An Ontology Mapping Service to Support Agent Reputation Models Interoperability

[Extended Abstract]

Luis G. Nardin[1], Anarosa A. F. Brandão[1], Jaime S. Sichman[1], Laurent Vercouter[2]
[1]Laboratório de Técnicas Inteligentes - EP/USP
Av. Prof. Luciano Gualberto, 158 - trav. 3
05508-900 - São Paulo - SP - Brazil
{luis.nardin,anarosa.brandao,jaime.sichman}@poli.usp.br

[2]Ecole Nationale Supérieure des Mines de Saint-Etienne
158, cours Fauriel, 42023
Saint-Etienne Cedex 2, France
Laurent.Vercouter@emse.fr

ABSTRACT

Interaction among agents about reputation is essential to accelerate the convergence of the reputation evaluation. Semantic heterogeneity of reputation models turns the establishment of interaction among agents into a problem especially in open MAS. We propose in this paper a service-oriented architecture to deal with this semantic heterogeneity and detail the implementation of an ontology mapping service based on a hybrid approach. This service is used to map two reputation model ontologies to a common ontology thus providing data for analysis of limitations when using the hybrid approach for heterogeneous agents interoperation.

Categories and Subject Descriptors

D.2.12 [**Interoperability**]: Data mapping; D.2.11 [**Software Architecture**]: Domain-specific architectures; I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Ontology Mapping, Reputation, Multiagent Systems

1. INTRODUCTION

In multi-agent systems (MAS), the agents have two important capabilities: to act autonomously at some extent and to engage in social activities as cooperation, coordination and negotiation [16]. The engagement in any of these social activities implies that the agent will exchange information. In open environments where there is no control about the agents that enter or leave the system, agents that participate on those social activities are exposed to risks, for example, when taking a decision based on inaccurate information received from a malevolent agent.

Some solutions to this problem are based on trust models which serve as a decision criterion for an agent to engage in social activities. Reputation has being used by most of the researchers as the preferred mechanism to implement trust models.

Reputation is a social property or a social process. It is a social property when considered as an agent's mental representation about other agents (i.e. an evaluative belief) and a social process when considered as the result of the belief's transmission (i.e. a meta-belief) [6].

Considering reputation as a social process requires that the agents in the system interact in order to get more information to accelerate the reputation evaluation convergence. However, because there is no consensus about a reputation definition, the several reputation models already proposed ([5], [17], [6], [12], [11] and [10]) were developed using different approaches and different semantics attached to the reputation concepts.

In order to overcome this semantic heterogeneity of reputation models, [4] proposed a Functional Ontology of Reputation (FORe) and used the hybrid approach [15] to demonstrate that FORe could be used as a common ontology in the reputation domain.

In this paper a service-oriented architecture for reputation interaction is proposed as well as the implementation of one of its components: the Ontology Mapping Service. This service component supports the problem of interoperability during the exchange of reputation messages between agents that implement heterogeneous reputation models by mapping the concepts from their reputation model ontologies to a common ontology.

The rest of the document is organized as follows. Section 2 presents a service-oriented architecture to support agent reputation models interoperation. The Ontology Mapping Service module implementation is presented in section 3. In section 4, the usage of the Ontology Mapping Service is presented. In addition, it is designed the reputation model ontologies of L.I.A.R. [10] and Repage [11] and their concepts are aligned in terms of FORe. The service is then used to map the L.I.A.R. and Repage reputation model ontologies to FORe. Finally, some discussion about the mapping results and comments about future works are presented in section 5.

2. ARCHITECTURE

The use of hybrid approach proposed by [15] for the interoperation of agents with heterogeneous reputation model requires the agent to perform two distinct but interrelated ontology functions, mapping and translation. Mapping is a collection of functions assigning the concepts and relations in one ontology to the concepts and relations in another ontology. Translation is the application of the mapping functions to translate sentences from an ontology to another [9].

Agent architectures that implement a reputation model do not have those ontological functions integrated. Because of this, they are not ready to deal with agents with different reputation models. Thus, for those agents to interact new architectures are required.

2.1 General Agent Architecture for Reputation Interaction

In previous work [14] and [3], a general architecture for reputation interaction was proposed considering the interoperation in an environment where agents have heterogeneous reputation models. In this architecture, a new component denominated Reputation Mapping Module (RMM) was integrated to the agent architecture in order to provide the mapping and translation ontology functions.

This general agent architecture facilitates the interoperability of heterogeneous reputation models but it has two main drawbacks from the perspective of one agent. First it combines in a single module the mapping and translation functions. There is no clear distinction between these two tasks. Then, it requires that each agent knows both an ontology of its own reputation model and the common ontology shared by every agent. Each agent should also know all the mapping functions. These are important assumptions on the agents and make them quite heavy.

2.2 Service-Oriented Architecture for Reputation Interaction

We propose a service-oriented architecture for reputation interaction (Figure 1), which extends the general agent architecture from previous work [14] and [3]. It extends the general agent architecture in two ways:

1. it subdivides the Reputation Mapping Module in two distinct and specialized modules: **Ontology Mapping Service** and **Translator**

2. it performs the ontology mapping function as a service outside the agent architecture

The benefits of these extensions are: (1) the elimination of the mapping function in all the agents, which reduces the resources required to run it; and (2) the reutilization of mapping results, since two agents may have the same internal reputation models.

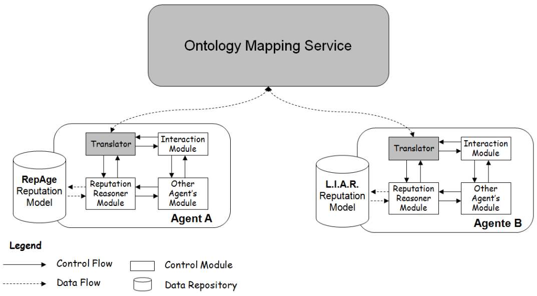

Figure 1: The proposed Service-Oriented Architecture

The **Ontology Mapping Service** module is a service and resides outside the agent. It implements the mapping ontology function and it has two main activities: (1) to map concepts from a reputation model ontology to concepts of a common ontology, which can be directly inferred by simply classifying the resulting ontology from the integration and alignment of a given reputation model and the common ontology. The integration and alignment are made by considering that both ontologies use the same common vocabulary and are described in OWL DL [1] language; and (2) to answer concept translation requests from the Translator module.

The **Translator** module resides inside the agent and it implements the translation ontology function. It has four main activities: (1) to translate the reputation messages from the common ontology to the agent's reputation model ontology whenever the message comes from the Interaction Module; (2) to translate the reputation messages from the agent's reputation model ontology to the common ontology whenever the message is sent to the Interaction Module; (3) to trigger some function in the Reputation Reasoner Module based on the interpretation of messages written using the reputation model ontology; and (4) to create a message using the reputation model ontology whenever requested by Reputation Reasoner Module. This module is out of the scope of this paper and it will be treated in future work.

In the next section the Ontology Mapping Service is detailed and its implementation considerations are described.

3. ONTOLOGY MAPPING SERVICE IMPLEMENTATION

In the context of the architecture depicted on Figure 1, the Ontology Mapping Service (Figure 2) is considered the core component since it is responsible for supporting the Translator module that resides inside each agent.

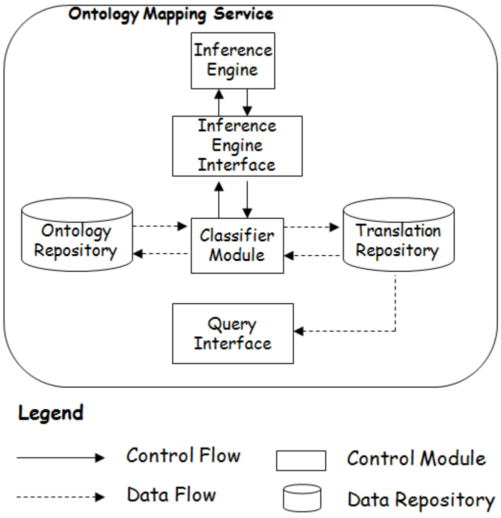

Figure 2: Ontology Mapping Service components

A brief description of each component of the service follow:

Ontology Repository stores the aligned reputation model ontologies.

Translation Repository stores the reputation models concept mapping to a common ontology concepts.

Classifier Module reads the ontologies, described in OWL DL, stored in the *Ontology Repository*; classifies it using the *Inference Engine* and stores the result in the *Translation Repository*. Its pseudo-code follows:

```
read ontology
classifyOntology(ontology)
allAsserted = getAllAssertedConcepts(ontology)
for each asserted of allAsserted do
  allInferred = getAllInferedConcepts(asserted)
  for each inferred of allInferred do
    write TranslationRepository[asserted,
                               inferred]
  end for
end for
```

Algorithm 1: Classifier Module Algorithm

Inference Engine Interface is a communication module between the *Classifier Module* and the *Inference Engine*. This interface provides flexibility to the service if for some reason there is the need of changing the *Inference Engine*.

Inference Engine is an ontology reasoner that classifies the reputation model ontology concepts and the common ontology concepts.

Query Interface answers the requests for translation of concepts. At the moment, there is only one request that can be made to this service, which returns the correspondent concept `fromConcept` from ontology `(fromOnt, fromOntVersion)` found on ontology `(toOnt, toOntVersion)`:

```
String translateConcept(String fromConcept,
String fromOnt, Integer fromOntVersion, String
toOnt, Integer toOntVersion)
```

The Ontology Mapping Service is implemented as a Web Service [2] and it is fully implemented in Java programming language using the Protégé-OWL Plugin [8]. The inference engine chosen was Pellet [13] since it (1) is completely developed in Java and (2) has a method call integration to the Protégé-OWL Plugin.

4. ONTOLOGY MAPPING SERVICE USAGE

In order to demonstrate the usage of the Ontology Mapping Service described in section 3, we applied it to map the reputation models L.I.A.R. and Repage to a common reputation domain ontology, the FORe.

The following two steps are required prior to apply the Ontology Mapping Service:

1. to design the reputation model ontologies of L.I.A.R. and Repage reputation models, since both reputation models were not described in ontological terms and the Ontology Mapping Service maps ontologies.

2. to align the L.I.A.R. and Repage ontologies to FORe, since the Ontology Mapping Service processes only ontologies that are already described in terms of a common ontology.

4.1 Designing the Reputation Model Ontologies

Ontology is an explicit specification of a conceptualization, where conceptualization is a formal representation of a knowledge domain, which is composed of objects and relations [7].

We built the reputation model ontology using OWL, which is the most recent standard ontology language from the World Wide Web Consortium (W3C)[1] . The development of the ontologies was performed using Protégé [8].

The FORe aims not only at interoperability about reputation models, but also at covering a broad knowledge about reputation. [14] identified the concepts *reputation role*, *reputation entity*, *reputation type*, *reputation evaluation value* and *reputation final value* as the useful ones for the purpose of reputation modeling mapping.

In the sequence, the terminologies identified as concepts in the L.I.A.R. and Repage reputation models and that are related to the ones identified as useful concepts for mapping are described. Those concepts compose the design of the reputation model ontologies.

L.I.A.R. (Liar Identification for Agent Reputation) is a model for the implementation of social control of agent interaction. The idea is to provide tools that allow agents to (1) reason about other agent's interaction; (2) detect any interaction rules violation; and (3) maintain a reputation model of other agents [10].

Its reputation model distinguishes reputation in five different types which are based on seven roles involved in the reputation-related processes. Each of the seven roles are defined by the source and kind of information used to calculate the reputation value. *Target role* refers to the agent that is being judged; *Participant role* is played by an agent that interacts with the target; *Observer role* involves the agents that observes the messages and transforms them into social commitment; *Evaluator role* is played by agents that evaluates the compliance of social commitments regarding some social norm; *Punisher role* refers to the agents that computes the reputation values based on evaluations; *Beneficiary role* is the agent that reason and decides based on the reputation values; and *Propagator role* is played by agents able to transmit messages about observed messages, social policies and reputation values.

The five different types of reputation are: *Direct Interaction-based Reputation* (DIbRp) which is built on messages from the target to the beneficiary; *Indirect Interaction-based Reputation* (IIbRp) which is built on messages observed by a beneficiary; *Observation Recommendation-based Reputation* (ObsRcbRp) which is built on observed messages sent from a propagator to a beneficiary; *Evaluation Recommendation-base Reputation* (EvRcbRp) which is built on evaluations sent from a propagator to a beneficiary; and *Reputation Recommendation-based Reputation* (RpRcbRp) which is built on reputation values sent from a propagator to a beneficiary.

Each reputation is associated to a facet, which is the subject the evaluation is about.

Repage. The Repage (Reputation and ImAGE) system is a computational module based on a reputation model proposed by [6]. *Image* and *reputation* are the two main concepts in this model and they represent social evaluations. *Image* is an evaluative belief which is formed using information acquired by agent experience or propagated third-party images. *Reputation* is a meta-belief which is formed based on anonymous reputation value transmitted on the social network about the target agent.

The social evaluations are context-based which means that the agent may hold different social evaluation for the same target (*AgentImage* and *AgentReputation*).

The model distinguishes the types of agents involved in the image or reputation formation in four different types. *Target* agents are individuals, groups or artifacts that are evaluated. *Evaluator* agents are entities that form an evaluative belief about other agents based on its own belief. *Propagator* agents are entities that transmit evaluations to other agents. *Beneficiary* agents make use of the evaluation and information transmitted in its own behalf.

4.2 Aligning the Reputation Model Ontologies to FORe

Alignment is the establishment of binary relations between the concepts of two ontologies [9]. The binary relations used to perform this operation in our case are defined in FORe.

We manually defined each of the reputation model concepts identified in the previous subsection in terms of FORe. For example, the L.I.A.R. instances of the *Direct Interaction-based Reputa-*

[1] www.w3c.org

tion concept have at least one association through the *hasInformationSource* relation to instances of the *DirectExperience*, formally defined:

$$\exists hasInformationSource(DirectExperience)$$

In addition, the Repage instances of the *Image* concept have at least one association through the *hasInformationSource* relation to instances of the *DirectExperience* or *Observation* or *SecondHandInformation*, formally defined:

$$\exists hasInformationSource(DirectExperience\ or\ Observation\ or\ SecondHandInformation)$$

4.3 Mapping using the Ontology Mapping Service

Having the alignment done, we manually store it in the *Ontology Repository*. Therefore, the Ontology Mapping Service detects this new ontology and executes the *Classifier Module* (Algorithm 1) to process it. As a result, the output of such process is stored in the *Translation Repository*. This is made for each reputation model ontology. Table 1 shows the use of the Ontology Mapping Service for L.I.A.R. reputation model ontology. Table 2 shows the same for Repage.

Table 1: Mapping L.I.A.R. reputation model ontology to FORe

L.I.A.R concept	FORe concept
Facet	ReputationNature
Punisher	EvalutorRole MaintainerRole
Beneficiary	RecipientRole
Target	TargetRole
Propagator	TransmitterRole
DIbRp	DirectReputation
IIbRp ObsRcbRp EvRcbRp	ObservedReputation
RpRcbRp	PropagatedReputation
ReputationValue	ReputationEvaluationValue ReputationFinalValue

Table 2: Mapping Repage reputation model ontology to FORe

Repage concept	FORe concept
AgentReputation AgentImage	ReputationNature
Evaluator	EvaluatorRole MaintainerRole
Beneficiary	RecipientRole
Target	TargetRole
Propagator	TransmitterRole
Image	ReputationType
Reputation	SecondaryReputation
Value	ReputationEvaluationValue ReputationFinalValue

Analysing the mapping results, we detect that some mappings are not one-to-one relations. This is true from the reputation model ontology to the FORe and vice-versa. For instance, in the L.I.A.R. mapping, the concept *Punisher* is mapped to the *EvaluatorRole* and *MaintainerRole* concepts of FORe, and in the Repage mapping, the concepts *AgentReputation* and *AgentImage* are mapped to the *ReputationNature* concept of FORe. This may cause some translation inconsistencies that should be treated by the Translator module which is out of the scope of this paper.

5. DISCUSSION AND FUTURE WORK

In this paper we presented an extension to the general agent architecture for reputation interaction proposed by [14]. Such extension divided the Reputation Mapping Module in two components: the Ontology Mapping Service and the Translator module. Moreover, the Ontology Mapping Service was extracted from the internal agent architecture to become an external service. Such service was detailed and its implementation was made in pure Java programming language.

In order to validate the service, two experiments were conducted: one using the L.I.A.R. reputation model ontology and other using the Repage reputation model ontology. The results analysis showed that although the mapping would be correctly performed, some problems related to translation may arise. This will be treated by the Translator module as future work and some possible suggestions are: (i) to simply generate an error related to translation inconsistency; (ii) to establish some priority level among concepts which have the same root; or (iii) to return all the possible translations.

6. ACKNOWLEDGEMENTS

Anarosa A. F. Brandão is supported by CNPq/Brazil under grant 310087/2006-6 and Jaime S. Sichman is partially supported by CNPq/Brazil and FAPESP/Brazil. Laurent Vercouter was partially supported by FAPESP/Brazil under grant 2005/02902-5.

7. REFERENCES

[1] S. Bechhofer, F. Harmelen, J. Hendler, I. Horrocks, D. L. McGuiness, P. F. Patel-Schneider, and L. A. Stein. OWL web ontology language reference. W3C recommendation, W3C, February 2004.

[2] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris, and D. Orchard. Web services architecture. W3C recommendation, W3C, February 2004.

[3] A. A. F. Brandão, L. Vercouter, S. Casare, and J. S. Sichman. Extending the art testbed to deal with heterogeneous agent reputation models. *Proceedings of Workshop on Trust in Agent Societies*, 2007.

[4] S. Casare and J. S. Sichman. Using a functional ontology of reputation to interoperate different agent reputation models. *Journal of the Brazilian Computer Society*, 11(2):79–94, 2005.

[5] C. Castelfranchi and R. Falcone. Principles of trust in mas: Cognitive anatomy, social importance and quantification. *ICMAS'98*, pages 72–79, 1998.

[6] R. Conte and M. Paolucci. *Reputation in Artificial Societies. Social Beliefs for Social Order*. Boston: Kluwer, 2002.

[7] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, June 1993.

[8] M. Horridge, H. Knublauch, A. Rector, R. Stevens, and C. Wroe. A practical guide to building owl ontologies using the protégé-owl plugin and coode tools edition 1.0. Technical report, The University Of Manchester Stanford University, 2004.

[9] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. *The Knowledge Engineering Review*, 18:1–31, 2003.

[10] G. Müller and L. Vercouter. L.i.a.r. achieving social control in open and decentralised multi-agent systems. *Journal of Autonomous Agents and Multi-Agent Systems*, 2008.

[11] J. Sabater, M. Paolucci, and R. Conte. Repage: Reputation and image among limited autonomous partners. *Journal of Artificial Societies and Social Simulation*, 9(2), 2006.

[12] J. Sabater and C. Sierra. Social regret, a reputation model based on social relations. *Journal SIGecom Exch.*, 2(1):44–56, 2001.

[13] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical owl-dl reasoner. *Journal of Web Semantics*, 2006.

[14] L. Vercouter, A. A. F. Brandão, S. Casare, and J. S. Sichman. An experience on reputation models interoperability based on a functional ontology. *Proceedings of 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, 2007.

[15] U. Visser, H. Stuckenschmidt, H. Wache, and T. Vögele. Enabling technologies for inter-operability. *Workshop on the 14th International Symposium of Computer Science for Environmental Protection*, 2000.

[16] M. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons, June 2002.

[17] G. Zacharia and P. Maes. Trust management through reputation mechanisms. *Journal of Applied Artificial Intelligence*, 14(2):881–907, 2000.