Ejercicios Clase: Comportamientos y Hebras

Sistemas Multiagentes

15 de octubre de 2018

En este ejercicio el alumno deberá comparar las salidas de los dos programas descritos a continuación y debe identificar si ocurre alguna condición de carrera:

1. Versión sin utilización de Hebras.

Se creará un agente llamado *AgenteHebra* con una variable entera compartida y en el que su comportamiento principal será un comportamiento paralelo. Este comportamiento tendrá asociado tres subcomportamientos:

- Un comportamiento wakeup que borrará el agente tras 12 segundos.
- Dos comportamiento cíclicos Comp1 y Comp2 que realizan la misma acción:
  1. Incrementan el valor de la variable entera, esperan n milisegundos e imprimirán el valor de la variable. Para realizar las pausas se utilizará el método Thread.sleep(time ms).
  2. Los dos valores de n se pasarán al constructor y serán de 2000ms para Comp1 y 3000ms para Comp2.

2. Versión con utilización de Hebras.

Se variará la versión anterior del programa para que Comp2 se ejecute en una hebra dedicada. Para asociar un comportamiento a una hebra:

- Se creará una variable de tipo **ThreadedBehaviourFactory** llamada **CompHebras**.
- Cuando se añada el comportamiento Comp2 al agente se llama al comando: **addBehaviour(CompHebras.wrap(Comp2))**.

Para finalizar un comportamiento que se ejecuta en una hebra dedicada:

- El método **removeBehaviour**() no funciona para comportamientos de este tipo.
- Por lo anterior, este tipo de comportamientos deben ser "killed" de forma explícita.
- Se recuperará el thread llamando al método **getThread**() de ThreadedBehaviourFactory.
- Posteriormente se llamará al método **interrupt**().

```
1
2  import jade.core.Agent;
3  import jade.core.behaviours.Behaviour;
4  import jade.core.behaviours.CyclicBehaviour;
5  import jade.core.behaviours.ParallelBehaviour;
6  import jade.core.behaviours.ThreadedBehaviourFactory;
7  import jade.core.behaviours.WakerBehaviour;
8
9
10 public class XXXXXXXXXX extends Agent {
11
12
13         @Override
14         protected void setup() {
15                 XXX = new ThreadedBehaviourFactory();
16                 pb.addSubBehaviour(XXXX.wrap(XXXX));
17
18
19
20         }
21
22         @Override
23         protected void takeDown() {
24                 System.out.println("Taking down");
25                 Thread td = XXXX.getThread(XXXXX);
26                 td.interrupt();
27                 super.takeDown();
28         }
```