A Logic for Expressing Trust in Intelligent Agents

Juliana Carpes Imperial
TECMF – Departamento de Informática
Pontifícia Universidade Católica do Rio de Janeiro
(PUC-Rio)
Rua Marquês de São Vicente 225 – Gávea – 22.453-900
Rio de Janeiro – RJ – Brazil
Email: juliana@inf.puc-rio.br
Telephone: (+55 021) 3114-1500 (4054)

Edward Hermann Haeusler
TECMF – Departamento de Informática
Pontifícia Universidade Católica do Rio de Janeiro
(PUC-Rio)
Rua Marquês de São Vicente 225 – Gávea – 22.453-900
Rio de Janeiro – RJ – Brazil
Email: hermann@inf.puc-rio.br
Telephone: (+55 021) 3114-1500 (4345)
Fax: (+55 021) 2511-5645

***Abstract*—Trust is a fundamental concern in large-scale open distributed systems. It lies at the core of all interactions between the entities that have to operate in such uncertain and constantly changing environments. Given this complexity, these components, and the ensuing system, are increasingly being conceptualised, designed, and built using agent-based techniques. Therefore, the presence of trust is imperative in a multi-agent system (MAS). Consequently, this work studies how to have a explicit trust model in an intelligent agent, which has beliefs, desires and intentions (BDI agent). That is, the agent now has a fourth component called Trust. This way, a logic to include the concept of trust in an open BDI MAS is interesting, so that the different aspects of a trust model can be expressed formally and accuratelly. This is achieved by using an indexed multi-modal logic, where the possible worlds which model a multi-agent system represent which agents are in the system in a given moment. Moreover, for each one of the three original components of a BDI agent, where the components represent beliefs, desires and intentions, there is a representation of possible worlds, because these are treated as modalities. However, trust is modelled as a predicate, not as a modality.**

I. Introduction

Many computer applications are open distributed systems in which the (very many) constituent components are in a decentralised control regime, and which are subject to constant change throughout the system's lifetime. Examples include peer-to-peer computing, the semantic web, web services, e-business, pervasive computing environments, and many others. In all of these cases, however, there is a need to have autonomous components that act and interact in flexible ways in order to achieve their design objectives in uncertain and dynamic environments [1]. Given this, agent based computing has been advocated as a natural computation model for such systems [2], [3].

More specifically, open distributed systems can be modelled as open multi-agent systems (MAS) [4] that are composed of autonomous agents that interact with one another using particular mechanisms and protocols. In this respect, interactions form the core of multi-agent systems. Thus, there are a number of models of interactions between agents [3]. However, their application in large-scale open distributed systems presents a number of new challenges. First of all, the agents are likely to represent different stake holders where each one has its own aims and objectives. Secondly, given that the system is open, agents can join and leave at any given time. Thirdly, an open distributed system allows agents with different characteristics to enter the system and interact with one another. Fourthly, an open distributed system allows agents to trade products or services, and collaborate in very many ways. Therefore, agent designers are faced with a choice of a number of potential interaction protocols that could help them to achieve their design objectives.

A protocol defines how the agents must interact with each other, that is, it can be seen as a set of rules. Moreover, due to security techniques, it helps to ensure the security of the system by imposing restrictions on the way the agents interact. However, very restrictive protocols can be impractical and security techniques do not guarantee the quality of the behaviour of the agents during the interactions.

In such cases, it is left to the agents to decide how, when, and with whom to interact without any guarantees that the interaction will actually achieve the desired benefits. To make such decisions would require agents to be fully informed about their opponents, the environment, and the issues at stake.

However, both the system and the agents may have limited computational, observational, and storage capabilities that restrict their control over interactions. Thus, in practical contexts it is usually impossible to reach a state of perfect information about the environment and the interaction partners' properties, possible strategies, and interests [5], [6], [7]. Agents are therefore necessarily faced with significant degrees of uncertainty in making decisions. In such circumstances, agents have to trust each other in order to minimise the uncertainty associated with interactions in open distributed systems.

An open MAS is composed of many agents, which are defined in this work using the BDI (belief-desire-intention) model [8]. This model is used because it has its roots in the philosophical tradition of human being practical reasoning, it has a software architecture that can be implemented in real systems, and a family of logics that supports a formal theory of intelligent agents.

Since trust is fundamental to have interactions between agents, the agent must have a trust model that says which agents it can trust or not. Therefore, in this text, the BDI

agent has a new layer representing its trust model explicitly. This is done because it is believed that the trust in another agent should not be described just as a belief.

To provide a formal semantics to an open MAS with trust, modal logic is used [9]. With this, it is possible to model xusing the possible worlds semantics, where each world represents which agents are in the system in each time. This way, the external part of the system is modelled, but there is also the need to model the mental states of all the agents. Thus, a semantics must be given to them. The approach adopted is to have the original BDI components as modalities and trust as a first-order predicate related to the beliefs of an agent, which is defined according to what can be understood as trust in another agent. With this formalization, it is possible to formulate desirable properties and axioms so that the open MAS works as expected.

The rest of this paper is organized as follows. In the next section, it is explained why trust is considered to be different from belief. Section 3 shows how to include trust in a BDI agent. Next, the logic to formalize the model is presented. In section 5, some relations concerning the trust model are shown. In the following section, some related work is described. Finally, this paper ends with the conclusions and future work.

II. Trust x Belief

Although trust and belief are similar concepts, a belief is an internal representation of something that the agent perceives in its environment. A belief will be wrong if the agent cannot perceive its environment adequately. Even though it is possible to say the same thing about trust, a trust in another agent can be wrong even if the former can correctly perceive the behaviour of the latter. For this to happen it is only necessary that the agent being observed deceives the other. Thus, whenever an agent trust another or not, even if it can perceive correctly the behaviour of its partner, it can not be sure that its behaviour will be as expected when interacting with it. That is, trust is a kind of bet, which can result in gain or loss [10].

Obviously, trust in another agent could be dealt just as a belief that a certain agent is trustful. However, to put it in a more abstract level, separated from the other components of a BDI agent was the selected choice. In addition to this, if trust is in the same level as belief, the fact that an agent trusts another is the same as the fact that an agent believes that the other is trustful, turning the definition circular. Other difference is that in many situations the trust is not established because of an observed behaviour of another agent, but it is due to suppositions about others behaviour, which can be right or wrong. This is a fact specially when an agent has to deal with an unknown partner.

In this text, the term trust will always be used between agents, although it could be used to refer to messages and actions of other agents. This approach was chosen since this is the way the subject is treated in the MAS literature. When an agent trusts another thing which is not another agent or related to its behaviour, this will be treated just as a belief in this thing if the agent cannot be deceived by this thing. If it is about the behaviour of another agent, such as a message sent by it, the beliefs of the agent will have a representation that the other agent said something. This representation is similar to what is done in BAN [11]. After all, the agent cannot be sure that the sent message is true. The representation that an agent sent a message is considered to be a belief because it is a percept of the agent environment.

III. Trust in a BDI Agent

Figure 1 shows how to include trust in a generic BDI agent [4]. When trust is included in the model, the functions which update the beliefs ($brf()$), the desires ($options()$) and the intentions ($filter()$) now depend on trust. Moreover, the new function which updates trust ($trf()$), also depends on the beliefs.

Basically, there are four ways to implement the trust that an agent has in another:

- The first option is the binary trust. That is, the agent just trusts or distrusts other agents. Nevertheless, this model is not considered to be realistic because it is possible to trust more or less another agent. Moreover, with binary trust, how should be classified an unknown agent? However, it can be said that trust is in fact binary and that when an agent chooses to interact with another, in that moment it is trusting its partner. The same can be said when the agent chooses not to interact with another one;
- Secondly, there are a fixed number of trust levels depending on the model to be implemented. Supposing that the values range from 0 to 1, if there are n levels, they would be 0, $\frac{1}{n-1}$, $\frac{2}{n-1}$, ..., $\frac{n-2}{n-1}$, 1. This is still rigid, because there are only fixed pre-determined levels of trust;
- Thirdly, there can be the case where the trust level is simply a value ranging from one value to another, usually from 0 to 1. This has the problem of having only one global value of trust, and
- Finally, it is possible having many degrees of trust for various aspects of what is expected from an agent such as integrity and the many possible aspects of competence. In this case, each one of these degrees can be implemented using one of the above approaches [3].

The trust degree is updated whenever an agent receives a message from another agent. The sending of a message is considered to be a percept, which is used to update the beliefs of both agents. The recipient updates its beliefs with the information that it received the message. The update of the beliefs obviously depends on the trust that the agent has in the one which sent the message. After all, the contents of a message sent by an distrustful source should not be taken into consideration. The update of the beliefs with the contents of the message can only occur when the agent has a meaningful degree of trust in the recipient. After updating the beliefs (with the fact of the receiving of a message, or just with its contents, or both), depending on the consistency of the information given by the other agent with its own beliefs and trust model,

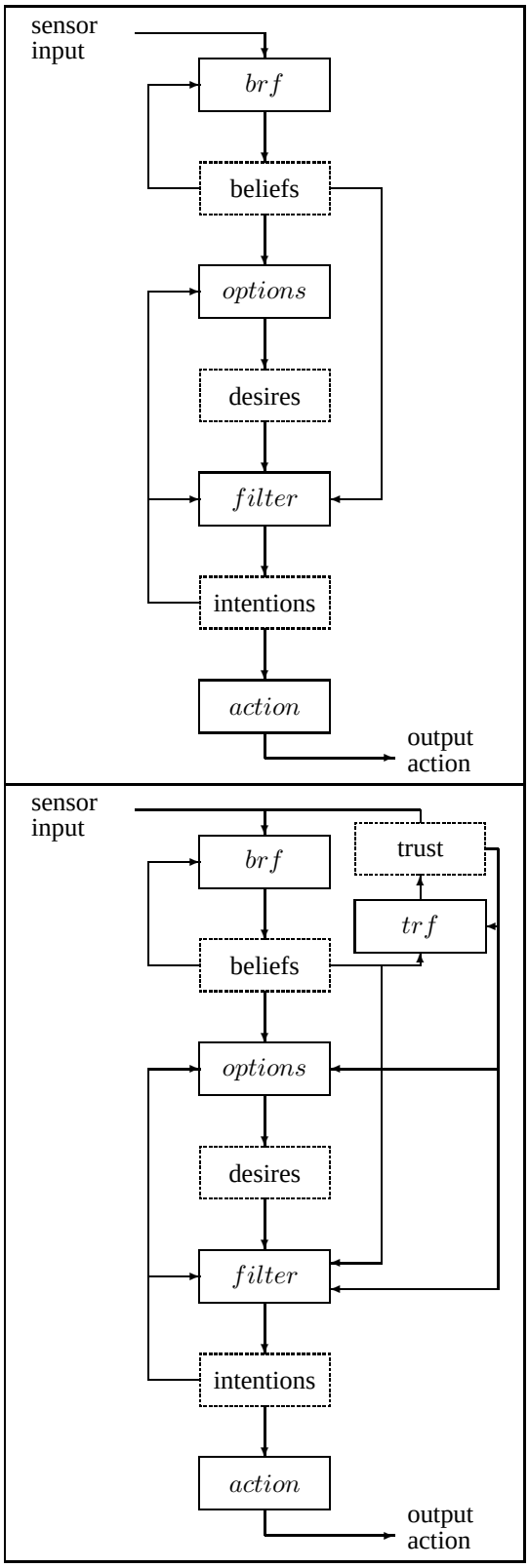

Fig. 1. Schematic diagram of a generic BDI agent architecture without and with trust, respectively

it will update its trust model, increasing or decreasing the trust degree concerning the agent which sent the message or other agents, if the message is about the behaviour of other agents.

The agent should always take into consideration the trust it has in its partners, preferring an honest and capable agent to execute the desired task. When delegating a task to another agent, it expects that the other intents to execute it.

When an agent decides to interact with others to achieve an intention, some actions of its plan must include messages to the others, maybe with tasks to be delegated. For the intention to be achieved, the other agents must do what they were asked for with the desired quality. In general, the agent needs to verify if its partner did exactly what was asked. This verification can be part of the plan. That is, some actions of the plan must be verifications to find out if the delegated task was concluded with success. Otherwise, this verification can be registered in the intention itself. In this case, when the agent verifies if the intention was already concluded, or is possible to be concluded, or the agent is already motivated to achieve it in its control loop [8], it can also verify if the task was done adequately. Depending on what the other agent did, the plan can be discarded and even the intention can be discarded, if there is not another agent that can be of help.

Finally, the agent must register the message sent and the delegation asked in its beliefs because it is a modification in its environment. Moreover, this information can be useful in practical reasoning and trust updating.

IV. The Logic Model

A. Dealing with the System

The model used can be seen as an extension of LORA's first-order multi-modal logic [8], which is an indexed one. First, just the system is modelled, showing which agents are present in a given moment and how the time passes. Since the system is open, at each time agents can enter, leave or stay in the system. In addition, using the possible worlds semantics [9], at each time there is a transition from a world to another, which can be the same if the set of agents does not change. That is, even if no agents leave or enter the system, they are executing actions and possibly interacting with each other, which is not instantaneous. Furthermore, they may be altering their mental states, which also takes time.

In this paper, time is considered to be global. That is, the time passes in the same way and is the same for all the agents in the system. The model of time and the relation between the possible worlds and how they evolve can be found in [8].

The presence of an agent is given by the propositional variables a_0, a_1, ... corresponding to the agents 0, 1, ... That is, the variable a_i represents the agent i. When an agent is in the system, the value of the variable is $\top$. If not, it is $\bot$. The individual variables concerning the agents have rigid designation (they have same meaning in all the worlds). Concerning the logical language, $\Box$ represents "necessity" and $\Diamond$ represents "possibility". The time is implicit in the logic, which is always passing.

For a generic MAS, the model is reflexive and a graph containing cycles, where each world can be related with all the

others. This happens because at each time any agent can enter, leave or stay in the system. The generic system is obviously non-deterministic, because the agents which will leave, stay or enter are usually not known. However, in practical systems, this may not necessarily happens. For instance, if new agents can enter but not leave the system, the future worlds cannot have transitions to worlds with the same set of agents of past worlds. Therefore, depending on the behaviour of the system, certain modal logic axioms can be valid or not. The table below shows some axioms and the kind of system in which they are valid. More details about the axioms can be found in [9].

Axiom	**Formula**	**Kind of System**	**Description**
K	$\Box(a_i \rightarrow a_j) \rightarrow (\Box a_i \rightarrow \Box a_j)$	all systems	It is always valid.
T	$\Box a_i \rightarrow a_i$	reflexive systems	The set of agents can be repeated in the future.
D	$\Box a_i \rightarrow \Diamond a_i$	serial systems	There are no final states (finiteness is an undesirable property).
4	$\Box a_i \rightarrow \Box\Box a_i$	transitive systems	The agent can enter the system but cannot leave it.
5	$\Diamond a_i \rightarrow \Box\Diamond a_i$	euclidean systems	If the set of agents can change in two ways from one future configuration, it is possible to reach the other and vice-versa.
B	$a_i \rightarrow \Box\Diamond a_i$	symmetrical systems	The set of agents in the future can be identical to a past one.
$Triv$	$a_i \leftrightarrow \Box a_i$	one reflexive world	The system is not open.
Ver	$\Box a_i$	one dead end	The system is stopped (undesirable property).
D_C	$\Diamond a_i \rightarrow \Box a_i$	deterministic system	The set of agents can change only in one way (or stay the same), or the system is stopped.

TABLE I
MODAL LOGIC AXIOMS AND SYSTEMS WHERE THEY ARE VALID

It is important to highlight that the axiom K is valid in all the systems shown in table I. Moreover, there are systems where more then one axiom besides K are valid. This way, it is possible to have other properties for the systems. For instance, the generic MAS, in which agents can leave, enter or stay in the system in the next worlds, satisfy T and 5, known as $S5$. Examples of systems in which agents can enter or leave at any time are the ones concerning mobile computing.

Concerning the axiom 4, this axiom says that an agent in the system in all future worlds will still be present in the following future world. That is, if there are agents after the initial state of the system, they cannot leave it.

A system in which $(a_i \wedge a_j \wedge \Diamond(a_i \wedge \neg a_j)) \rightarrow \Box a_i$ and T are valid, from one world to another, the set of agents can be modified in only one way. That is, the set of agents can be the same or be modified at most in only one way. In possible worlds semantics, this means that there is only a transition from a world to itself and at most a transition to only one other world.

B. Dealing with the Agent

The logic used is an extension of LORA [8], which uses CTL^* syntax. Therefore, only the additional part and the differences are shown in this paper due to lack of space. Syntactically, the inclusions are below, where i and j are agents, c is a capability, n is the number of levels of trust, v is the degree of trust, α is an action and φ is any formula of the logic. α denotes a term of sort $action$. A specific grammar definition is omitted due to lack of space.

- the modality $said$ ($(said\, i\, \varphi)$);
- First order predicates to refer to the different kinds of trust models:
  - $TrustB$ – TrustB(i, j) (pure binary);
  - $TrustF$ – TrustF(i, j, n, v) (fixed trust levels);
  - $TrustV$ – TrustV(i, j, n) (any trust value);
  - $TrustPB$ – TrustPB(i, j, c) (binary with capabilities);
  - $TrustPF$ – TrustPF(i, j, c, n, v) (fixed trust levels with capabilities), and
  - $TrustPV$ – TrustPV(i, j, c, v) (any trust value with capabilities).
- Actions of agents:
  - $says$ – an agent says something ($says(i, \varphi)$);
  - do – an agent delegates an task to another ($do(j, \alpha)$);
  - in – an agent enters the system ($in(i)$) , and
  - out – an agent leaves the system ($out(i)$).
- $capabilities()$ – function that returns the capabilities needed to do or say something ($c \in capabilities(\varphi)$).

Due to lack of space, the logic semantics is omitted. What is present and explained in [8] is also suppressed. Note that the messages to other agents seem to be it broadcasts, since there is no information concerning the recipient. However, this information can be passed in an above layer of the communication protocol. Thus, this does not need to be present in the messages. Furthermore, LORA has no specific actions such as in, it only has generic actions called α and composition of actions.

Some properties of the logic must be explained. The beliefs are represented by a $KD45$ modality (Bel). However, the desires and the intentions are represented by KD modalities (Des and Int, respectively) [8]. The new modality introduced in this work ($said$) can be a K or a KD modality, depending on the model desired. If the agent cannot say the opposite of what has said before, then it is a KD modality. In the other case, it is just K. An example where only K is valid for $said$

is the case in which an agent sends messages about its percepts about its environment. If the environment and some property described by it turns out to be not true any more, D will not be valid for this modality because it will say something contradicting what it has already said.

An important property is that if some information is in one of its mental state, then the agent believes that this information is in the respective mental state. Furthermore, the converse is also true. That is, if the agent believes that something is in one of its mental state, then it is in the respective mental state. Below, $\models_S$ means that the formula is a valid state formula for the logic.

$\models_S (Bel\, i\, (Bel\, i\, \varphi)) \leftrightarrow (Bel\, i\, \varphi)$
$\models_S (Bel\, i\, (Des\, i\, \varphi)) \leftrightarrow (Des\, i\, \varphi)$
$\models_S (Bel\, i\, (Int\, i\, \varphi)) \leftrightarrow (Int\, i\, \varphi)$
$\models_S \forall i \forall j((Bel\, i\, TrustB(i,j)) \leftrightarrow TrustB(i,j))$
$\models_S \forall i \forall j((Bel\, i\, TrustPB(i,j,c)) \leftrightarrow TrustPB(i,j,c))$

In the above formulas and for the rest of the paper, the cases concerning the trust models with more than two values are omitted because of lack of space. It must be noted that quantifiers are not used related to the "variables" bounded to the modalities. This happens because, in fact, the "variable" identifying the agent is not a parameter of the modality, but an index. After all, there is a modality of each type for each agent.

One important property about the modality $said$ is that an agent believes that has said something to another: $\models_S (said\, i\, \varphi) \rightarrow (Bel\, i\, (said\, i\, \varphi))$. In addition to this, if an agent i says something and j perceives this, it must store this information. Therefore: $(said\, i\, \varphi) \rightarrow (Bel\, j\, (said\, i\, \varphi))$. Thus, the modality $said$ can also be viewed as an additional layer specific for agent communication to the sets of the beliefs of an agent.

V. Some Properties about Trust

In this section, some relations of the trust model of an agent with its other mental states and the behaviour of other agents are shown. This is done using formulas of the logic described above. Due to lack of space, only some properties are presented.

First of all, in the pure binary case, if pathological agents are not considered (they are rational), it can be assumed the trust in itself as an axiom: $\models_S \forall i\, TrustB(i,i)$.

When dealing with the parametric trust model, the trust that an agent has in its own capabilities varies according to what it knows it can do. In this case, it can trust one agent concerning certain capabilities and not trust itself about it. Thus, if the other agent is capable of doing a specific task that the first is not, it may prefer to delegate the task to a partner. This does not mean that it will delegate the task: the other agent may not want to do it. Moreover, as the accomplish of a task usually needs more than a capability, the other agent may have some capabilities and do not have others.

Consequently, task delegation can be formalized in the following way: if an agent i needs to accomplish an action α and there is another agent j with a capability needed to accomplish it and i does not have the capability, i may delegate the task to j in some future state. This way:

$\forall i \exists j \exists c(c \in capabilities(\alpha) \wedge TrustPB(i,j,c)$
$\wedge \neg TrustPB(i,i,c) \rightarrow E\Diamond(Happens\, do(j,\alpha)))$

Below the trust predicate is related with the beliefs of an agent. In the pure binary case, an agent i will not trust agent j if it has said something contradicting i's beliefs:

$\forall i \forall j((Bel\, i\, (said\, j\, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma)$
$\wedge(\Gamma \wedge \varphi \rightarrow \neg\psi) \rightarrow \neg TrustB(i,j))$

In the parametric trust model, the agent i will not trust j concerning the capabilities needed to talk about φ:

$\forall i \forall j \forall c((Bel\, i\, (said\, j\, \varphi)) \wedge (Bel\, i\, \psi) \wedge (Bel\, i\, \Gamma)$
$\wedge(\Gamma \wedge \varphi \rightarrow \neg\psi) \wedge c \in capabilities(\varphi) \rightarrow \neg TrustPB(i,j,c))$

An agent i will also not trust an agent j if the latter does not do a task delegated by the former. It can be assumed that $(Happens\, \alpha)$ means that the action was executed adequately. Thus, if an agent does not do it the way it should have, it can be said that it executed an action $\alpha' \neq \alpha$.

Below, if an agent j never does what was asked for, the agent i will not trust it or its capabilities needed to do the requested action in a future time.

$\forall i \forall j((Bel\, i\, (do\, j\, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \rightarrow A\Diamond\neg TrustB(i,j))$
$\forall i \forall j \forall c((Bel\, i\, (do\, j\, \alpha)) \wedge \neg A\Diamond(Happens\, \alpha) \wedge c \in capabilities(\alpha)$
$\rightarrow A\Diamond\neg TrustPB(i,j,c))$.

It is important to notice that it is too strong to say that i will not trust j if it never does what was asked for. There is nothing explicit in the logic and in LORA to express a deadline for a task to be concluded. After all, it is not reasonable that i waits eternally for its solicitation to be carried on. Obviously, the action itself can include a deadline and doing the action after the deadline may mean a different task being accomplished.

In the above formulas, it was said when an agent i should not trust an agent j, but nothing was said about when i should trust its partner. Consequently, as a desired property, in some execution path of the system, i will trust j:

$\forall i \forall j(\neg TrustB(i,j) \rightarrow E\Diamond TrustB(i,j))$
$\forall i \forall j \forall c(\neg TrustPB(i,j,c) \rightarrow E\Diamond TrustPB(i,j,c))$

However, these properties not necessarily have to be present. After all, it can be the case where the agent j always has a behaviour such that it can be considered to be distrustful.

VI. Related Work

In this work, trust and beliefs are considered to be different concepts. Nevertheless, in [10], [3], trust is a belief an agent has that the other party will do what it says it will or reciprocate, given an opportunity to defect to get higher payoffs. In particular, these texts highlight the importance of a cognitive view of trust, specially for BDI agents. The context of [10] is about task delegation, in which an agent x wishes to delegate a task to y. To accomplish that, x needs to evaluate the different beliefs it has about y motivations. However, in the approach shown here, when the trust in y is analysed in terms of its motivations (or capabilities), different variables of trust are considered, since all are bets. That is, x cannot be sure about the behaviour of y in every aspect.

In the present work, an agent must ask his partners about the behaviour of other agents if it does not know nothing about

it or wants to update its trust model. This is a distributed trust model. In some practical contexts, such as at Amazon.com and at eBay, there is a centralized reputation model, in which any agent can see the reputation of other agents to decide with whom it should interact. In this kind of system, the agents consult the system itself and not their partners. Moreover, there is only a global value of reputation for each agent.

Finally, in [12], the inquire about the trustfulness of an agent is done to its organization, which is supposed to know all information about its agents and about their behaviour inside it. However, an agent can belong to more than one organization and have different behaviours in different organizations.

VII. Conclusion

In this text it was presented how to provide a trust model explicitly to a BDI agent by means of a new kind of mental state. To accomplish it, a trust notion was added to the definition of a BDI agent as a first step. Secondly, it was shown how the trust model interacts with the other components of the agent as presented in figure 1. Moreover, the entrance and exit of agents from the system and the trust of an agent in its partner were modelled in modal logic and used the possible worlds semantics. This model extends LORA [8]. To deal with messages exchanges, task delegations and the entrance and exit of agents, some specific actions were defined, which are the actions done by the agents in their environment.

Furthermore, were defined some properties relating the updating of its trust model with the updating of the beliefs of an agent, when and how an agent must update its trust model, and about trusting in itself.

As future work, it would be interesting to extend the logic to deal with sets of agents instead of single agents in the mental states modalities and trust predicates. This would reflect collective mental states of agents. In LORA, this is allowed for the natural BDI mental states. It would be interesting to extend this concept for trust, although it is not a modality. Other approach would be creating another mental state to include the messages sent and received by the agent, since the agents change a lot of information with each other, which are currently stored in the beliefs of the agent. This mental state would also be formalized as a modality, as it was done in this work.

Moreover, just as trust is a probabilistic value in the non-binary cases, the same could be done to the beliefs. That is, the agent, instead of just representing a percept in its beliefs, it would also give a probabilistic value to it, which would mean the probability that the percept is correct.

Finally, it is intended to implement a CAV (Computer Aided Validation) tool to verify open MAS models against formulas of the logic described here to see if the model has desired properties. Using this tool and the logic, some practical contexts in which the trust model described in this paper can be implemented efficient and satisfactorily are desired to be explored.

Acknowledgment

The authors would like to thank CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnologia) and PUC-Rio for the funds that made this work possible.

References

[1] H. A. Simon, *The Sciences of the Artificial*, 3rd ed. MIT Press, 1996.
[2] N. R. Jennings, "An agent-based approach for building complex software systems," *Communications of the ACM*, vol. 44, no. 4, pp. 35–41, 2001.
[3] S. D. Ramchura, T. D. Huynh, and N. R. Jennings, "Trust in multi-agent systems," *The Knowledge Engineering Review*, vol. 19, no. 1, March 2004.
[4] M. Wooldridge, "Intelligent agents," in *Multiagent Systems - A Modern Approach to Distributed Artificial Intelligence*, G. Weiss, Ed. The MIT Press, April 1999.
[5] K. Binmore, *Fun and Games: A Text on Game Theory*. D. C. Health and Company, 1992.
[6] S. Russel and P. N. Norvig, *Artificial Intelligence: A Modern Approach*. Prentice-Hall, 1995.
[7] R. Axelrod, *The Evolution of Cooperation*. New York, USA: Basic Books, 1984.
[8] M. Wooldridge, *Reasoning About Rational Agents*. Cambridge, Massachusetts: The MIT Press, 2000.
[9] G. E. Hughes and M. J. Cresswell, *A New Introduction to Modal Logic*. Routledge, 1996.
[10] C. Castelfranchi and R. Falcone, "Social trust: A cognivive approach," in *Trust and Deception in Virtual Societies*, C. Castelfranchi and Y.-H. Tan, Eds. Kluwer Academic Publishers, 2000, pp. 55 – 90.
[11] M. Burrows, M. Abadi, and R. Needham, "A logic of authentication," in *Proceedings of the Royal Society*, 1989.
[12] J. S. P. Guedes, V. T. da Silva, and C. L. P. de Lucena, "A reputation model based on testimonies," in *Proceedings of the AOIS2006@CAiSE workshop*, Luxembourg, Grand-Duchy of Luxembourg, June 2006.