Hands-on Practice 2016

Developing Multi-agent Systems with JADE

Based on tutorial by: Arash Mousavi , LTU, 2013

If you run this assignment on a Windows machine you need to work with Windows Command Line (cmd). Closing a Program Running on an open cmd: Ctrl + C. Open a cmd from an open cmd: > start cmd.exe Open a cmd from an open cmd and pass an argument to it: /> start cmd.exe /C argument

/> start cmd.exe /C java jade.Boot –container –container-name MyContainer

On a Linux machine open a terminal window, navigate to your working folder with sample code and execute java jade.Boot –container –container-name MyContainer This assumes that you correctly set up all path and classpath environment variables.

1. **Starting up Jade**
   java jade.Boot -gui
2. **Creating Containers (default, User Defined Name)**
   java jade.Boot –container
   java jade.Boot –container –container-name My_Container

3. **Creating Container in Different Machine**
   java jade.Boot –container –container-name My_Container –host [IP/Computer Name] –port 1099

4. **Creating first Jade Agent (HelloWorldAgent.java), compiling Jade Agent**
   Copy or create a folder (programs) in your drive C for ease of use
   Creat/save your java files in "programs" folder

   - The setup() method is intended to include agent initializations. The actual job an agent has to perform is typically carried out within 'behaviours'
   - Examples of typical operations that an agent performs in its setup() method are:
     Showing a GUI.
     Opening a connection to a database
     Registering the services it provides in the yellow pages catalogue
     Starting the initial behaviours.
   - It is good practice not to define any constructor in an agent class and to perform all initializations inside the setup() method.
   - This is because at construction time the agent is not yet linked to the underlying JADE run-time and thus some of the methods inherited from the Agent class may not work properly.

5. **Running HelloWorldAgent**

   a) Running together with GUI:

   java jade.Boot –gui Bob:HelloWorldAgent

   b) Running more than one Instances of the HelloWorldAgent

   java jade.Boot –gui Bob:HelloWorldAgent Alice:HelloWorldAgent

   java jade.Boot –gui Bob:HelloWorldAgent Alice:HelloWorldAgent John:HelloWorldAgent

   Now activate the `//doDelete();` command in the source file and run Bob and Alice. Observe the execution of the program.

   c) Running HelloWorldAgent in specific container:

   Open a Command Line and run GUI: java jade.Boot –gui

   Open a New CommandLine and Type: java jade.Boot –container Bob:HelloWorldAgent Alice:HelloWorldAgent

   d) Run Bob and Alice in user-defined container name

   java jade.Boot –container –container-name MyContainer Bob:HelloWorldAgent Alice:HelloWorldAgent

   e) Run Bob and Alice in different machines:

   java jade.Boot –container –container-name LocalContainer Bob:HelloWorldAgent

   java jade.Boot –container –container-name RemoteContainer –host "IP" –port 1099 Bob:HelloWorldAgent

6. **Get and diplay full Agent Identification Info.**

   Open file: HelloWorldAgentFull and observe the classes used

   java jade.Boot –container MyContainer Bob:HelloWorldAgentFull

7. **Agent with Argument**

   Examine AgentArg.java File

   Compile and Run it with the following command:

   java jade.Boot –container Bob:AgentArg(arg1 arg2 arg3)

   java jade.Boot –container Bob:AgentArg(Hi arg2 arg3)

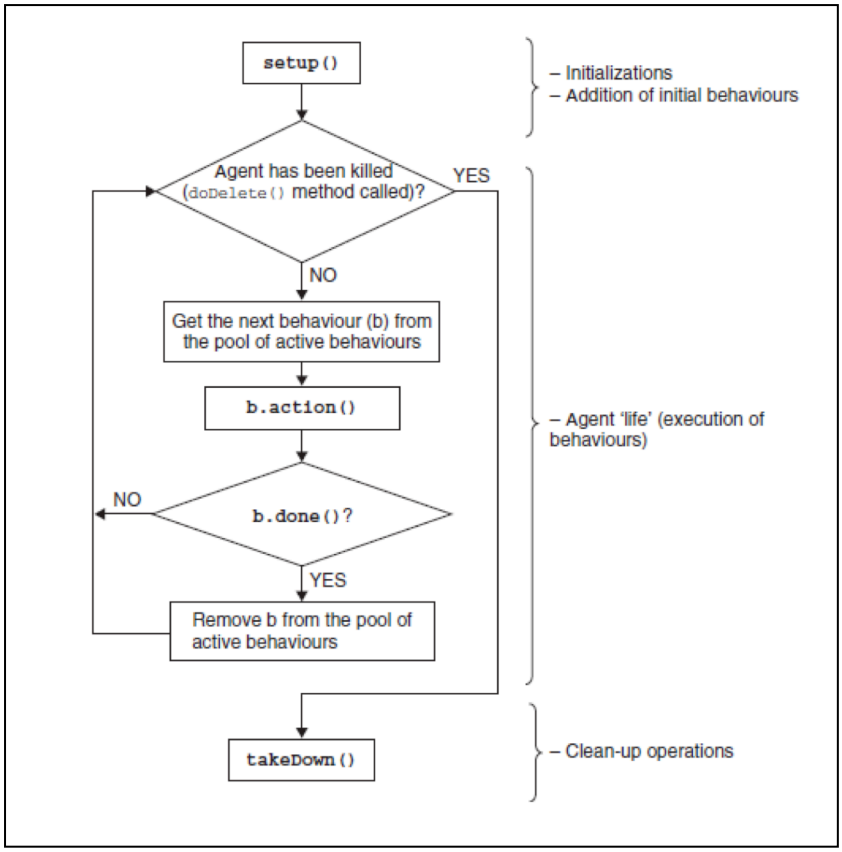
   java jade.Boot –container Bob:AgentArg(Bye arg2 arg3)

8. Agent Behavior

A behaviour is basically an *Event Handler*, a method which describes how an agent reacts to an *event*. Formally, an *event* is a relevant change of state; in practical terms, this means: reception of a message or a Timer interrupt. In Jade, Behaviours are classes and the *Event Handler* code is placed in a method called **action**.

An agent can execute several behaviours concurrently

Agent Thread Path of Execution



9. SimpleBehaviour Class

Go to LooperAgent Folder

Compile Classs: Looper.java , LooperAgent.java

Run an Instance of LooperAgent.java (call it LA)

In this class 2 instances of Looper class which is extended from SipleBehaviour class are added and compared. Observe the codes. action() and done() methods. Try to write your own Looper class by modifying the existing class, action() and done() methods

10. TickerBehaviour Class

It is a standard scheduling behavior class.
Go to TickerAgent Folder
Compile all the classes
Run an Instance of TickerBehav1 class, observe the result
Run an Instance of TickerBehav2 class, observe the result
Run an Instance of TickerLoopr in which 3 behaviors are added, identify the behaviors and observe the output.

11. CyclicBehaviour

12. Agent Communication

Go to programs/AgentCommunication
Open ReceiverAgent.java and SenderAgent.java and Observe the Program.
Run Jade:
:> java jade.Boot -gui
Run an Instance from ReceiverAgent called R
:> java jade.Boot –container R:ReceiverAgent
Run an Instance from SenderAgent called S1
:> java jade.Boot –container S1:SenderAgent
Observe the Output of R and S, Observe the Communication

- **Communication with User Interface:**

Open SenderAgent_UI.java and Observe the Program
Try to find the differences between SenderAgent_UI and SenderAgent

Run an Instance from ReceiverAgent called R
:> java jade.Boot –container R:ReceiverAgent
Run an Instance from SenderAgent_UI called SUI
Observe the Command line, Enter : “Hello How Are You?”
Observe the Command Line, Enter Text different than above, See how the communication goes by.

13. Multi-agent System Example I

Address:
C:\Programs\Banking
Agents:

- BankerAgent.java
- CustomerAgent.java
- LoanAgent.java

Files:

- Customers.txt
- Loan.txt

1. Upon receiving a Loan Request BankerAgent sends a REQUEST message to CustomerAgent with the content of Name to check if the Name of the applicant is in it's database (Customer.txt)
2. CustomerAgent Search within the Customer.txt File
   - If Found, send a INFORM message back to BankerAgent with the content of "EXIST".
   - If not Found, sends an INFORM message back to BankerAgent with the content of "NOT EXIST".
3. BankerAgent sends REQEST message to LoanAgent with the content of Name of the applicant if Name exists in Customer.txt.
   - LoanAgent stores the Name within the Loan.txt file and sends an INFORM message back to BankerAgent with the content of "Done" if Name does not exist in Loan.txt and if exists, with the content of "EXIST"
4. BankerAgent prints on its command line that "Loan Recorded" if customer did not exist in Loan.txt and it prints "Loan Recorded and Customer is not new".
5. BankerAgent prints on its command line that "Applicant is not a Valid Customer Yet" if name does not exist in Customer Agent

- Go to Programs/Banking folder and compile the entire classes.

  javac *.java

- Run 3 Agents :

  java jade.Boot –gui B:BankerAgent C:CustomerAgent L:LoanAgent

- Change the Content of the Customers.txt file and run the scenario with different parameters as customer name. observe the output
- Run a scenario and run Sniffer Agent to observe the communication amongst agents.

14. Distributed Platform

Machine A: arash-pc
:> java jade.Boot –gui –platform-id Platform1
Machine B:arash-pc2
:> java jade.Boot –gui –platform-id Platform2

Machine A:
:> java jade.Boot –container –host arash-pc2 –port 1099 R:ReceiverAgent

Machine A:
:>java jade.Boot –container S1:SenderAgent
:>java jade.Boot –container S2:SenderAgent S3:SenderAgent S4:SenderAgent

15. Agent with GUI

- Go to programs/AgentGUI
- Open file AgentWithGUI.java and observe the file. This class extends GuiAgent instead of Agent class. You have to Override its onGuiEvent() method. When an actionPerformed () event is triggered you have to create an GuiEvent object and post it to the Agent Event. When a GUI event is posted to Agent, its onGuiEvent method will be triggered and therefore the codes you have written in onGuiEvent will be executed.
- Compile AgentWithGUI.java:

  Javac AgentWithGUI.java

- Run an Instance of AgentWithGUI.class:

  Java jade.Boot –container MyGUIAgent:AgentWithGUI

- A JFrame windows will be appeared. Click on Get My Agent butoon and see the name of the Agent on its Title bar.