Trust in information sources on a competitive and distributed environment

João Pedro Dias, João Correia and Eduardo Martins

Faculdade de Engenharia da Universidade do Porto, Portugal
{ei11137,ei12159,ei11104}@fe.up.pt

Abstract. Distributed systems based on agents are sometimes used in competitive environments with information sources that we have little or none knowledge about. In these cases we need to make the agents take the best decisions on what information source is most valuable to them. For this we have little more than the agent experience itself. This paper simulates this type of environment approaching a question and answer game, where the agents have to decide the best information source for each question based on question thematic. For this experiment we use a set of computation trust algorithms for making decisions on the best information sources for each question, and we verified that the most simple approaches are capable of good results when the information sources maintain the same behaviour throughout the execution. Complex approaches can get us better results when the informations sources behaviour changes.

Keywords: Artificial Intelligence, Multi-Agent, Computational Trust, Distributed Systems

1 Introduction

Multi-agent distributed systems are in use in a lot of cases from smart-grids[1] to autonomous on-line auctions[2]. There are some cases when the agents in this systems need to take actions based in information sources which the agents don't know about. And, additionally, there are cases where this agents are competing between themselves. In these cases there is a need to use computation trust algorithms capable of choosing and using the most valuable information sources for each type of information we needed and get better results over other agents.

The purpose of this work is to examine the efficiency of different trust algorithms against a similar environment.The specific goals are:

- To demonstrate that trust algorithms are more suited for this type of environment.
- To demonstrate a possible architecture for this kind of systems.
- To ensure all the communication between agents.

In this paper we start presenting the chosen scenario at section 1.1, followed by identification of the agents in the systems and algorithms used in section **??**.

Next we show in section 3 the used tools and the interaction protocol. Finally we present all the experiments realized at section 4 and a conclusion in section 5.

1.1 Scenario description

The competitive environment is easily replicated and modelled as a game of questions and answers, where every player is an agent and there is a list of information sources with one master agent responsible for validating the answers by the players.

With the purpose of adding variety to the information sources, there are questions distributed between 5 thematics: Sports, History, Science, Arts and Places. The information sources can have specific knowledge in one theme or more as well they can be totally random.

The game sequence begins with the master doing a random question to all players with four answer options. Then all the players in-game choose one and only one information source that they will use to answer that question. The choice of the information source correctly is the core function of the players, since the player answers depending on that. This choice depends on the trust algorithm used by the player and by the category of the question. The game ends when the master reaches the number of questions defined to be done.

2 Specification

2.1 Agents identification and specification

With the purpose to have the most precise simulation of the environment described, there was the need to implement different types of agent behaviours that can represent every role in the system, namely, a master, various players and various information sources, each one with a different role as identified in Table 1.

Specifying each one of the agents present in the system, we have four players:

- Player *RANDOM*: A player with no trust algorithm implemented which chooses one expert to answer randomly.
- Player *DUMMY*: A player which uses a very simple algorithm that build a matrix based on trust (experts/theme). Specified in section 2.2.
- Player *FIRE*: The player uses the *FIRE* algorithm [4]. Specified in section 2.2.
- Player *BETA*: The player uses the *BETA* algorithm [5]. Specified in section 2.2.

and seven experts as information sources:

- *Sports* Expert: Have 100% expertise in questions about Sports.
- *History* Expert: Have 100% expertise in questions about History.

Agent	Input	Output	Specification
Master	Number of iterations and Questions/Answers Database	Execution statistics (number of right/wrong questions by player)	Game engine, responsible for making questions and validating answers.
Player	Theme, Question, 4 answer options	Answer to the question.	Ask one of the *experts* for the answer. This choice depends on the trust algorithm implemented on the player.
Expert	Question.	Answer for the given question.	Answer the question made even if its not in the knowledge domain of the expert. If the question its outside the knowledge domain it will give a random answer.

Table 1. Different agents tasks.

- *Science* Expert: Have 100% expertise in questions about Science.
- *Places* Expert: Have 100% expertise in questions about Places.
- *Arts* Expert: Have 100% expertise in questions about Arts.
- *All* Expert: Have 100% expertise in all thematics.
- *Random* Expert: Have a 25% probability of answer correctly (the number of possible answer is 4).
- *Variable* Expert: Have a change to get right all the first 50 questions or less.
- *Common* Expert: Have 50% knowledge over each thematic.

2.2 Trust algorithms

The various players in game have different behaviour accordantly with the trust algorithm they use. As listed we have three trust algorithms implemented. This algorithms in major part are only experience based due to the competitive environment.

The *DUMMY* player who uses a basic approach of building a matrix given points for each information source depending on its performance on each theme, in this case specifically, we give one positive point for each right answer and one negative for each wrong answer. In each iteration the player will use the experts with the biggest punctuation for that question theme. This algorithm implementation is described on Algorithm 1.

The *BETA* player who uses the BETA reputation system [5] for choosing the best information source for each question. This algorithm uses, as said in *The beta reputation system*, the *"beta probability density functions to combine feedback and derive reputation ratings"*[5].

The *FIRE* player who uses an approximation to the FIRE algorithm [4] to choose the expert to use. This algorithm uses its experience, but uses as well opinions from other agents, dividing itself in four different parts, two directed to

Algorithm 1 DUMMY Algorithm

```
1: function PONTUATION(A[cat][exp], answer, cat, exp)
2:     if answer == true then
3:         A[cat][exp] ← A[cat][exp] + + ;
4:     else
5:         A[cat][exp] ← A[cat][exp] − − ;
6:     end if
7: end function
8: function GETBESTEXPERTBYCATEGORY(A[cat][exp], cat)
9:     tempPoints ← −∞;
10:    tempExpert ← NULL;
11:    for i ← 1, size(A[category]) do
12:        if A[i] == cat then
13:            for j ← 1, size(A[i][expert]) do
14:                if A[i][expert] > temp then
15:                    tempPoints ← A[i][expert];
16:                    tempExpert ← expert;
17:                end if
18:            end for
19:        end if
20:    end forreturn tempExpert;
21: end function
```

experience and two directed to opinions. For this specific case this algorithm can be a bad choice since it's a competitive environment and the communication between concurrent agents can be non-existent. So, in our specification, this player will have a collaborator that acts like another normal player with experience-based behaviour, but with the difference that it will give an opinion to the *FIRE* player with the ratio of wrong/right questions for each expert existent in game.

2.3 System architecture

The system architecture follows accordantly with the framework used (described in section 3.1), ensuring all the synchronous communication needs between agents for the correct simulation and evaluation. In this architecture as showed in Fig.1 we have one and only one master agent, a minimum of one player and a minimum of one information source.

3 Development

3.1 Tools

For implementing the system its used the JADE[1] framework, that is fully implemented in the JAVA programming language, and so it is cross-platform. It

[1] JAVA Agent DEvelopment Framework is an open source platform for peer-to-peer agent based applications.

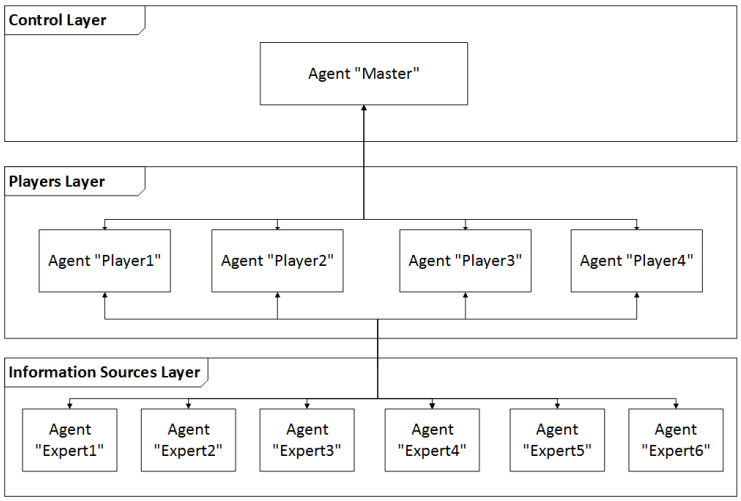

Fig. 1. System architecture. The number of agents *player* and *expert* can vary.

simplifies the implementation of this kind of system through a middle-ware that complies with the FIPA[2] specifications for multi-agent systems [7].

This middle-ware consists in various components based on FIPA standard architecture (Fig.2):

- The Agent Management System (AMS) controls the platform. Is the only one who can create and destroy other agents, destroy containers and stop the platform. Also, it contains a list of Agent Identifiers (AID) with every agent on the system.
- The Directory Facilitator (DF) provides a directory which announces which agents are available on the platform.
- The Message Transport Service (MTS) is the default method of communication between agents. Each message its compliant with the FIPA Agent Communication Language (ACL).

3.2 Interaction protocol

Due to the JADE being message compliant to the "FIPA ACL Message Structure Specification" give us the capacity of label all the communication between agents with FIPA performatives [6].

With this, it was established a interaction protocol between the agents in the system as showed in Fig.3. The execution start with the master asking a question and it enters in a loop of question followed by a answer by the player and a validation of the answer by the master. It ends when the master reach the pre-defined number of questions to make.

[2] Foundation for Intelligent Physical Agents is a body for developing and setting standards for heterogeneous and interacting agents and agent-based systems.

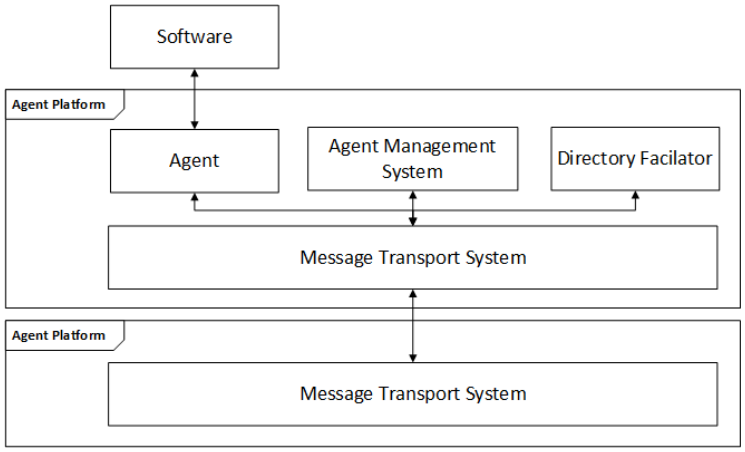

Fig. 2. Agent Management Reference Model

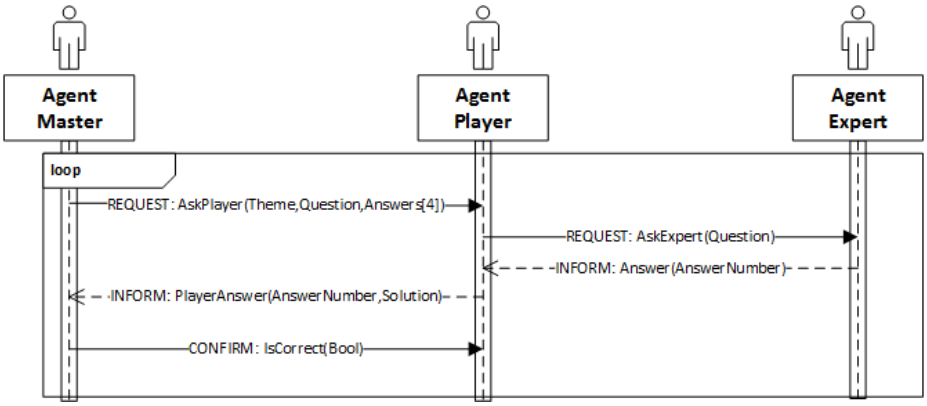

Fig. 3. System's sequence diagram.

4 Experiments & Discussion

To test the system developed it was carried out some experiences, with the various agents developed and presented in section 2.1, analysing the different results obtained. All tests are made using the same question database that contains five question per thematic. Additionally its used a fixed number of questions, in the case, 200 questions.

To certify that all communication are been done correctly it was made various experiences with one single expert with 100% knowledge in all thematics.

The experiments carried give us a perspective of the behaviour of the different algorithms used as showed in section 2.2. So for a better capacity of presenting more precise conclusions we made a set of experiments with different groups of experts.

Between all the experiments done, the results of the different players changes during the different tries for each experiment, more specifically:

- *FIRE* player varies in the maximum of 21 answers;
- *DUMMY* player varies in the maximum of 16 answers;
- *RANDOM* player varies in the maximum of 18 answers;
- *BETA* player varies in the maximum of 11 answers;

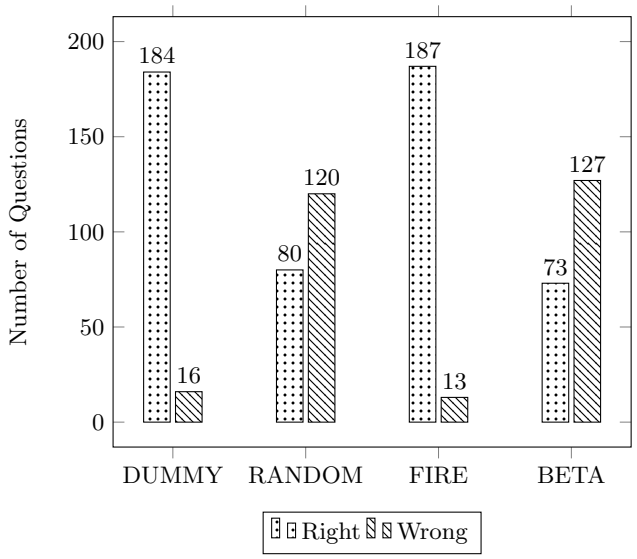

Fig. 4. Experience with five experts with 100% knowledge by question theme.

In the first experiment, Experiment 4, we used one expert per thematic, each one with 100% knowledge in each theme. This is our first experiment used to

analyse the behaviour of the different players in a environment were the information by thematic is distributed among different sources. The results pointed out by the experiment showed us that the algorithms that use classification of experts by thematic, like FIRE or DUMMY, given us better results than the algorithms that just analyse the number of questions answered right or wrong by expert like the BETA algorithm.

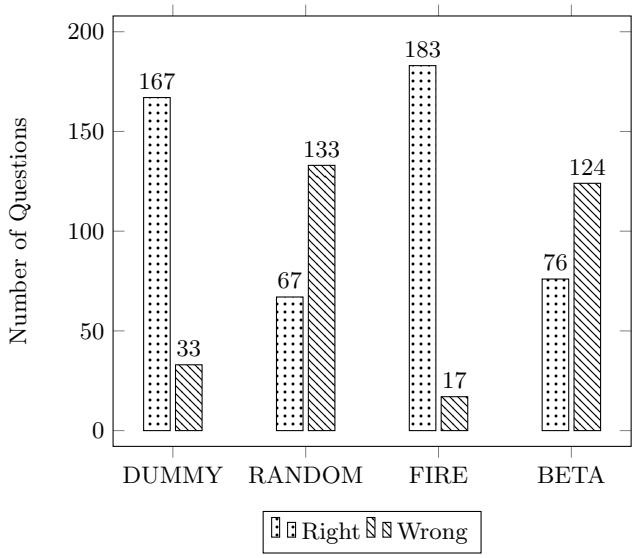

Fig. 5. Experience with five experts with 100% knowledge by question theme and one random expert.

The Experiment 5 showed us that even when we add a random expert to the Experiment 4, the results given by the player have small changes only.

In the Experiment 7 we verified how that the lack of knowledge is manage by the different players. The results showed us that the percentage of right answers decreases due to this lack of knowledge, since the players are limited too the only two players that exists in the environment.

The Experiment 6 we verified how that the lack of knowledge is manage by the different players in a environment where exists a random information source. The results showed us that the percentage of right answers decreases as showed in experiment 7 and the addition of the random expert don't affect significantly the results.

At the Experiment 10 we add the *Variable* expert. This experiment give us a perspective of the way that the different algorithms used adapt to the addition of a variable information source. Namely, this showed that the *FIRE* and *DUMMY* algorithms are more slow to adapt to this changes and, on the other hand, the *BETA* adapts more quickly to this change.

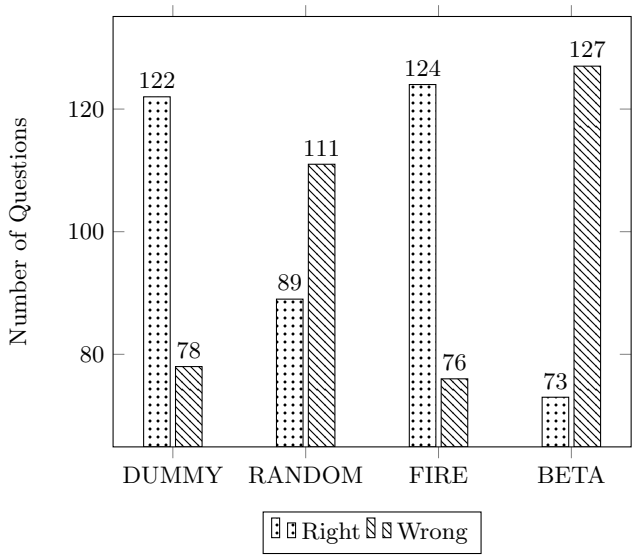

Fig. 6. Experience with two experts with 100% knowledge in the thematics Places and History

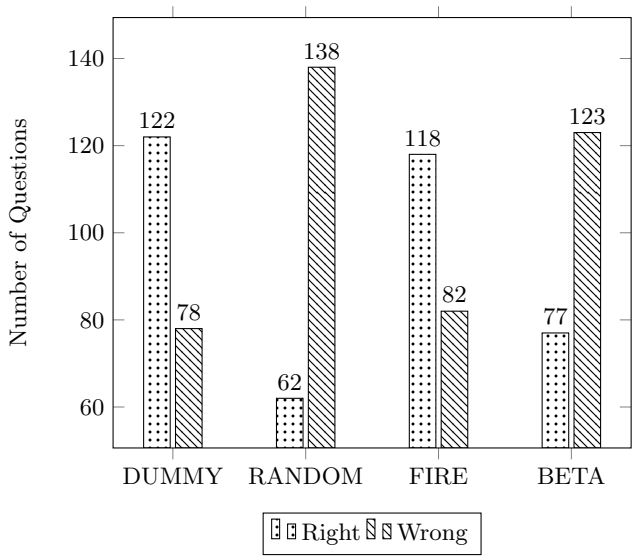

Fig. 7. Experience with two experts with 100% knowledge in the thematics Places and History, plus a random expert.

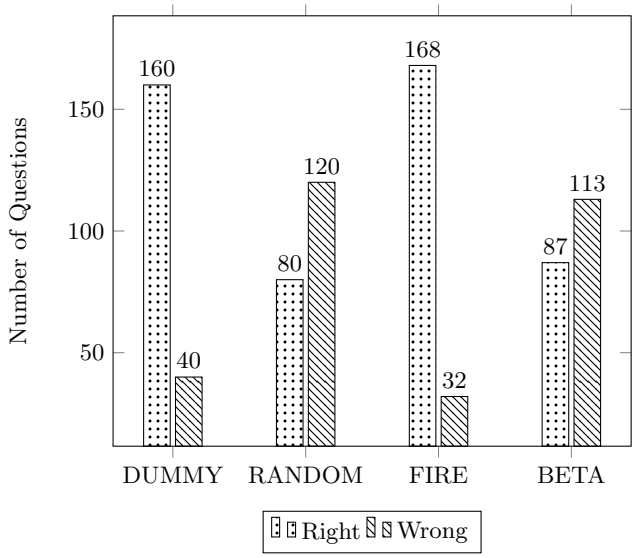

Fig. 8. Experience with five experts with 100% knowledge in each thematic and the *Variable* expert

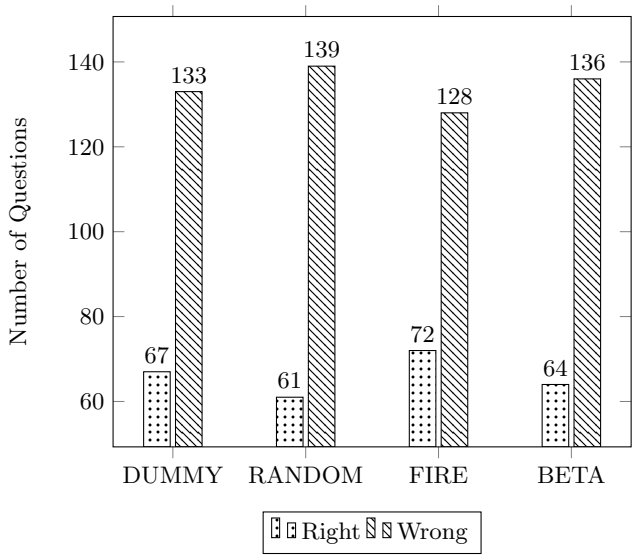

Fig. 9. Experience with the *Variable* expert and the *Common* expert.

The Experiment **??** adds to the environment the *Variable* and the *Common* experts only. This showed us how the different players manage the existence of one information source with only a percentage of knowledge(*Common*) and one with variable knowledge in the execution runtime. At the end of the experiment we verified that the *BETA* adapts well to this environment given similar results to the other experiments and the other players suffer more in this environment given worst results than in the other experiments.

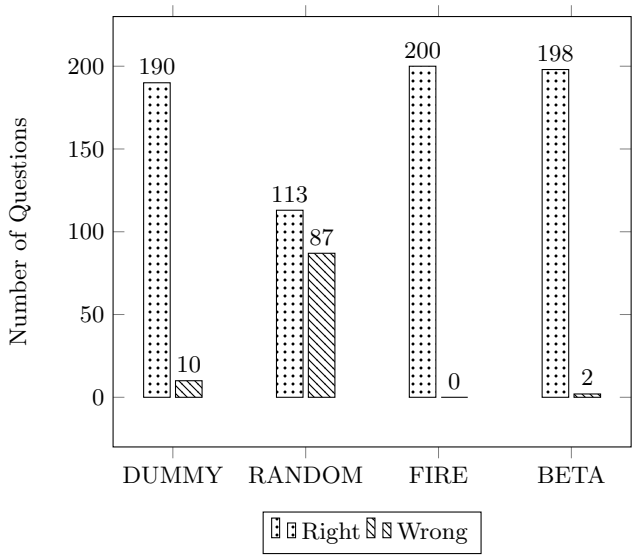

Fig. 10. Experience with the *Variable* expert, *Common* expert and the *All* expert.

In the environment showed in Experiment **??** we show how quickly the players find the best information source, in the case, the *All* expert. We verified that the *FIRE* algorithm give us excellent results since it have a opinion source. By other side if the *FIRE* couldn't use the opinion, it will have a behaviour similar to the *DUMMY*. This show us that the *BETA* algorithm is one of the best adapted algorithm to this case, since its rapidly find the best and more trustful information source.

5 Conclusion

The purpose of this work was to make a comparative between different trust algorithms implemented in a multi-agent system in a distributed and competitive environment. In this context, for the development of this work it was used the framework JADE which has showed to be a simple and powerful way to develop the agents and manage all the communication between them.

We made various experiments with the computational trust algorithms implemented, namely the Fire and Beta algorithms, and additional the simple Dummy algorithm created by us. In the experiments we verified that the simple algorithms which choose information sources by each information needs, in this case by each thematic, performance better. Examples of this are Dummy algorithm and even the Fire because they categorize the experts by thematic. By other side, we verified that worst results are presented by the trust algorithms that uncategorised the experience by thematic, in this case, the Beta algorithm and the Random approach.

This is verified when the behaviour of the information sources don't change during the execution. When this change through the execution, all the algorithms verify a loose of performance, but this is more noticeable with basic algorithms like the Dummy, while other algorithms such as the Fire adapt more quickly to the changes in information sources.

So, we recommend that when isn't certain that the information sources will maintain the same behaviour through the execution we have to use more complex algorithms like the Fire algorithm. On the other hand, if we know that the information sources used are trustful and their behaviour will not change spontaneously, we can use more simple algorithms like the matrix-based punctuation Dummy algorithm.

Acknowledgments

We would like to thank the freedom and support that Professor Henrique Lopes Cardoso has given us to adapt a scholar project in this paper.

References

1. Pipattanasomporn M., Feroze H. and Rahman S., Multi-agent systems in a distributed smart grid: Design and implementation, 2009.
2. Ivaschenko A. and Lednev A., Auction Model of P2P Interaction in Multi-Agent Software,2013.
3. Vaucher J. and Ncho A., JADE Tutorial and Primer, http://www.iro.umontreal.ca/ vaucher/Agents/Jade/JadePrimer.html, 2004.
4. Huynh T., Jennings R. and Shadbolt N., FIRE: An Integrated Trust and Reputation Model for Open Multi-Agent Systems, ECAI 2004, IOS Press, 2004.
5. Jøsang A. and Ismail R., The beta reputation system. Proceedings of the 15th Bled Electronic Commerce Conference (Vol. 160), 2002.
6. Foundation for Intelligent Physical Agents, FIPA Communicative Act Library Specification, 2002-12-06. http://www.fipa.org/specs/fipa00061/
7. Foundation for Intelligent Physical Agents, FIPA Agent Management Specification, 2004-03-18. http://www.fipa.org/specs/fipa00023/