ÉCOLE POLYTECH'NICE SOPHIA
SI5
2018-2019

Julien Deantoni

Behavioral Language Engineering

Meta programming an interpreter/debugger/model checker

Project delivery

To be defined.

Objectives: Defining a tooled executable language

Your objective here is to define, by using the meta languages provided in the GEMOC studio, the syntax and the behavioral semantics of a language dedicated to, either the programming of an exploration robot or the manipulation of information stream provided by sensors. Your language will define a graphical concrete syntax that will be used to represent useful debugging information about the model state. You can also equip your language with a textual concrete syntax. In order to elaborate your language, it might be useful to think about existing behavioral languages like the one in UML and to browse the languages available in `http://gemoc.org/gallery.html` or `https://www.eclipse.org/sirius/gallery.html`.
Eventually, it is expected to provide a development environment with an omniscient debugger, model animation and a model checker, if possible. Note that, for this last purpose, in the last steps of your development, you may have to slightly modify the meta languages used in the GEMOC Studio (or their tooling)..

Development steps

to develop your language, do not try to have a pure waterfall approach. Adopt an iterative development in which, for each step, you will

- define (a part of) the syntax (abstract and concrete)
- define the corresponding behavioral semantics
- provide appropriate debugging representation
- make a test of your language in several execution mode (execution, debugging, omniscient debugging)

But before to play[1], please choose one of the two following subjects.

[1] or work, depending on you

subject 1: V-REP Robot DSL

In this subject, your objective is to define full development environment based on an appropriate language to control of a specific exploration robot. The robot to control is defined in the V-REP environment (`http://www.coppeliarobotics.com/`). This environment allows several robots robots and scenes but the expected language is restricted to the control of the *PolyBot* robot defined here: `https://github.com/jdeantoni/PolyBot/tree/master/scenes`.

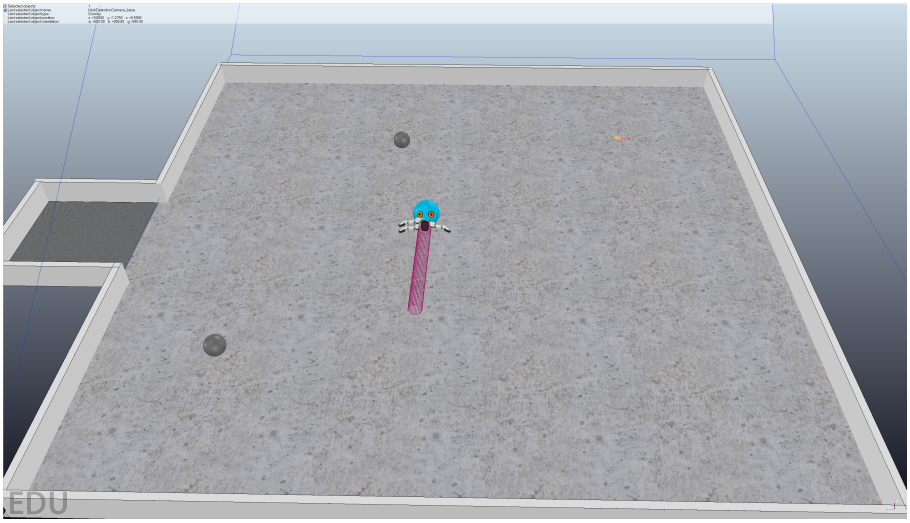

Figure 1: a view of the proposed VREP environnement

In `https://github.com/jdeantoni/PolyBot` you will find a a Java abstraction layer on top of the vrep remote API. while not sufficient for all the required purpose, it can be used as a starting point. Your language should allow the realization of different missions where the robot is programmed towards specific objectives. The simulation and debugging of the program should control polyBot in the simulated environment. This means that the time in your execution/debug session and in the simulator should be synchronized.
You are the decider concerning the expressiveness of your language; however it is obviously required to deal with temporal and concurrent actions. During the development of your language, you should be aware of what is the *State* of your system (and document it) and if it allows for exhaustive exploration as well as V&V activities.

Subject 2: Smart Home Data Fusion DSL

In this subject, your objective is full development environment based on an appropriate language to manipulate time series from sensors in order to construct higher abstraction level understanding then the data themselves. For instance, If someone sits in her office for 2 hours then stand up, open the kitchen door, open the fridge, leave the kitchen 5 minutes later and then sit again in her office, then one can deduce a five minute break for refreshing or eating something. More elaborated, if someone goes to the toilets "too often", one can recommend this person to have an appointment with his doctor. Etc. Your goal is to provide a language to (1) define the different sensors that are used by the system, together with their position in a flat; and to (2) elaborate the construction of time series reasoning as shown earlier. Below is an example of such language.

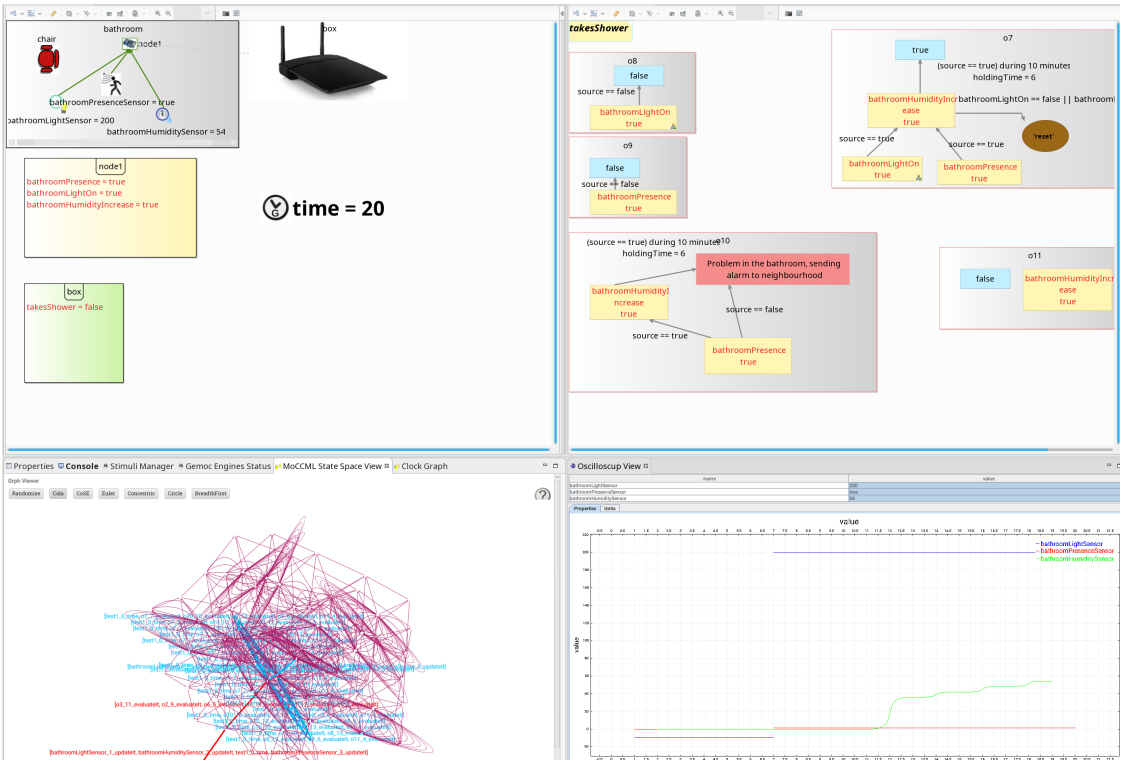

Figure 2: example of a possible language for smart home data fusion, under simulation

In `https://archive.ics.uci.edu/ml/datasets/Activities+of+Daily+Living+%28ADLs%29+Recognition+Using+Binary+Sensors#` or `https://archive.ics.uci.edu/ml/datasets/Localization+Data+for+Person+Activity` or `http://traces.cs.umass.edu/index.php/Smart/Smart` or more generally here `https://archive.ics.uci.edu/ml/datasets.html?format=&task=&att=&area=&numAtt=&numIns=&type=ts&sort=nameUp&view=table` you will find an example of a time series obtained based on (home) monitoring. Depending on the evolution you want to give to your language you can decide to mock some sensors (e.g., to create test scenario), to link with existing sensors (for example through MQTT) or to read open data bases. In any cases, the time in your execution/debug session and from the sensor should be synchronized.

You are the decider concerning the expressiveness of your language; however it is obviously required to deal with temporal and concurrent actions. During the development of your language, you should be aware of what is the *State* of your system (and document it) and if it allows for exhaustive exploration as well as V&V activities.