ID 2209 Distributed AI and Intelligent Agents: Homework 1

avneesh@kth.se, gaiddon@kth.se (Group 6)

November 14, 2016

Introduction

The objective of the assignment was to simulate a virtual smart museum consisting of three interacting entities:

- Profiler Agent representing a user with a profile attached.
- Tour Guide Agent servicing several user agents to accept and process 'create' virtual tours orders.
- Curator Agent representing museum with various artifacts

JADE, an open source platform for peer-to-peer agent platform was the given framework for developing the aforementioned agents.

Design Notes

1. The 3 entities; Profiler, Tour Guide and Curator are implemented as JADE agents by inheriting from jade.core.Agent class.
2. All three agents register to JADE's 'yellow pages' service (DFService Class) as soon as they start-up.
3. Tour Guide Agents depends on Curator Agents to query artifact lists. And therefore it subscribes itself to DF to receive notification whenever an agent with service type 'museum-curator' registers with DF.
4. Profiler Agent searches for agents with service type 'virtual-tour' (used by Tour Guide) in the DF and then sends CFP (call for proposal) message to all found Tour Guide Agents.
5. Tour Guide Agent responds with a PROPOSE message with a price as message content.
6. Profiler Agent compares the prices received from all Tour Guide Agents and chooses the best priced Tour Guide Agent and send to it 'ACCEPT_PROPOSAL' message. In the message content, it also includes the list of 'interests'.
7. Tour Guide Agent queries all the Curator Agents to provide their list of artifacts.
8. Tour Guide Agent then prepares a short list of artifacts by matching Profiler Agent's list of interests. It forwards that short listed artifacts mapped against its Curator Agent Id to Profiler Agent.
9. Profiler Agent then can go through the shortlisted artifacts and gather more details from corresponding curator Agent.

Instructions to run the program

1. Download the source code and create an eclipse project (or net-beans).
2. The system depends on jade.jar and therefore add it to the project's build path as an external jar file.
3. Create a run configuration for the project with jade.Boot as main class and -gui as program arguments.
4. Run the program. This will launch JADE's Remote Agent Management GUI.
5. Select 'Main Container' in the left AgentPlatforms tree.
6. Go to Action Menu and choose 'Start New Agent' to start the three Agents with different local names.
    a. Tour Guide Agent : Choose class agent.TourGuideAgent and provide some price (say, 5) as agent argument. One may start a number of agents with different price values.
    b. Curator Agent: Choose class agent.CuratorAgent to start the CuratorAgent
    c. Profiler Agent: Choose class agent.ProfilerAgent and provide a list of comma separated interests [valid values : Nature,History,Science]
    Note: the agents can be started in any order and in any number.
7. In the console output, Profiler Agent shall log the list of artifact details matching its interests.

Conclusion

In the agent implementation, the following JADE provided behaviours were used: CyclicBehaviour, SimpleBehaviour, SimpleAchieveREInitiator Behaviour , SequentialBehaviour, and ParallelBehaviour.

Also JADE's,Directory Facilitator (DF) service was used to register, subscribe and look up agents at run-time. The JADE's multi-agent framework provide convenient APIs to start agents and also to pass messages among agents. It also allows distributing agents across a network.