, vous pouvez intégrer le code dans les robots. Ceux-ci doivent avoir des rôles différents. Cette spécification est gérée par deux classes Main, dans le package “robots”, portant chacun le nom du rôle qu’elle déclenche :

- Sauvageon.java pour le rôle du sauvageon
- GardeNuit.java pour le rôle du garde de nuit

Une fois que le robot vous le demande, vous pouvez appuyer sur n’importe quelle touche afin qu’il démarre.



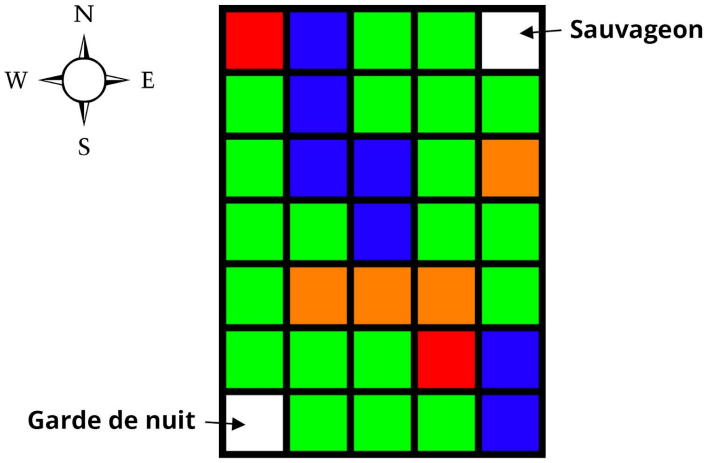
2. Initialisation des couleurs

Au départ, le robot va vous demander d’initialiser chacune des couleurs. Cette initialisation permet de mieux détecter ces dernières. En effet, cette calibration permet de se baser sur les couleurs et l’intensité lumineuse au moment du lancement du robot. En mesurant trois fois les valeurs RVB de chacune des couleurs, le robot peut établir des seuils entre lesquels il reconnaîtra le rouge, le orange, le bleu, etc.

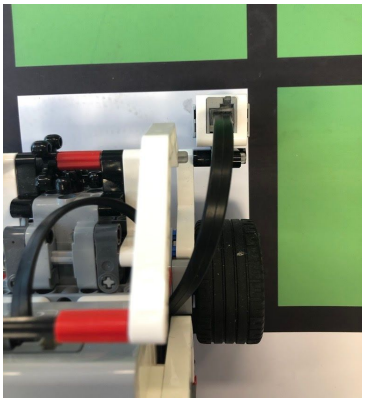
Pour effectuer cette initialisation, le robot va vous indiquer une couleur. Il faudra alors placer son capteur sur cette couleur et appuyer sur une touche. Il vous indiquera ensuite une autre couleur et ainsi de suite.

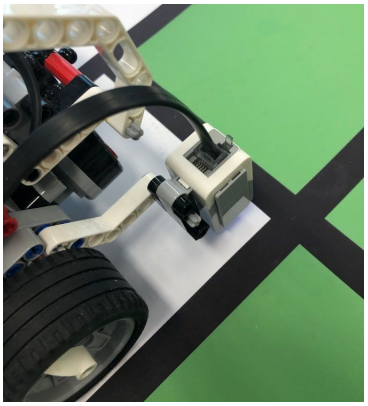
3. Positionnement des robots

Une fois l'initialisation des couleurs terminée et avant de lancer le programme, il faut positionner le robot sur la carte. Le Sauvageon se place sur la case en haut à droite, dans la direction Ouest. Le Garde de Nuit, quant-à lui, doit être placé sur la case en bas à gauche de la carte, dans la direction Nord.



Les roues et le capteur doivent être positionnés comme sur les photos suivantes.

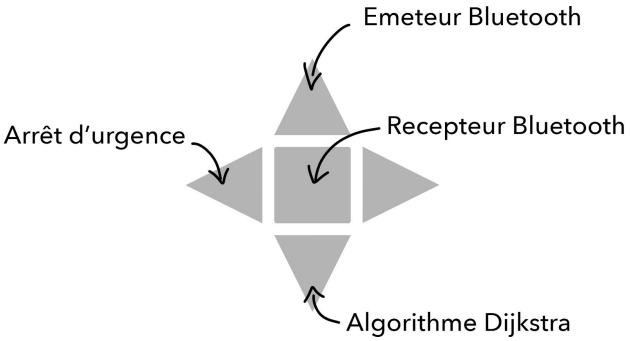




Suite à cela, le robot peut être lancé. Il est possible d'appuyer sur n'importe quelle touche afin de faire démarrer l'Arbitrator qui permettra de coordonner les différents comportements du robot.

4. Lancement des objectifs

Le lancement de l'Arbitrator, effectué lors de l'étape précédente, enclenche automatiquement la réalisation de l'objectif 1. Des boutons spécifiques servent, quant à eux, à lancer les autres objectifs et l'arrêt d'urgence.



Retour sur les objectifs

La vidéo des robots effectuant les différents objectifs est disponible au lien suivant : https://youtu.be/VaKjMOLtRIU. Nous estimons avoir implémenté les trois objectifs donnés aux alternants à 100 %.

Objectif 1

L'objectif 1 consiste à faire avancer le robot jusqu'au camp militaire le plus proche (Nord-Ouest pour le Sauvageon et Sud pour la Garde de Nuit).

Afin de le faire avancer au bon endroit, nous avons fourni à chacun d'eux son itinéraire (enchaînement des actions qu'il doit faire). De plus, à chaque fois que le robot avance d'une case, il vérifie que la couleur qu'il détecte est celle qu'il devrait voir (d'après la carte qu'il connaît).

Nous avons également implémenté un comportement d'arrêt d'urgence pour pouvoir arrêter le robot à n'importe quel moment en appuyant sur le bouton LEFT.

Objectif 2

L'objectif 2 concerne l'envoi de données en Bluetooth. Afin de réaliser cet objectif, nous avons fait le choix d'utiliser un bouton pour le récepteur et un bouton pour l'émetteur. Cette solution nous a permis de ne pas laisser un robot "tourner" dans le vide en attendant que l'autre robot se connecte à lui.

Il faut donc appuyer sur le bouton CENTER du robot à qui nous voulons envoyer les informations (le récepteur). Lorsqu'il affiche "Socket is ...", vous pouvez relâcher le bouton et aller voir l'autre robot.

Sur le robot émetteur, il faut appuyer sur le bouton UP jusqu'à voir apparaître le nom et "Socket is ...". A ce moment, le robot émetteur est en train d'envoyer la carte qu'il connaît et sa position au robot récepteur. Sur le robot récepteur, les 'X' correspondent aux cases qui ne sont pas connues et le 'o' est la position du robot.

Objectif 3

Étant un groupe d'alternantes, nous n'étions pas concernées par cet objectif. Nous ne l'avons donc pas réalisé.

Objectif 4

L'objectif 4 consistait en l'implémentation d'un algorithme d'intelligence artificielle pour que le robot trouve le chemin le moins coûteux pour aller de sa position à la ville adverse.

Pour cela, nous avons décidé d'utiliser l'algorithme Dijkstra. Cet algorithme a pour objectif de trouver le chemin le plus court entre un sommet de départ et un sommet d'arrivée. Cet algorithme nous semble parfaitement adapté à notre problématique. Dans le cadre de celle-ci, la pondération des différents sommets s'effectue par le coût d'exploration associé à chaque case. De ce fait, chaque nœud ou sommet représente ainsi une case du plateau avec un coût d'exploration.

Suivant l'algorithme Dijkstra, notre programme calcule le chemin le moins coûteux en terme de coût d'exploration entre la position du robot et la ville adverse. Une fois ce chemin trouvé, le robot affiche à l'écran le résultat sous la forme d'une carte. Cette carte, orientée du Nord au Sud, affiche la position actuelle du robot par le signe 'O', la position de la ville adverse par le signe 'V' et le chemin à emprunter par les signes 'X'. Les signes '_' représentent les autres cases de la carte.

Objectif 5

Étant un groupe d'alternantes, nous n'étions pas concernées par cet objectif. Nous ne l'avons donc pas réalisé.