Projet 1 : Un DSML (Domain Specific Modeling Language) pour la robotique mobile

1. Introduction

Le but de ce projet est de développer un DSL et un générateur de code pour faciliter le développement de petites applications de robotique mobile. Le DSL proposé doit se baser sur les diagrammes dites d'Arkin présentés ci-dessous et qui sont le fruit d'une démarche basée sur les comportements (behavior-based).

Le but est de permettre à partir des programmes écrits avec votre DSL de produire le code vers le simulateur open source Webots (https://www.cyberbotics.com/) .

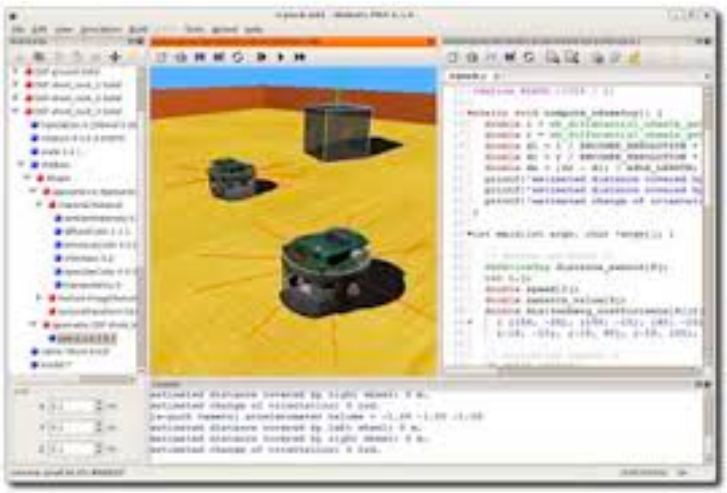

Fig. Le simulateur Webots

2. Le domaine

La démarche basée sur les comportements organise la description comportement global d'un robot en un réseau de comportements qui produisent collectivement le comportement global du robot. Chaque comportement est un module de contrôle qui vise à atteindre et à maintenir un certain but. En entrée (voir la Figure 1) un comportement reçoit des données d'un capteur ou d'un autre comportement et produit des données vers les effecteurs du robot ou vers d'autres comportements. Toutefois, les sorties des comportements sont au préalable interceptés par un module arbitre qui décide typiquement de la priorité de chaque comportement. Par convention les comportements sont rangés de haut en bas par priorité décroissante. Ci-dessous les concepts du diagramme d'Arkin sont présentés en détail.

2.1 Les capteurs

Chaque robot possède un ensemble de capteurs (sensors) qui lui permettent d'avoir une vue sur son environnement a partir de sa position courante. Il existe plusieurs types de capteurs selon le type du robot :

- Les capteurs de distance : ces capteurs retournent la distance par rapport à un obstacle. Plus le robot se rapproche d'un obstacle, plus la valeur retournée par ses « distance sensors » est proche de 0.

- Les capteurs de lumière : ces capteurs retournent une valeur très élevée lorsque le robot est proche d'une source de lumière.

- Les capteurs Infrarouge, etc.

2.2 Les comportements

Ils prennent en entrée les valeurs retournées par les capteurs (ou par d'autres comportements) et agissent sur les effecteurs. Chaque comportement vise a atteindre et maintenir un certain but et est typiquement décrit par une machine a états.

2.3 L'arbitre

Il intercepte les actions envoyées par les comportements aux effecteurs et les transmet (ou non) selon sa politique d'arbitrage qui typiquement consiste à faire respecter une certaine priorité définie statiquement (avant l'exécution du diagramme).

2.4 Les effecteurs

Les effecteurs (actuators) permettent au robot de se déplacer et plus généralement d'agir dans son environnement.

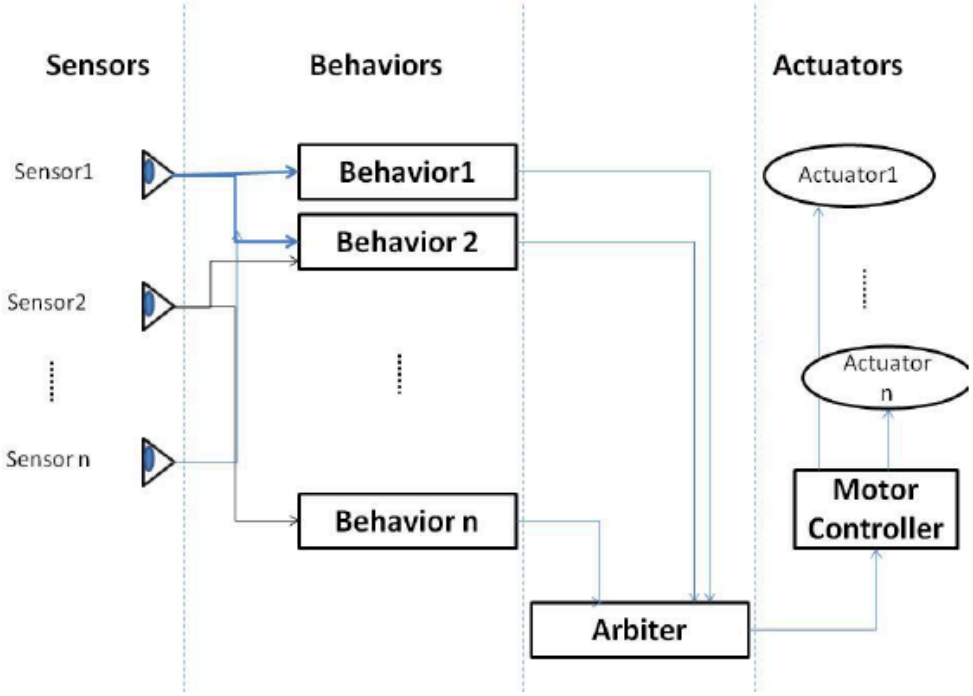

Figure 1 diagramme Arkin pour les robots mobiles.

3. Etude de cas

Pour illustrer le méta-modèle et le générateur de code que vous avez développé, nous vous proposons une étude de cas simple. Il s'agit de la mission *SearchForLight* qui consiste à se diriger vers une source de lumière. Cet exemple a été testé sur un robot e-puck sous le logiciel webots et peut être adapté au robot wifibot. l'environnement du robot est composé d'obstacles (objets), d'une source de lumière et est délimité par un mur. Le robot doit se diriger vers la lumière tout en évitant les obstacles. Pour cela, nous avons défini les 2 comportements suivants comme le montre la Figure 3 :

- *Escape* qui permet au robot d'éviter les obstacles sur son passage en utilisant des capteurs de distance.
- *Home* qui permet au robot de se diriger vers la lumière en utilisant des capteurs de lumière.

Avant de modéliser cet exemple en utilisant votre méta-modèle, il est demandé de décrire chaque comportement par une machine à états que nous vous demandons de la réaliser.

4. Travail à réaliser

Le projet consiste principalement à :

1. Proposer la syntaxe abstraite de votre DSL sous forme d'un méta-modèle pour représenter des missions pour un robot mobile en utilisant les diagrammes d'Arkin.
2. Proposez une syntaxe concrete de votre méta modèle en utilisant XText.

3. Implémenter un générateur de code pour pouvoir exécuter la mission au sien du simulateur Webots. Webots permet d'éxecuter des missions ecrites dans plusieurs langages : des langages généralistes (c, java) ou des DSL comme le langage urbiscrip.

Planning proposé

Etape 1 : analyse du domaine

- Comprendre les diagrammes d'Arkin et leurs concepts
- Jouer avec le simulateur webots et comprendre comment on peut l'utiliser pour executer une application robotique.

Etape 2 : proposez le méta-modèle (syntaxe abstraite) & une syntaxe concrete avec XText

Etape 3 : implementer le générateur de code. Vous avez le choix pour le langage cible parmis les langages supportés par Webots

Etape 4 : test et rédaction du rapport

Le projet suivra une démarche itérative et autant que possible dirigée par les tests. Il ne s'agit donc pas de faire complètement telle étape avant de passer à la suivante mais au contraire de développer rapidement une version du méta-modèle puis du générateur du code qui permettent de voir un comportement très simple tourner sur le robot rapidement.

Deadlines

Projet à réaliser dans 4 semaines (datelimite pour rendre le projet 12/11/2020)

Références

[1] R.C. Arkin. Behavior-based robotics. The MIT Press, 199

[2] O. Michel. Webots: Symbiosis between virtual and real mobile robots. In Virtual Worlds, pages 254{263. Springer, 1998.

[3] Webot : http://www.cyberbotics.com/

[4] URBI for Webot : http://www.gostai.com/doc/en/webots/