E-Commerce Oriented Negotiating Agent Communication Model

Mukun Cao
School of Management
Xiamen University
caomukun@yahoo.cn

Yuqiang Feng
School of Management
Harbin Institute of Technology
fengyq@hit.edu.cn

Zhenyu Liu
School of Management
Xiamen University
zhenyliu@xmu.edu.cn

Abstract

Automated negotiation has become the core of the intelligent e-commerce. Traditional research in automated negotiation is focused on negotiation protocol and strategy. However, current research is lack of unified technology standard, which causes the system’s practical application difficult. This paper designs a negotiating agent architecture, which is based on the agent’s ability of communication, and can support both goal-directed reasoning and reactive response. In order to construct a general interaction mechanism among negotiating agents, a communication model is proposed, in which the negotiation language used by agents is defined. Design of the communication model and the language has been attempted in such a way so as to provide general support for a wide variety of commercial negotiation circumstances, and therefore to be particularly suitable for electronic commerce. Finally, the design and expression of the negotiation ontology is discussed.

1. Introduction

Electronic commerce is affecting business profoundly. It is changing the way businesses interact with consumers, as well as the way the consumers interact with each other. No matter whether it is a case of B to B purchase or a case of online shopping, it becomes more and more important to make the traditional negotiation price mechanism automated and intelligent [1]. Automated negotiation will become the core of the next generation intelligent e-commerce.

Research in automated negotiation to date has been focused on the development of negotiation protocols and strategies [2]. For example, Jennings proposed that automated negotiation research can be considered to deal with three broad topics, they are negotiation protocols, negotiation objects and agents decision making models [3]. Negotiation Protocols is the set of rules that govern the interaction. This covers the permissible types of participants, the negotiation states, the events that cause negotiation states to change and the valid actions of the participants in particular states. Negotiation Objects is the range of issues over which agreement must be reached. At one extreme, the object may contain a single issue, while on the other hand it may cover hundreds of issues. Agents’ Decision Making Models is the decision making apparatus the participants employ to act in line with the negotiation protocol in order to achieve their objectives. [3]

Although there are many research achievements about protocols and strategies in the field of automated negotiation nowadays, realization and real application of automated negotiation system still has a long way to go [4, 5, 6]. Up to now, most work on automated negotiation still in experimental stage. One important reason for appearing this kind of situation is lack of unified technology standard, which make the sharing and reuse of software very difficult. Take the protocol as an example; almost all the protocols proposed in former literatures are represented by researchers’ own way [7]. It is easy to understand that, in an open environment like e-commerce, it is difficult for negotiating agents using different interaction protocol to negotiate with each other automatically, just because they have not common and general message exchange format.

In order to solve the above problem, some organizations proposed some solutions. At the moment, there are two main technical specifications suit the standardization of the agent communication format. One is Knowledge Query and Manipulation Language (KQML) [8]; another is the Foundation for Intelligent Physical Agents (FIPA) specification [9]. Our work is based on KQML, simply because of its technical maturity and current lead in market share. However, KQML has some limitations need to be improved, when it is used in the field of automated negotiation. One is that the performatives of KQML is not enough for negotiation. KQML only has a “tell” speech act to support negotiation. But the actual situation of negotiation is more complex. Negotiation need not only “tell”, but also “accept”, “reject”, “argue” and other speech acts to describe negotiation situation comprehensively and accurately. Another is that it still suffers from poorly defined semantics. As a result, each of the many
KQML implementations seems unique. This makes it difficult to communicate with other agents from heterogeneous systems.

The main aim of the work is to find a way to construct an understandable interaction mechanism for agents in an automated negotiation process. Concretely, we want the communication model can achieve the following goals. First, negotiating agents should communicate with each other, and understand correctly what others say. Second, the negotiating agents should share common concepts about the objects they are negotiating over. These problems are basic and essential for realizing an applicable automated negotiation system.

In order to achieve the first goal, we propose a communication model which is mainly composed of a negotiation communication language. One hand, negotiation is definitely a linguistic form, and the automated negotiation particularly depends on the interaction or communication between agents in open environments such as the Internet or the Semantic Web [10]. On the other hand, automated negotiation system is in essence a multi-agent system, whose run is heavily dependent on the communication between agents. This part of work needs to solve following problems: What kinds of language do the negotiating agents use when they are in a commercial negotiation process? How to express the negotiation language correctly and formally?

In order to achieve the second goal, we propose negotiation ontology. The ontology is a way of categorizing objects such that they are semantically meaningful to a software agent. For no one can ignore the fact that agents communicating in a common language will still be unable to understand each other if they use different vocabularies for representing shared domain concepts. Ontology is required to ensure that the agents are referring to exactly the same good. Through using negotiation ontology, an agent can understand implications of all variables delivered from other agents. As a result, how to design and express the negotiation ontology is another important problem which must be solved before the negotiation interaction between agents is built.

In order to provide a running platform for the above communication model, it is necessary to construct negotiating agent. Generally speaking, the research of negotiating agent includes negotiating agent’s theoretical model, architecture and the method of communication between two or more negotiating agents. Recent theoretical work about agent has clarified the role of goals, intentions, and commitment in constraining the reasoning that an agent performs [11] [12], and has classified agent architecture into reactive system, real-time reasoning system and hybrid system. The hybrid agent has features in common with both reactive agent and real-time reasoning agent. There has been some work in the design of agent architecture that attempts to integrate goal directed reasoning and reactive behavior. For example, the PRS system [13] supports both the goal directed reasoning and the ability to react rapidly to the unanticipated changes in the environment. It interacts with the environment through four mechanisms: sensors and monitor (which are in charge of perception from the environment), effectors and command generator (which are used for acting on the environment) [14]. However, although PRS has some features in common with the architecture described in this paper, it is a common architecture for solving general problems in the field of reasoning systems and seems to be too complex for the applications of negotiation on-line. Negotiating agent is a special kind of agent; its architecture must present its specific features. We propose negotiating agent architecture to satisfy the requirement.

Ultimate aim of researching automated negotiation is to realize a real and practicable automated negotiation system. Taking advantage of existing software agent technology is a good way for developing an automated negotiation system. Moreover, most agent development platform is based on the agent communication language. As a result, the work presented in this paper is helpful for realizing a practical automated negotiation system.

The remainder of this paper is organized as follows. Section 2 describes the negotiating agent architecture. Section 3 describes the frame of the communication model for automated negotiation. Then, Section 4 discusses the negotiation language used by the agents during the negotiation process. Section 5 presents the design and expression of negotiation ontology which is an important component in the negotiation language. Finally, Section 6 draws conclusions and presents future work.

2. Negotiating Agent Architecture

Negotiating Agent Architecture (NAA) is designed for describing internal structure of negotiating agent. NAA is a kind of hybrid agent architecture (Figure 1). That means NAA has features coming from both BDI deliberative agent, which is based on goal directed reasoning, and reactive agent, which is controlled by reactive behavior. Because the negotiation is a kind of linguistic form, the NAA, different to traditional agent architecture, just needs a communication mechanism and a language generator for interaction with other agents. They are NAA’s communicator and speech-act planner. In addition, NAA has a reactive filter for the
purpose of increasing the system’s capacity for reactivity. Figure 1 shows the negotiating agent architecture.

**Belief Base** is a container for the current beliefs of the agent. It contains four kinds of beliefs. They are:

- **Static beliefs**: contains static integrant properties for negotiating application making decision. They are concretely negotiation strategy, decision model, and so on. For these decision apparatus has been established when a negotiating agent is instantiated, we call them static.

  It is necessary to give a detailed explanation for negotiation strategy. It is a kind of knowledge about how to accomplish given initial beliefs or react to certain bids from other agents, and is presented by declarative procedure specifications. Each strategy consists of a **body**, which describes the algorithm of the strategy, and a **condition** that specifies under what situations the strategy is applicable. Together, the condition and body express a declarative fact about the results and utility of performing certain negotiation strategies under certain conditions.

- **Beliefs triggered by interaction**: is received from the environment (or other negotiating agent, for example) during the process of negotiation.

- **Run time beliefs**: records beliefs acquired when the agent executes its reasoning. It may change over time.

We prescribe the belief knowledge is expressed as a triple form (object, attributes, value). For example, price of a computer is 1000$, and it can be expressed as (computer, price, 1000).

- **Desires** are expressed as conditions over some interval of time, and are described by applying various temporal operators to state descriptions. This allows representation of a wide variety of goals, including goals for achieving maximum price, goals for shorter bargaining time and so on. A given speech-action or sequence of speech-actions, is said to be appropriate for achieving a given goal, if its theoretical execution results satisfy the goal description.

Initial beliefs set by users: typically are conclusions and basic description about the current negotiation, mainly set by users when a negotiating agent is instantiated.

Beliefs triggered by interaction: is received from the environment (or other negotiating agent, for example) during the process of negotiation.

Run time beliefs: records beliefs acquired when the agent executes its reasoning. It may change over time.

We prescribe the belief knowledge is expressed as a triple form (object, attributes, value). For example, price of a computer is 1000$, and it can be expressed as (computer, price, 1000).

Desires are expressed as conditions over some interval of time, and are described by applying various temporal operators to state descriptions. This allows representation of a wide variety of goals, including goals for achieving maximum price, goals for shorter bargaining time and so on. A given speech-action or sequence of speech-actions, is said to be appropriate for achieving a given goal, if its theoretical execution results satisfy the goal description.

**Intention structure** is a data structure organizing all those desires that the agent has chosen for execution, either immediately or at some later time. These adopted desires are called intentions. The set of intentions comprising the intention structure form a partial ordering. An intention earlier in the ordering must be either realized or dropped (and thus disappear from the intention structure) before intentions appearing later in the ordering can be executed.

**Reasoner** runs the entire system. From a conceptual standpoint, it operates in a relatively simple way. At any particular time, when certain negotiation strategies are active in the system and certain beliefs are held in the belief base, then a subset of desires in the system will be invoked. One or more of these applicable desires will then be chosen for execution and thus will be placed on the intention structure.

**Speech-Act Planner** is a language generator in NAA. No one can ignore the fact that the agent must select what it should say based on the relevance of the speech-act's expected outcome or rational effect of its intentions. However, it cannot assume that the rational effect will necessarily result from sending random messages. The question then becomes which
message or set of messages should be sent to another agent to assist or cause intention to be satisfied? If the agent is behaving in some reasonable sense, it will not send out a message whose effect will not satisfy the intention and hence achieve the desire. *Speech-Act Planner* is just such a mechanism, which can select appropriate *performatives* and form KQML messages according to the prospective rational effect of a certain intention. In other words, with the aid of the *speech-act planner*, the *communicator* has something to say.

*Communicator* is in charge of the agent’s interaction with the environment, including other agents. It has the ability to process Agent Communication Language (ACL). It receives KQML messages about negotiation from the environment, and then parses them to get useful information for the agent to process. Finally, it sends KQML messages back to the environment.

*Reactive Filter* is a reactive mechanism, whose purpose is to provide an agent with fast, reactive capabilities for coping with events that are unnecessary or difficult for the reasoning mechanism to process. A typical event, for example, would be the wrong KQML message received by the *communicator*, which is always difficult to process by the *reasoner*. The reactive filter provides the agent with a series of situation-reaction rules for processing wrong messages, and for preventing other unpredictable situations. When a given rule is activated, an appropriate action is sent to the agent’s communicator, which will send a responsive KQML message very quickly and directly to the environment. So, this mechanism guarantees a certain degree of reactivity.

### 3. Negotiating Agent Communication Model

Negotiating Agent Communication Model (NACM) is designed for solving three problems. First is entity which possibly participates in the negotiation process. They are *Buyer Agents, Seller Agents* and *Facilitator* in the model. Second is communication language possibly used by the agents. Third is the logical relation between each language. However, NACM shouldn't be regarded as an agent interaction protocol, for there are many mature protocols which can be used nowadays. The main aim is to focus on an explicit definition of the language which the agents can use in any negotiation protocol.

In order to describe what the agents should say in different circumstances, we define two kinds of meta-languages: negotiation and dialog speech act, and network service speech act. The negotiations and dialog speech act is an expansion to the performatives defined in KQML, which contains *tell, propose, argue, accept, reject, ask-if, error*. The network service speech act inherits from KQML, which contains *register, unregister, advertise, subscribe, recruit-one, recruit-all, forward*. Table 1 shows details of these speech acts and their concrete meaning. Where *S* indicates sender agent, *R* indicates receiver agent, and *F* indicates facilitator (communication server).

**Table 1. Automated negotiation’s speech acts**

<table>
<thead>
<tr>
<th>Classification</th>
<th>Speech Act</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Negotiation</td>
<td>Tell</td>
<td>S let R know some of his own knowledge in his own belief base</td>
</tr>
<tr>
<td></td>
<td>propose</td>
<td>S let R know his proposal about a certain round of negotiation</td>
</tr>
<tr>
<td></td>
<td>argue</td>
<td>S let R know the reason for proposing the proposal in a certain round of negotiation</td>
</tr>
<tr>
<td></td>
<td>accept</td>
<td>S let R know he accept R’s last proposal</td>
</tr>
<tr>
<td></td>
<td>reject</td>
<td>S let R know he reject R’s last proposal</td>
</tr>
<tr>
<td></td>
<td>ask-if</td>
<td>S wants to know whether some knowledge is in the R’s belief base</td>
</tr>
<tr>
<td></td>
<td>error</td>
<td>S tell R he cannot understand the last message sent by R</td>
</tr>
<tr>
<td>Network Service</td>
<td>register</td>
<td>S send registration request to F</td>
</tr>
<tr>
<td></td>
<td>unregister</td>
<td>S send logout request to F</td>
</tr>
<tr>
<td></td>
<td>advertise</td>
<td>S declares to R he is fit for handling some affair</td>
</tr>
<tr>
<td></td>
<td>subscribe</td>
<td>S ask R to inform him some affair in future</td>
</tr>
<tr>
<td></td>
<td>recruit-one</td>
<td>S ask F to send a message to a negotiating agent who can handle this message, and is responsible for handling this message</td>
</tr>
<tr>
<td></td>
<td>recruit-all</td>
<td>S ask F to send a message to all negotiating agents who can handle this message, and are responsible for handling this message</td>
</tr>
<tr>
<td></td>
<td>forward</td>
<td>S asks R to forward some message to a negotiating agent which is indicated by the grammar item “ to”</td>
</tr>
</tbody>
</table>

KQML is a protocol for carrying and formatting messages based on speech acts theory. It integrates a concept of *facilitator* [15], which refers to a class of agents who traffic in meta-knowledge about other agents, and provide communication services such as message forwarding and broadcasting, resource discovery, matchmaking and so on. Therefore, it is the main tool for users to construct their own interaction model. Figure 2 is the communication model based on the expanded KQML.
A bargaining based on multi-agent automated negotiation must experience three stages.

First, Buyer Agents and Seller Agents register on the Facilitator (using performatives register or unregister), which is set up by a third party, provide some basic private information, such as name, host’s IP and port number; the Facilitator assigns an ID to them at a later stage.

As an important step, the Seller Agents will advertise some product information (using performative advertise(ask-if), as showed in fig2) to the facilitator who will help them forward the advertisement to Buyer Agents (using performative forward(ask-if) or tell).

Then, when a Buyer Agent wants something, it will query information from the Facilitator, using performatives recruit-one or recruit-all(ask-if).

If the Facilitator is ignorant of the product, the Buyer Agent will send a subscription to the Facilitator for further information, using performatives subscribe(recruit-one) or subscribe(recruit-all(ask-if)).

When the required information is available, the Facilitator will forward it to the claimer. Finally, a communication between the Buyer and Seller is constructed with the help of the Facilitator, and then direct negotiation can be implemented. Here, we extend the original KQML performative, using propose, argue, reject, accept, tell to satisfy the complicated linguistic circumstances occurred in a process of negotiation.

4. Negotiating Agent Communication Language

Automated negotiation relies on the idea that agents must use a shared format of communication in order to interact smoothly. In contemporary multi-agent negotiation systems, different formats of communication have been used in different systems. However, in order to fully exploit the potential of open environments like the Internet, agents should use a united negotiating language which is most suitable to any type of negotiation in which they participate. In fact, there is currently no standard widely accepted specification for the language. Negotiating Agent Communication Language (NAACL) aims to find an approach that permits agents to negotiate with most of the negotiation mechanisms. It is a syntax specification for exchanging message between agents. Its user is negotiating Agent. Using NAACL, negotiating agents can make synchronous and asynchronous communication with each other through TCP/IP and HTTP.

In order to take advantage of existing resources, the syntax of NAACL is compatible with KQML. For NAACL is a kind of Agent Communication Language based on message, it defines the formal format of message, which contains, similar to KQML, three layers: communication layer, message layer and content layer.

Communication layer: describes a group of communication parameter related to message communication. The all technological communication parameter is specified in the communication layer. The item “: sender” and “: receive” denote respectively the sender and expectant receiver of message.

Message layer: is the core of NAACL. It specifies the speech-acts related to message. Every message has at least one speech act performative. In addition, message layer provides description of attributes related to content layer, for example, the language for expressing content, ontology and so on. The item “: language” indicates the language used for expressing the content of message. Here prescribe the knowledge expression language used in NAACL can be KIF (Knowledge Interchange Format), Prolog, Lisp, XML and so on. Item “: ontology” indicates ontology used in message.
Content layer: specifies the content of message using item :: content”. The content is expressed by knowledge expression language which is defined by item :: language” in message layer.

In terms of interaction scene described in the communication model mentioned above, NACL can be divided into 10 meta-languages in detail. To do so, NACL is defined formally as the meta-negotiation language according to the KQML language specification, and expressed in BNF style as follows:

(1)<registration>::=(register | unregister
:sender<Agent_name>
:receiver<Facilitator_name>
:language KIF | Prolog | Lisp | XML
:ontology kqml-ontology
:content<agent_information>)

(2)<advertisement>::=(advertise
:sender<SellerAgent_name>
:receiver<Facilitator_name>
:language KQML
:ontology kqml-ontology
:content(ask-if
:sender<facilitator_name>
:receiver<SellerAgent_name>
:language KIF | Prolog | Lisp | XML
:ontology negotiation-ontology
:content<sale_information>)

(3)<query>::=(recruit-one | recruit-all
:sender<BuyerAgent_name>
:receiver<Facilitator_name>
:language KQML
:ontology kqml-ontology
:content<query_<content>)

(4)<subscription>::=(subscribe
:sender<BuyerAgent_name>
:receiver<Facilitator_name>
:language KQML
:ontology kqml-ontology
:content(recruit-one | recruit-all
:sender<BuyerAgent_name>
:receiver<Facilitator_name>
:language KIF | Prolog | Lisp | XML
:ontology negotiation-ontology
:content<query_<content>)

(5)<transmission>::=(forward
:from<BuyerAgent_name>
:to<SellerAgent_name>
:sender<Facilitator_name>
:receiver<SellerAgent_name>
:language KQML
:ontology negotiation-ontology
:content<product_information>)

Although the sender of <transmission> is facilitator, and receiver is seller agent, the factual meaning it expresses is transmitting content in “ask-if” (<from>) buyer Agent (<to>) seller Agent.

(6)<notification>::=(tell
:sender<Agent_name> | <Facilitator_name>
:receiver<Agent_name> | <Facilitator_name>
:language KIF | Prolog | Lisp | XML
:ontology negotiation-ontology
:content<notification_content>)

<notification> is standard reply for <query>. At the same time, it can be used for daily matter-of-fact notification.

(7)<negotiation>::=(propose | argue
:sender<Agent_name>
:receiver<Agent_name>
:language KIF | Prolog | Lisp | XML
:ontology negotiation-ontology
:content<negotiation_content>)

(8)<acception>::=(accept
:sender<Agent_name>
:receiver<Agent_name>)

(9)<rejection>::=(reject
:sender<Agent_name>
:receiver<Agent_name>)

(10)<exception>::=(error
:sender<Agent_name><Facilitator_name>
:receiver<Agent_name>)

Finally, we define meta-language exception, in order to process some exception cases. When one agent cannot handle messages from the other agents, for example, the communicator receives wrong KQML message, and cannot process it, then, performative error can be used.

5. Automated Negotiation Ontology

As can be seen from the above, ontology is an important component in the communication language.
Ontology is an agreement about a shared conceptualization, which includes frameworks for modeling domain knowledge and agreements about the representation of particular domain theories, often formally captured in some form of a semantic web. Its aim is to represent the shareable conceptual model in formalized specification [16].

There are two kinds of ontology in NACM, kqml-ontology and negotiation-ontology. Kqml ontology has been defined formally. We can find the OWL version from The DARPA Agent Markup Language web site.

For example, the concept of “Agent” and “Facilitator” in kqml ontology can be expressed as: [17]

```
<owl:Class rdf:ID="Agent">  
  <rdfs:comment>Agent</rdfs:comment>  
  <rdfs:label>Agent</rdfs:label>  
</owl:Class>
```

The ontology is defined in terms of the following concepts; each of them highlights a different aspect of a negotiation:

1. Classes: concepts related directly to the negotiation, such as negotiation protocol, negotiating Agent, role and negotiation rules and so on. Concretely, negotiation protocol defines a generic protocol defining the “rules of encounter” that are followed by negotiation participants during a negotiation process. The rules describe the conditions defining the interactions between agents and the deals that can be made [18];

Figure 3. UML model of the negotiation ontology

Negotiation ontology is based on the idea that there are some general concepts that are presented in any negotiation, and it is built on finding commonalities across different negotiation protocols. From an analysis of the classification framework illustrated in [18], the generic software framework for automated negotiation [19], and the work by Samir Aknine, Suzanne Pinson, and Melvin F. [20], we have identified the concepts and the relationships that are shared by most negotiation protocols. Figure 3 shows the negotiation ontology resulting from this merging process using an UML model to represent concepts and relationships.
Agent describes a single agent or an organization of agents which participate in a negotiation. Several agents can negotiate, and they can play different roles in the negotiation; objects describes the objects of the negotiation, that is the material or immaterial goods that are transferred once an agreement has been reached; Negotiation rule is a set of rules that govern a specific negotiation protocol. In the ontology this means that we identify a number of negotiation rules, and the way in which they are specified defines a specific negotiation protocol. This concept is specified by the different types of rules identified in [20]; Role describes the role an agent or an organization of agents plays in the ontology. Role represents the participants in the negotiation rule. It is specified by the roles identified in [20].

Offer: in order to express demand, negotiating agent proposes possible value for various issues of negotiation. For example, the quantity and price of the product.

2) Relations: describes the mutual effect between concepts in the field of automated negotiation. The basic relation in the model is from the object oriented theory. As fig3 shows, English_Auction class inherits Protocol class. Negotiation_Host and Negotiation_Participant are component of class Role. Class Protocol, Agent and Object have certain relation between them. Besides, some special relation is constructed according to concrete model. For example, there is a relation has Actor between concept Protocol and concept Agent, for describing some negotiating agent interact with each other in a negotiation protocol. The relation plays between concept Agent and Role describes that different Agent plays different role in a process of interaction. Negotiation protocol is controlled by negotiation rules through relation governs.

The relationships between the concepts are also defined to describe how the identified concepts interact to define the negotiation protocol domain. For example, a Protocol Has Agent which models the fact that at least two or more (2...*) agents interact in one negotiation protocol. That A Party Plays Role is modeling the fact that one agent can play a number of (1...*) different roles in the interaction. A protocol is also governed by a number of negotiation rules, and this aspect is also modeled by means of the relationship.

In the process of expressing ontology knowledge, the concept is expressed in detail through constraining or adding concept’s attribute. For example, concept Protocol is described through adding attribute hasActor and hasObject. The following example is based on English Auction, whose ontology is expressed by DAML+OIL language. The following code is a part of whole.

6. Conclusions and Future Work

In this paper, a novel agent concept, namely negotiating agent, is established. On the basis of the theoretical model of negotiating agent, NAA and NACM defined above are explicit and formal specifications for the agents negotiating in an E-business environment; especially, NACM defines the negotiation language template shared among all agents formally and explicitly. The novelty of the communication model is twofold. In fact it is synthesis work in both agent communication technology and automated negotiation theory, which are important areas of e-business research. More importantly, NAA and NACM build the foundation for developing an automated negotiation system.

The approach we have presented in this paper is still at a very early stage, and there are a number of issues that need to be further investigated. One is how to perfect the negotiation ontology. Creating and expressing any size of ontology is difficult and time consuming work. We still need to investigate whether this negotiation ontology is sufficient to permit the necessary interaction or whether a different type of knowledge should be included in the ontology. Another aspect which we have disregarded in the paper but which we are planning to investigate is how to design an analogous negotiating model complying with FIPA specifications.

7. REFERENCES


