KTH H16P02 Distributed Artificial Intelligence and Intelligent Agents: Homework 1

Group 25: Fannar Magnusson (fannar@kth.se)
Thorsteinn Thorri Sigurdsson (ttsi@kth.se)

November 14, 2016

1 Introduction

The objectives of this first homework was to start getting hands on experience with agent platforms. We used the JADE agent platform to implement a Virtual Exhibition, i.e. a Smart museum framwork that consists of three agents: Profiler Agent that requests a virtual tour, Tour Guide Agent that handles the request and builds the virtual tour and Curator Agent, that monitors the museum.

In our solution we assume that we can have many profiler agents that can talk to many tour guide agents. Each tour guide agent is specialized in one or more topics. Finally we have one curator agent that manages the whole museum and holds the list of all artifacts.

The Profiler Agent represents a user (museum visitor) that has certain interests. When the Profiler Agent requests a virtual tour from the tour guides, the tour guides build a virtual tour of the items that are interesting to the user and they are specialized in. The Profiler Agent then picks the virtual tour that has the greatest number of items.

2 Implementation

To begin with we created the three agents: profiler, tour guide and curator. The profiler agent gets the user input, i.e. the users' interests, as a command line parameter. With the interests, the profiler looks up available tour guide agents and sends to each of them a call for proposal for a virtual-tour. That is done with a ticker behaviour that fires every 10 seconds. The ticker behaviour adds a sequential behaviour that starts by sending the proposal to all available tour guide agents in parallel. Then it receives all proposals from the tour guides in parallel and selects and accepts the best tour, that is the one that has the most artifacts. The last step of the sequential behaviour is receiving the virtual-tour. For that the profiler agent uses a MsgReceiver. He waits for a message that matches the ACCEPT-PROPOSAL conversation id. When he receives it, he prints out the virtual tour and then contacts the curator agent asking for details for all the artifacts.

When we start up the program, we can create multiple travel guide agents through command line parameters. Each agent gets a random specilization, for example paintings and sculptures or buildings. Then the travel guide agent registers its virtual-tour service on the yellow pages. The service is a CycleBehaviour that waits for either a call for proposal for a virtual tour, or accept proposal for a virtual tour. When a tour guide agent gets a call for proposal, it expects a parameter indicating the user's interests. It finds the intersection of his specilisations and the user's interests and contacts the curator agent for all artifacts that match this intersection. This is done in a basic behaviour: in step 0 the the tour guide asks for the artifacts and in step 1 the tour guide receives the artifacts and uses a call back function to return the artifacts count to the profiler agent. If the profiler accepts the tour guide's proposal for virtual tour, the tour guide goes into the same basic behaviour, asking the curator for the artifacts, then uses a call back function to return the artifacts and sends the virtual tour to the profiler agent.

Last but not least is the curator agent. We assume that there exists only one curator agent for the platform. He provides two services that he registers to the yellow pages: get-artifacts-for-interest and get-artifact-details. Both of these services are CycleBehaviours. The get artifacts for interests service expects an interests parameter and returns a list of artifacts that match these interests. The get artifact details service expects an artifact id parameter and returns the artifact details, if found.

2.1 Behaviours used

To begin with we did everything using the basic behaviour class provided by Jade, `jade.core.behaviours.Behaviour`, with a `switch` loop to control the processing within each behaviour.

We then refactored the application to use more specialized behaviours. In the end we used the following behaviours:

1. `CyclicBehaviour`
2. `MsgReceiver`
3. `OneShotBehaviour`
4. `WakerBehaviour`
5. `TickerBehaviour`
6. `SenderBehaviour`

7. `Behaviour (basic behaviour)`
8. `SequentialBehaviour`
9. `ParallelBehaviour`

2.2 Directory Facilitator communication

The tour guide agents register the `virtual-tour-guide` service with the DF. The curator agent registers the `get-artifacts-for-interest` and `get-artifact-details` services with the DF.

The profiler agent consumes the `virtual-tour-guide` service to find available tour guides, and the `get-artifact-details` service to get details on artifacts from the curator. The tour guide agent consumes the curator's `get-artifacts-for-interest` service to build virtual tours for the profiler agent.

All service registrations are done during the agents' `setup()` method. The agents that consume services register to get notifications from the DF when an agent that provides a service that they are interested in registers, using the `jade.proto.SubscriptionInitiator` class.

If an agent is attempting some operation (e.g. a profiler agent requests a tour offer from tour guides, or tour guides request artifacts from the curator) but he does not know of an agent that is able to provide the necessary service, we search the DF explicitly, using the `DFService.search()` method. If we are unable to find any agents that are able to fulfill the agent's needs, the operation is aborted.

3 Conclusions

This assignment was very useful in introducing us to the Jade framework, how to build systems using it and the different behaviours that are available. It was also useful to further develop our understanding of the underlying concepts used when developing systems with the agent metaphor.