Université de Lille 1 SCIENCES ET TECHNOLOGIES

UFR IEEA
Informatique, Electronique
Electrotechnique, Automatique

Multi-agent systems
Behaviour simulation
Individual-centered approach

Authors :
Salla DIAGNE
Anis TELLO

10th FEBRUARY 2016

Table of contents

1 Technical work **4**
1.1 Architecture 4
1.1.1 Core 4
1.1.2 Particles 5
1.1.3 Wator 6
1.1.4 Hunt 7
1.2 Behavior 8
1.2.1 Particles 8
1.2.2 Wator 8
1.2.3 Hunt 9

2 Evaluation **10**
2.1 Performance 10
2.1.1 Particles 10
2.1.2 Wator 10
2.1.3 Hunt 10
2.2 Ease of use 10

Références **12**

Chapter 1

Technical work

1.1 Architecture

1.1.1 Core

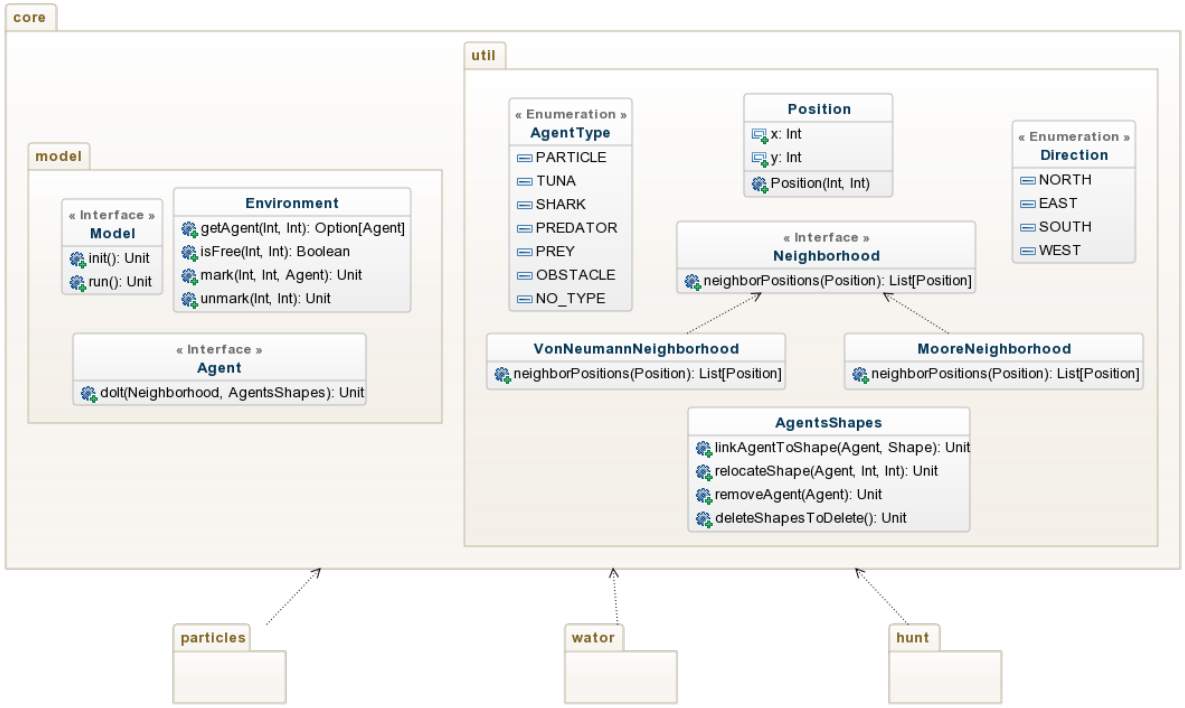

Figure 1.1: Overview

The overview

1.1.2 Particles

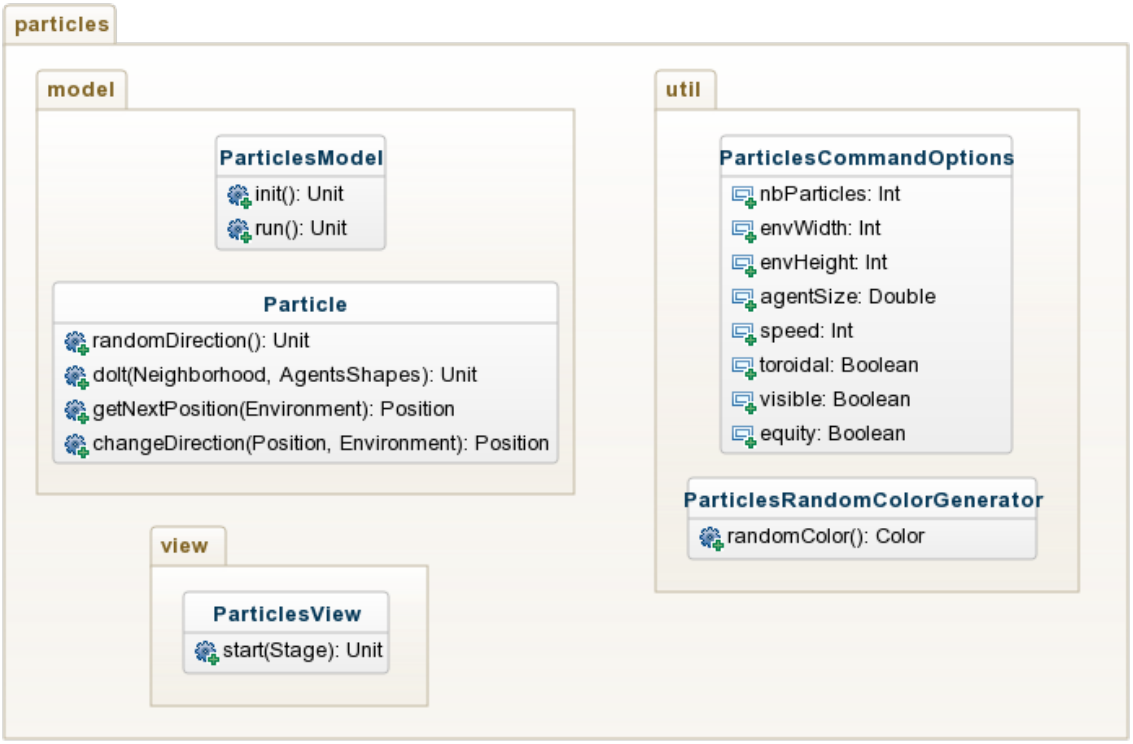

Figure 1.2: Particles diagram

1.1.3 Wator

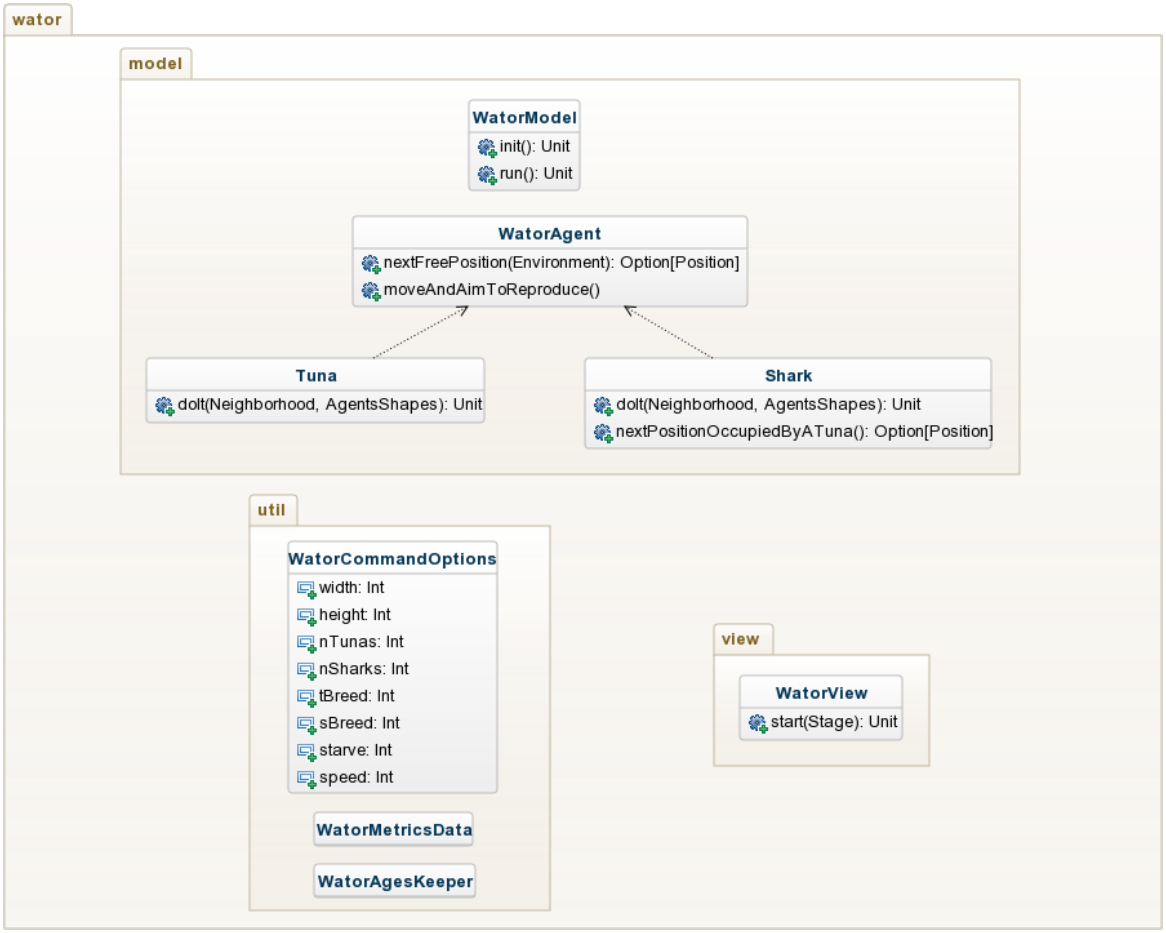

Figure 1.3: Wator diagram

1.1.4 Hunt

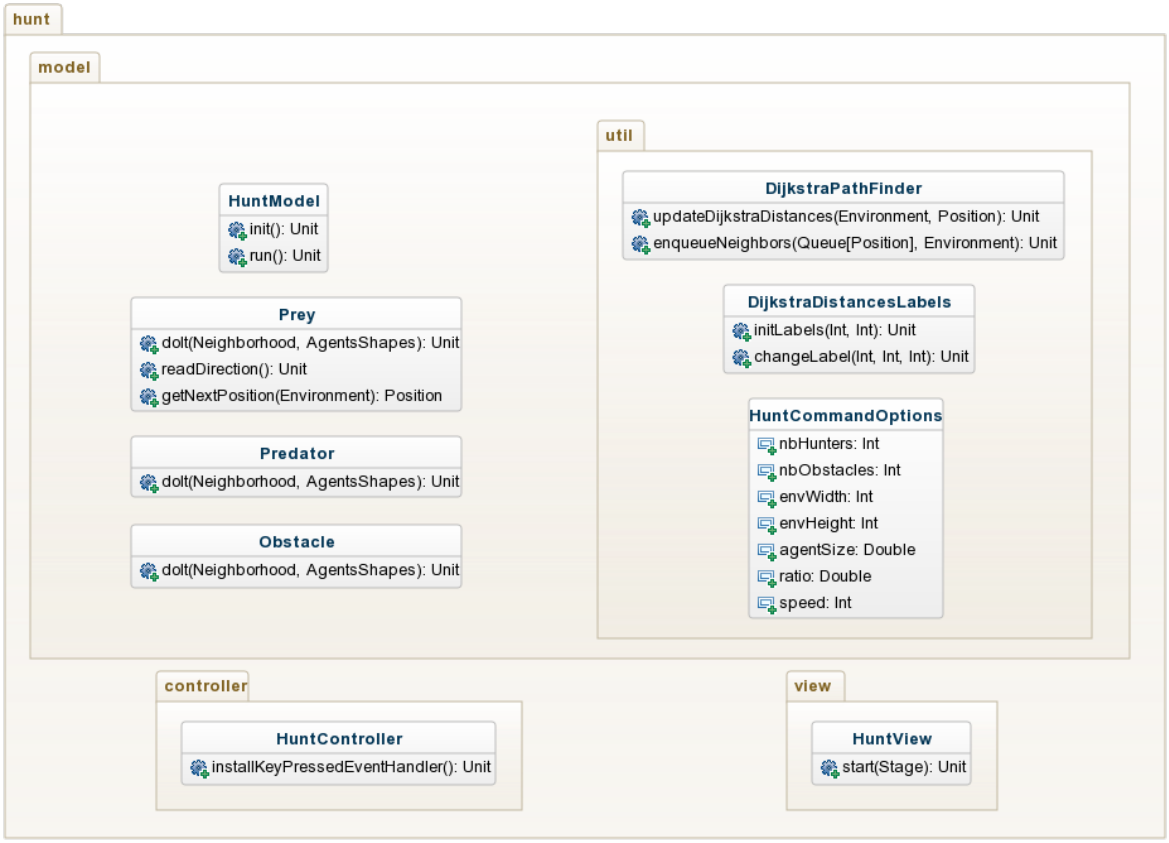

Figure 1.4: Hunt diagram

1.2 Behavior

1.2.1 Particles

Our particles behave as following: Each particle has directions on X-axis and Y-axis (-1, 0, +1). At each round each particle move one place in Moors-neighbors[1] according to its directions. They will check the next possible position is empty, if it is not they choose randomly the first empty position in Moors-neighbors.

The method responsible of getting the next position checks whether the environment is toroidal or not.
If it is not the next possible position if a particle reaches the wall would be the next position after multiplying the direction by -1.
Otherwise if the environment is toroidal then particles can get throw walls and next possible position would be the first position on the other side of the environment.

At the beginning, All particles are distributed randomly in the environment with random directions.

1.2.2 Wator

WatorAgent is an agent that has an age and breed counter. Each WatorAgent can move randomly one step at a time in a toroidal environment. Also WatorAgent can breed if its breed counter has reached the fertility time (the number of round an agent must exist before reproducing)

Tuna

Tunas agents do not die if it has not been eaten. At each round tunas try to move randomly within Moors-neighbors - if there is an empty position. If it is the case, a tuna will try to reproduce if it can, by moving to the next empty position and leaving a new tuna in its place. If there are no empty place around, tuna will stay in its place and won't breed.

Shark

Shark has starvation counter, shark can die if it reach its starvation point. the first thing a shark will do is to check if it has reached the starvation point and die.
If not, a shark looks around within Moors-neighbors if there is a tuna, if yes it will move and eat it. At the same time if a shark can reproduce, it will reproduce by moving to the next position occupied by a tuna and leaving a new shark in its place.

If there are no near tuna, a shark will try to move randomly within Moors-neighbors and reproduce if it can.
If there are no empty place around then the shark will stay in its place and won't breed.

1.2.3 Hunt

Prey

Prey is an agent that has directions on X-axis and Y-axis (-1, 0, +1) in a toroidal environment. The prey keeps on moving one or more step at a time (it depends on the parameter *speed ratio between prey and predators*) in the same direction. User can change the direction of a prey by pressing up, down, left or right button on the keyboard.
If a prey reaches an obstacle it stops (direction on both X and Y axis becomes 0).

Predator

Predators are agents that move within Moors-neighbors in a torodial environment. Predators use Dijkstra's algorithm to choose a next position in order to catch the prey.

Obstacle

Obstacles are agents that do nothing besides occupying places inside the environment.

Chapter 2

Evaluation

2.1 Performance

2.1.1 Particles

We can go up to more than 37000 particles and the application will run smothly.
More than that the application will start quickly and run normally, but refreshing time will start to be visible to the naked eye.

2.1.2 Wator

For the default environment size (180 x 180) our application can go up to 16000 agents (both sharks and tunas). Default environment size is recommended so both *time-dependent number of fish* chart and *population chart of fiches* will be rendered correctly.

2.1.3 Hunt

2.2 Ease of use

Using the command line we can execute the program using one of these options:

```
Usage

 sma [options] command [command options]

Commands

   hunt [command options] : a prey, guided by the user's directions, is chased by a defined number of predators, guided by Dijkstra's algorithm
      --agentSize       : the size of the agent (i.e the radius in pixels of the circles representing the particles) [default = 5]
      --envHeight=NUM   : the height of the grid representing the environment [default = 300]
      --envWidth=NUM    : the width of the grid representing the environment [default = 300]
      --nbHunters=NUM   : the number of hunters [default = 1]
      --nbObstacles=NUM : the number of obstacles [default = 2]
      --speed=NUM       : the speed of the game (i.e. the number of milliseconds per lap) [default = 100]

   particles [command options] : simulates a bubble chamber using a multi-agent approach
      --agentSize       : the size of the agent (i.e the radius in pixels of the circles representing the particles) [default = 2.5]
      --envHeight=NUM   : the height of the grid representing the environment [default = the height of the screen]
      --envWidth=NUM    : the width of the grid representing the environment [default = the width of the screen]
      --equity          : if activated, there will be no ordering in the speaking of the agents (i.e. random) [default = false]
      --nbParticles=NUM : the number of particles in the room [default = 10000]
      --speed=NUM       : the speed of the game (i.e. the number of milliseconds per lap [default = 10]
      --toroidal        : if activated, the grid will be toroidal [default = false]
      --visible         : if activated, the lines of the grid will be visible [default = false]

   wator [command options] : wator is a simulation of the interaction over time of tunas and sharks in a small rectangular area
      --height=NUM  : the height of the grid representing the environment [default = 180]
      --nSharks=NUM : the number of sharks in the environment, distributed randomly [default = 500]
      --nTunas=NUM  : the number of tunas in the environment, distributed randomly [default = 900]
      --sBreed=NUM  : the number of cycles a shark must exist before reproducing [default = 9]
      --speed=NUM   : the speed of the game (i.e. the number of milliseconds per lap) [default = 80]
      --starve=NUM  : the number of cycles a shark has to find food before starving [default = 5]
      --tBreed=NUM  : the number of cycles a tuna must exist before reproducing [default = 3]
      --width=NUM   : the width of the grid representing the environment [default = 180]

No command found, expected one of hunt, particles, wator
```

Figure 2.1: Usage

Example of executing the command line with default parameters:

```
$ java -jar target/scala-2.11/scalagent.jar hunt
```

Example of executing the command line with specific parameters:

```
$ java -jar target/scala-2.11/scalagent.jar particles --nbParticles=15000
    --toroidal=true
```

Bibliography

[1] Wikipedia. Moore neighborhood. https://en.wikipedia.org/wiki/Moore_neighborhood.