December 2006

Service Oriented Architecture for E-Commerce Negotiations: a Semantic Approach

Maricela Bravo
National Centre for Research and Technological Development (CENIDET)

Service Oriented Architecture for E-Commerce Negotiations: a Semantic Approach

MSc. Maricela Claudia Bravo Contreras
National Centre for Research and Technological Development (CENIDET)
mari_clau@cenidet.edu.mx

ABSTRACT

Traditional negotiation systems have been implemented using agent architectures, where agents communicate exchanging negotiation primitives generated by each system, based on particular language definitions implicitly encoded, giving different syntax and semantics to their messages. In this paper we address the problem of communicating heterogeneous negotiation agents in a Web-based environment, considering differences in their message implementations. Our research is based in the development of a shared ontology for publishing definitions of negotiation primitives, and a translator module, which is executed only when a misunderstanding occurs. We implemented a service-oriented architecture for executing negotiations and conducted experiments incorporating heterogeneous agents. The results of the tests show that the proposed solution improves communication between negotiation agents.

Keywords

Service oriented architecture, ontology, multi-agent systems, and electronic negotiation.

INTRODUCTION

Negotiation plays a fundamental role in electronic commerce activities, allowing participants to interact and take decisions for mutual benefit. Traditional negotiations have been implemented in small and medium-sized multi-agent systems (MAS). Recently there has been a growing interest in conducting negotiations over Internet, and constructing large-scale agent communities based on emergent Web service architectures. The challenge of integrating and deploying multiple negotiation agents in open and dynamic environments is to achieve effective communications.

The language used by agents to exchange messages is defined as agent communication language (ACL). An ACL allows an agent to share information and knowledge with other agents, or request the execution of a task. KQML was the first standardized ACL from the ARPA knowledge project. KQML consists of a set of communication primitives aiming to support interaction between agents. KQML includes many performatives of speech acts. Another ACL standard comes from the Foundation for Intelligent Physical Agents (FIPA) initiative. FIPA ACL is also based on speech act theory, and the messages generated are considered as communicative acts. The objective of using a standard ACL is to achieve effective communication without misunderstandings, but this is not always true. Because, standards specify the semantics of communicative acts, but the software implementation is not explicitly defined, leaving developers to follow their own criteria. Furthermore, standard ACL specifications consider the incorporation of privately developed communicative acts.

In this thesis we address the problem of language heterogeneity that occurs during the exchange of negotiation messages between agents, considering that agents may use the same ACL but messages generated by each agent may have different syntax and/or meaning not based on explicit semantics, but on particular definitions implicitly encoded. We have selected a translation approach solution based on the incorporation of a shared ontology. We have implemented the ontology explicitly describing negotiation messages in a machine interpretable form. The ontology represents the shared vocabulary that the translator uses during execution of negotiation processes for solving misunderstandings.

The rest of the document is organized as follows. We first present the related work concerning this research topic. Then we describe the semantic translator architecture. Then we present the description of the shared ontology. Then we present the prototype architecture for executing negotiation processes, and describe the results of experiments. Finally, we present conclusions.

RELATED WORK

According to Jürgen Müller (Müller, 1996), research in negotiation is organized in three classes: language, decision and process. We have concentrated our work in the language aspect of negotiation; in particular we are interested in analyzing research concerning the communication language interoperability between agents. In the revised works we have identified two trends in communications between negotiation agents, one is the generalized idea of using a standard, and the other is to provide mechanisms for solving heterogeneity. In particular, in this work we deal with the second trend. In this section we present the related work within this context.

Malucelli (Malucelli and Oliveira, 2004) stated that a critical factor for the efficiency of negotiation processes and the success of potential settlements is an agreement between negotiation parties about how the issues of a negotiation are represented and what this representation means to each of the negotiation parties. In (Pokraev, Reichert, Steen and Wieringa, 2005) authors explain that interoperability is about effective use of systems´ services. They argue that the most important precondition to achieve interoperability is to ensure that the message sender and receiver share the same understanding of the data in the message and the same expectation of the effect of the message. Sonia Rueda (Rueda, 2002) argues that the success of an agent application depends on the communication language, allowing agents to interact and share knowledge. Pokraev (Pokraev, Zlatev and Brussee, 2004) discussed the problems of automating the process of negotiation. In this work he argues that there is a problem of lack of common understanding between participants in a negotiation, because messages are created by different actors and different meaning is given to the concepts used in them. In (Haifei, Chunbo and Stanley, 2002) authors explain that there are two important aspects of a negotiation process: communication between negotiation parties and decision-making. They state that communication deals with how to represent negotiator's requirements and constraints on a product and service and how to convey intentions by passing messages between negotiation parties. The lack of common language implementations represents a problem during the exchange of messages between heterogeneous systems, and this lack of standardization is known as interoperability problem (Willmott, Constantinescu and Calisti, 2001).

In the above related works we can see that there is a common concern in communications in agent communities. Authors present the problem of lack of common understanding or the need for clarifying the meaning of concepts. They agree on using a standard language to overcome heterogeneity, but implementations of such languages are too far from standards, besides this solution lacks of flexibility to facilitate the automatic integration of multiple negotiation systems. In contrast in this thesis we adopt a different approach, using a shared ontology and reducing the requirement of redesigning and adopting a common language. In this paper we present our implemented solution and experimental cases showing our contribution.

TRANSLATION ARCHITECTURE

The translator acts as an interpreter of different negotiation agents. In figure 1, we present the architectural elements involved in translation. For example, suppose that agents A and B initiate a negotiation process, using their own local ACL, sending messages over the message transport. If happens that agent A misunderstands a message from agent B, it invokes the translator module sending the message parameters (sender, receiver, message). The translator interprets the message based on the definitions of the sender agent and converts the message into an interlingua. Then the translator converts the interlingua representation to the target ACL based on the receiver agent definitions. Finally, the translator sends the message back to the invoking agent A and continues with execution of negotiation. The translator is invoked only in the occurrence of a misunderstanding, assuring interoperability at run time.

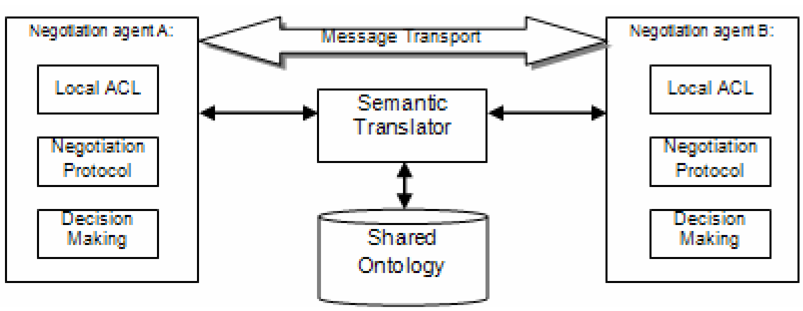

Figure 1. Semantic translator architecture

SHARED ONTOLOGY

Ontologies have been studied in various research communities, such as knowledge engineering, natural language processing, information systems integration and knowledge management. Ontologies are a good solution for facilitating shared understanding between heterogeneous information systems.

The principal objective of the ontology is to serve as a shared vocabulary of negotiation primitives, where all agent developers describe the primitives that their agents use for communication. In the ontology, primitives are organized following the classification proposed by Müller (Müller, 1996), this classification establishes that negotiation messages are divided into three groups: initiators, if they initiate a negotiation, reactors, if they react on a given statement and completers, if they complete a negotiation. Another important feature of ontologies is relations. Relations are useful for linking instances or individuals in an ontology, in this case we defined two types of relations: similarity and synonymy. Figure 2 shows the general structure of our ontology.

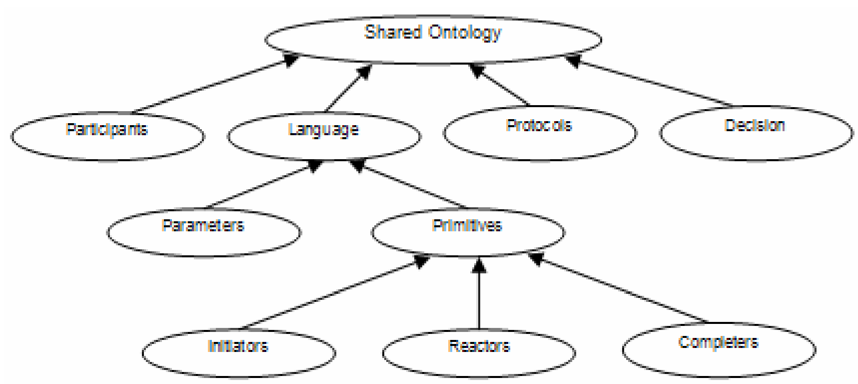

Figure 2. General structure of the negotiation ontology

Based on the concepts and negotiation primitives described above we built our ontology. To code the ontology we decided to use OWL as the ontological language, because it is the most recent development in standard ontology languages from the World Wide Web Consortium (W3C). An OWL ontology consists of classes, properties and individuals. We developed the ontology using Protégé (Gennari, 2003) and (Knublauch, 2003), an open platform for ontology modeling and knowledge acquisition. Protégé has an OWL Plugin, which can be used to edit OWL ontologies, to access description logic reasoners, and to acquire instances of semantic markup. Figure 2 shows part of the ontology code generated with of Protégé.

PROTOTYPE ARCHITECTURE

We constructed a Web service-based prototype, which in turn lets deployed agents to interoperate and execute negotiations. Web service protocols and standards are a good alternative for implementing an electronic marketplace system, because interactions between participants are often dynamic and ad-hoc rather than static and planned. Therefore, dynamic binding is preferable than design binding. Web services are built on existing and emerging standards such as HTTP, Extensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web Service Description Language (WSDL) and Universal Description, Discovery and Integration (UDDI). In this section we briefly describe the functionality and implementation techniques for each component.

```
<owl:Class rdf:ID="Participants">
  <rdfs:subClassOf rdf:resource="#negotiation"/>
</owl:Class>
<owl:Class rdf:ID="Language">
  <rdfs:subClassOf rdf:resource="#negotiation"/>
</owl:Class>
<owl:Class rdf:ID="Protocol">
  <rdfs:subClassOf rdf:resource="#negotiation"/>
</owl:Class>
<owl:Class rdf:about="#Primitives">
 <rdfs:subClassOf rdf:resource="#Language"/>
</owl:Class>
<owl:Class rdf:about="#Parameters">
 <rdfs:subClassOf rdf:resource="#Language"/>
</owl:Class>
<owl:Class rdf:ID="Initiator">
  <rdfs:subClassOf rdf:resource="#Primitives"/>
</owl:Class>
<owl:Class rdf:ID="Reactor">
  <rdfs:subClassOf rdf:resource="#Primitives"/>
</owl:Class>
<owl:Class rdf:about="#Completer">
  <rdfs:subClassOf rdf:resource="#Primitives"/>
</owl:Class>

<owl:ObjectProperty rdf:ID="isSuccessorOf">
  <owl:inverseOf rdf:resource="#hasSuccesor"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#hasSuccesor">
  <owl:inverseOf rdf:resource="#isSuccessorOf"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="isSynonymOf">
  <owl:inverseOf rdf:resource="#hasSynonym"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#hasSynonym">
  <owl:inverseOf rdf:resource="#isSynonymOf"/>
 </owl:ObjectProperty>
```

Figure 3. Part of the ontology code generated with Protégé

(1). *The Matchmaker* is a Java module which is continuously browsing buyer registries and seller descriptions, searching for coincidences.

(2). *Negotiation process* is a BPEL4WS-based engine that controls the execution of negotiation processes between multiple agents according to the predefined protocols. BPEL4WS provides a language for the formal specification of business processes and business interaction protocols. The interaction with each partner occurs through Web service interfaces, and the structure of the relationship at the interface level is encapsulated in what is called a partner link.

(3). *Seller and buyer agents* are software entities used by their respective owners to establish their preferences and negotiation strategies. For example, a seller agent will be programmed to maximize his profit, establishing the lowest acceptable price and the desired price for selling. In contrast, a buyer agent is seeking to minimize his payment. On designing the negotiation agents, we identified three core elements, strategies, the set of messages and the protocol for executing the negotiation process. The requirements for these elements were specified as follows:

a. Strategies should be private to each agent, because they are competing and they should not show their intentions.

b. Messages should be generated privately.

c. The negotiation protocol should be public or shared by all agents participating, in order to have the same set of rules for interaction.

(4). *The translator module* is invoked whenever an agent misunderstands a negotiation message from another agent. The translator module was implemented using Jena (Jena), a framework for building Semantic Web applications. It provides a programmatic environment for OWL, including a rule-based inference engine. For example, suppose that agents A and B initiate a negotiation process, using their own local negotiation primitives, sending messages over the message transport. In case that agent A misunderstands a primitive from agent B, it invokes the translator module sending the required parameters: sender, receiver and source primitive; in this example the sender agent is B, and the receiver agent is A. The translator reads the input parameters and identifies the class of the source primitive in the ontology based on the definitions of agent B; then searches in the classification to find if there is a target primitive from agent A holding a similarity relation with the source primitive. If the translator finds such a primitive sends the primitive to the invoking agent A and continue with execution of negotiation.

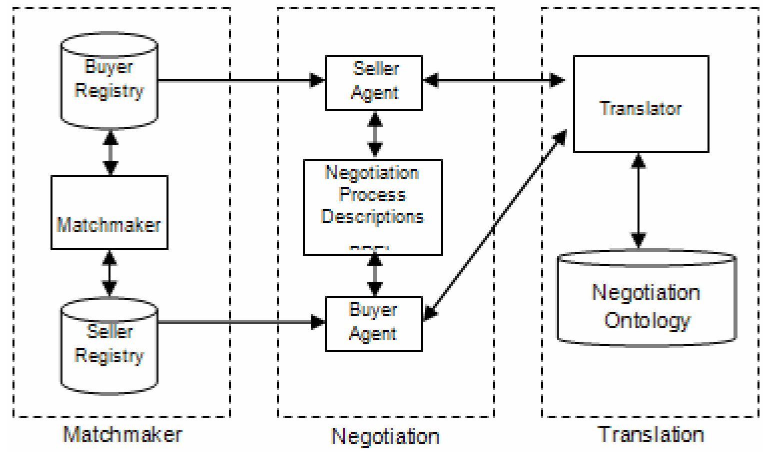

Figure 4. Architecture of the prototype

EXPERIMENTAL RESULTS

The negotiation experiments were executed in two phases. The first execution tested the interaction between agents, incorporating messages with different syntax, without the semantic translator. For the second execution we used the same scenario, but enabled the semantic translator module. Table 1 shows the results.

Last price	Max pay	Rounds	Qty	Final price	1st execution	2nd execution
$ 1,750.00	$ 849.00	12	847	$ -	no offer	no offer
$ 774.00	$ 1,760.00	3	887	$ 1,674.00	offer accepted	offer accepted
$ 1,788.00	$ 128.00	12	1660	$ -	no offer	no offer
$ 1,058.00	$ 110.00	12	1270	$ -	no offer	no offer
$ 694.00	$ 938.00	10	950	$ 894.00	offer accepted	offer accepted
$ 761.00	$ 77.00	12	1475	$ -	no offer	no offer
$ 1,940.00	$ 2,233.00	10	570	$ 2,140.00	offer accepted	offer accepted
$ 621.00	$ 446.00	12	56	$ -	no offer	no offer
$ 1,008.00	$ 1,235.00	10	30	$ 1,208.00	offer accepted	offer accepted
$ 114.00	$ 704.00	7	8	$ 614.00	offer accepted	offer accepted
$ 1,837.00	$ 2,199.00	9	53	$ 2,137.00	offer accepted	offer accepted
$ 1,665.00	$ 2,047.00	9	56	$ 1,965.00	offer accepted	offer accepted
$ 1,377.00	$ 1,783.00	8	31	$ 1,777.00	offer accepted	offer accepted
$ 1,920.00	$ 286.00	12	81	$ -	no offer	no offer
$ 172.00	$ 1,553.00	2	41	$ 1,172.00	offer accepted	offer accepted
$ 980.00	$ 1,541.00	2	67	$ -	not understood	offer accepted
$ 1,826.00	$ 2,464.00	2	99	$ -	not understood	offer accepted
$ 1,276.00	$ 500.00	2	43	$ -	not understood	no offer
$ 1,500.00	$ 1,108.00	2	110	$ -	not understood	no offer
$ 1,400.00	$ 1,520.00	3	4	$ -	not understood	offer accepted

Table 1. Negotiation results

The first execution results showed that there were some negotiations that ended the process with no agreement. This was due to the private strategies defined inside the agents. But there were some negotiation processes that ended due to lack of understanding of negotiation messages.

The second phase results showed a reduction in the number of negotiations finished by lack of understanding, which does not mean that the incorporation of a semantic translator module will ensure an agreement; but at least, the negotiation process will continue executing.

CONCLUSIONS

In this abstract we have presented the preliminary results of the service-oriented architecture designed to execute electronic negotiations, to address the problem of language heterogeneity. We have implemented an ontology solution to explicitly describe negotiation messages in a machine interpretable form, which the translator uses during execution of negotiation processes for solving misunderstandings at run time. We evaluated the architecture into which the negotiation processes are executed. We believe that language interoperability between negotiation agents is an important issue that can be solved by incorporating a shared ontology in a service-oriented architecture. The experimental tests showed that the proposed architecture improves the continuity of the execution of negotiation processes, resulting in more agreements.

REFERENCES

1. Finning T., Fritzon R., and McEntire R. (1994) KQML as an agent communication language, in *Proceedings of the 3rd International Conference on Information and Knowledge Management.*
2. FIPA – Foundation for Intelligent Physical Agents (2003) FIPA Specifications, available at http://www.fipa.org/specifications/index.html.
3. Uschold, M. and King M.(1995) Towards a Methodology for Building Ontologies, *Workshop on Basic Ontological Issues in Knowledge Sharing.*
4. Grüninger, M. and Fox, M. (1994) The Role of Competency Questions in Enterprise Engineering, *IFIP WG 5.7 Workshop on Benchmarking*. Theory and Practice, Trondheim, Norway.
5. Müller, H. J. (1996) Negotiation Principles, *Foundations of Distributed Artificial Intelligence*, in G.M.P. O´Hare, and N.R. Jennings, New York: John Wiley & Sons.
6. Haifei Li, Chunbo Huang and Stanley Y.W Su (2002) Design and Implementation of Business Objects for Automated Business Negotiations, *Group Decision and Negotiation*, Vol. 11; Part 1, pp. 23-44.
7. Gennari J., Musen M., Fergerson R., Grosso W., Crubézy M., Eriksson H., Noy N., and Tu S. (2003) The evolution of Protégé-2000: An environment for knowledge-based systems development, *International Journal of Human-Computer Studies*, 58(1): 89-123.
8. Knublauch H. (2003) An AI tool for the real world: Knowledge modeling with Protégé, *JavaWorld*, June 20.
9. Malucelli A., and Oliveira E. (2004) Towards to Similarity Identification to help in the Agents' Negotiation, *Proceedings of 17th Brazilian Symposium on Artificial Intelligence*, São Luis, Maranhão, Brazil.
10. Pokraev S., Reichert M., Steen M. and Wieringa R. (2005) Semantic and Pragmatic Interoperability: A Model for Understanding, *Proceedings of the Open Interoperability Workshop on Enterprise Modelling and Ontologies for Interoperability*, Porto, Portugal, 2005.
11. Rueda S., García A., Simari G. (2002) Argument-based Negotiation among BDI Agents, *Computer Science & Technology*.
12. Pokraev S., Zlatev Z., Brussee R., van Eck P. (2004) Semantic Support for Automated Negotiation with Alliances, *Proceedings of the 6th International Conference on Enterprise Information Systems*, Porto, Portugal.
13. Willmott S., Constantinescu I., Calisti M. (2001) Multilingual Agents: Ontologies, Languages and Abstractions, In *Proceedings of the Workshop on Ontologies in Agent Systems, Fifth International Conference on Autonomous Agents*, Montreal, Canada.
14. W3C, available at http://www.w3.org
15. Jena, http://jena.sourceforge.net