DEVELOPING A MULTI-AGENT AUTOMATED NEGOTIATION SERVICE
BASED ON SERVICE-ORIENTED ARCHITECTURE

Mu-kun Cao
School of Management
Xiamen University, Fujian, China, 361005
camukun@yahoo.cn

Robert Chi
College of Business Administration
California State University, Long Beach, CA, USA
rchi@csulb.edu

Ying Liu
College of Business Administration
California State University, Long Beach, CA, USA
yingli97@hotmail.com

ABSTRACT
Automated negotiation has become the core of the intelligent e-commerce. Traditional research in automated negotiation is focused on theory about negotiation protocol and strategy. However, the application of automated negotiation system has lagged far behind. This paper discusses reasons for such a situation and points out that making the automated negotiation system as a software service in line with the SOA (Service-Oriented Architecture) is a feasible way for the practical application of the automated negotiation system. It then discusses a technology roadmap for the development of automated negotiation system using the software agent technology. As a proof of concept, this paper proposes an application architecture using SOA and web services technology and describes the running process of the whole system.

Keywords: Automated negotiation, Agent, Software as a service, Service-oriented architecture, Web services

1. Introduction
Electronic commerce is affecting business profoundly. It is changing the way businesses interact with consumers, as well as the way businesses 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 pricing mechanism automated and intelligent [Bartolini, 2005]. Today, e-commerce is characterized by dynamic trade. Its role has not only been confined to provide trading places and related information for the buyers and sellers, but has changed from a simple trade matcher to a price coordinator. The tremendous successes of online auctions show that the dynamic trade based on e-negotiation will gradually become the core of e-commerce.

Traditionally, e-negotiation processes have been carried out by humans registering themselves at certain web pages, placing bids and making offers through fill-in forms, and receiving counter-offers of other participants by updating the respective web pages. One major drawback of this human-centred way of e-negotiation is that the underlying processes are not explicitly modelled and the knowledge and experience are kept within the human minds. The next step towards automated e-negotiation systems is to be achieved through software agents that act as automated participants in the e-marketplace. Automated negotiation is an important research area of e-commerce negotiation after the negotiation support system (NSS). It is the direction of development for the e-commerce automated trade. Its main principle is using software agent technology to make the process of negotiation to be part or full automation. The purpose is improving the efficiency of online negotiations and reducing costs of trade. Since the mid-90's last century, the research on automated negotiation system has been given high priority by researchers [Lomuscio, 2003].

At present, one severe problem for the research of automated negotiation is how to take the theoretical results into practical applications. For more than ten years, compared with the fruitful results of theoretical research, the application of automated negotiation systems has lagged far behind [Dubey, 2007; He MH, 2003]. So far, there is scarcely any practical automated negotiation system that can be applied in e-commerce [Resinas, 2006]. The realization of system remains at experimental stage as most prototype systems were developed in university
Laboratories and automated negotiation technology is still under investigation in research labs. It could take some time before practical B2B e-commerce systems to adopt a negotiation tool [Kwei-Jay Lin, 2008]. That has been a barrier for the research of automated negotiation. The fact that many good theories, models and algorithms cannot be verified without a practical application platform constrains the further development of automated negotiation research. This situation is widespread in the entire e-commerce oriented automated trading applications [Debenham, 2005]. While some systems (such as the Kasbah, Tete-a-Tete, ADEPT, AuctionBot, eMediator, MAGNET, MATE) have been widely cited in literatures, strictly speaking, they are not automated negotiation system. Because software agent technology in these systems is mainly used to achieve automated trading functions, such as product selection, price comparison [He MH, 2003] and so on. Even some applications involved in negotiation are mainly for auction. This situation can be mainly attributed to the followings:

First, the theory of automated negotiation is still under development, new theories and methods appear one after another. The industry and academia have not yet reached a consensus for some basic questions, and have not formed a unified standard.

Second, the research of automated negotiation has close relationship with the disciplines of artificial intelligence, multi-agent systems theory and software agent technology. These disciplines are not mature and constrain the actual application of automated negotiation system. Automated negotiation system is essentially a multi-agent system, whose development is inseparable from the software agent technology. Therefore, the development of automated negotiation system has no technology roadmap to follow nowadays.

Third, the basic rules of software engineering tell us that the creation, development and application of a new technology is a time consuming process. Human’s psychological acceptance of this new technology is also a time consuming process.

Finally, the automated negotiation system does not have a suitable application model for its own development. This is one of the most important reasons. The first three are common reasons for most new technologies, and are likely to be encountered during the process of development. However, under the circumstances of the rapid development of e-commerce applications and network technology nowadays, researchers cannot ignore the fact that the automated negotiation system has developed too slowly. We should solve this problem first to pave the way for future automated negotiation research.

2. Negotiation as a Service

As mentioned above, a lot of negotiation systems have been produced in an academic environment. Usually they are experimental systems developed to verify a particular model or theory. In those systems issues such as usability, scalability, and security need to be resolved. There is a long way to use them in the practical commercial application. At the same time, researchers find that it is difficult to meet completely such requirements of industrial development. Consequently these systems only stay in the early stage of system’s development and application. Moreover, many such systems were abandoned later, which is a great waste. As a matter of fact, we observed that:

First, in the current network environment, businesses and individuals do not often want software but a service.

Second, research teams in university are not fit to play the role of software companies. They are more suitable to act as a service provider.

Therefore, we need to introduce a new operation model to support the whole process from research to application. The wide adoption of Software as a Service (SaaS) validates the above observation. SaaS, as a new technology emerged in the 21st century, is a completely innovative application model of software. It provides software services through Internet. Software providers deploy application software on their own servers. Customers search and access software services via the Internet, consume software services based on their own demand, and pay the software providers based on the number of types of services and the length of time [Choudhary, 2008]. The emergence of SaaS subverts the traditional life cycle (design - development - testing – Installation) of software. Users buy software service as an alternative to permanent license. The design of software has changed from original function-oriented design into a service-oriented design. Users need not to concern any more about the upgrade and maintenance of software, but directly lock on services [CHEN Peng, 2008]. According to the data from IDC, the global scale of SaaS will grow continuously with a compound annual growth rate of 21%, and will reach 10.7 billion U.S. dollars in 2009. Gartner expected that, before 2011, 25% of the global enterprise software applications will be delivered in the way of SaaS [Dubey and Wagle, 2007]. These are evidences that SaaS is the future direction for the development of the software industry.

Deploying automated negotiation system in the form of web service is a trend in the future. The negotiation system is a software system that provides negotiation service to its users. SaaS points out the direction for the application of automated negotiation system, as well as provides an idea for scientific research in colleges and universities. Most of the current automated negotiation prototype systems are developed in laboratories in
universities and usually do not have completed set of e-commerce functions. Research teams should cooperate with third-party e-commerce platform, and deploy the application system in the e-commerce portal, perform test-run for the application in the mode of SaaS. We can use service-oriented architecture (SOA) to make the system running in any place as a service node that is seamlessly integrated with third-party e-commerce platforms. The system can play the role of negotiation service provider in the real business environment.

The benefits for doing so are obvious: first, we can obtain stable visiting quantity. Second, the latter maintenance and upgrade of system can be completed on the server independently, saving human and material resources. Third, the automated negotiation system can make use of the existing basic facilities provided by the e-commerce platform, such as security, authentication, transaction management, and so on, saving costs of development. In fact, this kind of development mode is meaningful for industrializing the other scientific research contributions generated from colleges and universities.

3. **Implementing Negotiation Service with Web Services**

   A very important requirement for the automation of e-negotiation processes is the interoperability between the partner systems in the e-negotiation application domain. In order to meet this requirement, the dynamic invocation of web services based on SOA (Service-Oriented Architecture) is the most appropriate approach [Rinderle and Benyoucef, 2005]. According to [Kim and Segev, 2005] web services are the most appropriate way to deploy e-negotiation systems for the following reasons: (1) relationships between negotiating partners are dynamic therefore run-time binding is preferable to design-time binding; (2) negotiation is part of procurement, therefore the interoperability of both internal and external IT systems is important; and (3) web services provide a standard and flexible integration technology that no organization can afford to ignore if it wants to interact with its partners.

   The term SOA refers to a style of building reliable distributed systems that delivers functionality as services, with the additional emphasis on loose coupling between interacting services. Technically, the term SOA refers to the design of a system, not to its implementation. We regard SOA as an architectural style that emphasizes implementation of components as modular services that can be discovered and used by clients. SOA has the good characteristic of loose coupling, which implies that the interacting software components minimize their built-in knowledge of each other; they discover the information on demand. For example, having learned about a service’s existence, a client can discover its capabilities, its policies, its location, its interfaces and its supported protocols. Once it has this knowledge, the client can access the service using any mutually acceptable protocol.

   Web service is a type of implementation of SOA. It allows e-commerce activities to be carried out based on a set of XML standards such as Simple Object Access Protocol (SOAP), Universal Description, Discovery and Integration (UDDI) and Web Services Description Language (WSDL). Traditional business-to-business applications connect trading partners through non-standardized architecture. A major drawback is the lack of interoperability where setting up connections with a new trading partner can be costly and time consuming. In contrast, the benefits of adopting Web services include faster time to production, convergence of disparate business functionalities, a significant reduction in total cost of development, and easy-to-deploy business applications for trading partners. In addition, Web services provide a flexible, highly coherent, and loosely coupled interface, which supports both programmatic and human users [OGSA, 2009]. Web services provide the means for software components to communicate with each other on the web using XML. A web service describes itself (using WSDL), can be located (using UDDI), and invoked (using SOAP). An online auction based on web service can be deployed on the e-marketplace, and located and invoked through the web by a distant Automated Negotiation System. [Rinderle and Benyoucef, 2005]

   The service-oriented architecture is very useful for the application of automated negotiation system. As mentioned above, there are many difficulties, such as security and audit, to make the automated negotiation system to be applicable. In fact, in most cases, the enterprises and individuals do not want a software system for commercial negotiation, but just a negotiation service. On the other hand, it is more convenient for the developers to just provide a negotiation service to the users, rather than to deploy software negotiation system in the user’s company, for there will be less work of late maintenance, update and so on. Therefore, we can make the automated negotiation system as a service node, and deploy it in an enterprise application integration system based on service-oriented architecture. To do so, the automated negotiation system can take advantage of the infrastructure, such as network, security, transaction and so on, provided by the whole integration system, and benefit from the existing customer resource.

   Negotiation service generally has the following characteristics. 1) It is individually useful, and also can be integrated and composed to provide higher-level services. Among other benefits, this promotes re-use of existing functionality. 2) It communicates with their clients by exchanging standard-based messages: they are defined by the messages they can accept and the responses they can give. 3) It can participate in a workflow, where the order in which messages are sent and received affects the outcome of the operations performed by a negotiation. 4) It is
completely self-contained, but to some extent, it depends on the existence of a resource such as a database. In the simplest case, a negotiation service can function without needing to refer to any external resource, or it may have pre-loaded all the data that it needs for its lifetime.

Moreover, the SOA-based automated negotiation system will have the following good cross-platform features: 1) Flexibility: A negotiation service can be located on any server, and relocated as necessary. As long as it maintains its registry entry, prospective clients will be able to find it. 2) Scalability: negotiation services can be added and removed as demand varies. 3) Replacing ability: provided that the original interfaces are preserved, a new or updated implementation of negotiation service can be introduced, and outdated implementations can be retired, without disruption to users. 4) High availability: If a server, a software component, or a network segment fails, or the negotiation service becomes unavailable for any other reason, clients can query the registry for alternate services that offer the required functionality, and continue to operate without interruption.

4. Automated Negotiation System Based On Software Agent Technology

Before the service-oriented application of the automated negotiation system is proposed, we must first design the negotiation service. A negotiation service is built on a real automated negotiation system, which is a self-contained software system.

As emphasized by Badica, multi-agent theory and technology is the main way for realizing the e-commerce automation [Badica, Ganzha and Paprzycki, 2007]. An automated negotiation system needs full support from multi-agent system theory and software agent technology for the following reasons.

Negotiating agent is composed of various cognitive components (believe, desire and intention), which make the agent has the ability to negotiate independently. The contents of these cognitive components define the cognitive state of agent. Agent makes decision to negotiation autonomously on the basis of its internal cognitive state. That is different from the basic module unit in traditional structured programming, process and function, and the basic module unit in object oriented programming, object. Therefore, it is not suitable to develop the negotiating agent through traditional process-oriented or object-oriented programming methods, but the Agent-Oriented Programming (Agent Oriented Programming, AOP).

The realization of automated negotiation system needs to use software agent technology. However, analysis, design and realization of multi-agent system, so far, do not have formal, systematic approach to follow. Although there are many development methods and modeling techniques have been put forward, they are still in the phase of research and experiment. Together with the variety and complexity of agent-oriented software development tools, the development of multi-agent automated negotiation system is not mature. This is an important factor hindering the realization of automated negotiation system.

As a result, we firstly discuss the technology roadmap for the development of automated negotiation system using the software agent technology. After that, we will propose a practical application architecture using SOA and web services technology for the service-oriented negotiation application.

4.1. Technical Background

It is necessary to make full use of existing software agent development platform, rather than reinvent the wheels. There are many agent-oriented software development platforms that provide the required infrastructure for the multi-agent system, such as message processing, tracking and monitoring, operation management and so on. Multi-agent automated negotiation system does not care about the underlying infrastructure, but needs to realize the business logic and communication logic in the negotiation process.

In automated negotiation system, negotiation between agents is a high-level social interaction, which is realized by agent communication language. Agent Communication Language is a special kind of message transport format, which can not only expresses clearly the intention of interaction between agents, but also describes in detail the contents of interaction. So that it can effectively contribute to the complicated collaboration between agents. This is different from the message transport between objects in object-oriented programming, for messaging between objects is essentially activating methods. Therefore, we have to use agent development platform to realize the communication between agents.

Agent development platform provides a full range of support for the development of agent system, such as support for communications, security, directory management and other services. Among them, the support for the realization of communication currently is one of the most important features of an agent development platform available for its applications. Some scholars even believe that a software system is called an agent development platform, if and only if it can support communication between agents [Genesereth and Katchpel, 1994]. Using agent development platform reduces the developers’ burden of the work, thus contributing to the rapid and efficient development and deployment of agent-oriented software systems.

Since 1980s, as the main driving force for applying agent technology, the agent development platform has been
received great attention from academy and industry. Up to now, dozens of Agent Development Platform tools have been developed, but also caused severe heterogeneous problems in programming languages, network transmission protocols, operating systems, communication languages, agent architectures and so on.

These heterogeneous hampered the applications of agent development platform, especially the interoperability between agent systems. For this reason, the academia and industry have committed to the standardization of agent technology and developed a number of general specifications that usually regard agent communication language as the core content. Standard communication language is a fundamental solution for the interoperability between heterogeneous agent systems. Two popular specifications are KQML developed by KSE and FIPA developed by Intelligent Physical Agent Fund.

Correspondingly, the current agent-oriented development platform can be divided into KQML-oriented tools and FIPA-oriented tools. KQML-oriented agent development platforms include JATLite developed by Stanford University, which is a Java software development kit used to build multi-agent system; Zeus developed by British Telecommunication Co. and so on. A FIPA-oriented agent development platform is JADE (Java Agent Development Environment) developed by Italy Telecom Lab, which is a Java-based agent integrated development platform [Java Agent Development Framework, 2009].

KQML specification is a product at the end of the last century because the most recent version was released in 1997. There is no new development in recent years. The agent development tools and platforms following KQML specification will gradually withdraw from practical applications, thus it is unsuitable to use these tools for the development of new multi-agent system.

FIPA, since its establishment in 1996, has maintained a strong momentum of development, has developed a package of specifications, the latest version is FIPA2000. Moreover, FIPA is more comprehensive than KQML, involving not only the content of agent communication language, but also agent life-cycle management, architecture, interactive protocol and more. It can be foreseen that FIPA will eventually be the common standards of industry in the future. Many agent development platforms based on FIPA specification also maintain good momentum of development.

In conclusion, the development of multi-agent automated negotiation system will follow the FIPA specification, and JADE will continue to be the development tool of choice for system implementation.

4.2. Software Architecture for Automated Negotiation System

According to the technology discussed above, we propose an architecture design of automated negotiation system based on JEE and JADE architecture.

A complete automated negotiation system, similar to a negotiation support system, needs some auxiliary components to support the negotiation process, such as user management, information release, contract management and so on, which can guarantee the negotiation process to be implemented smoothly. We will reuse the components of an existing negotiation support system to serve for the new automated negotiation system. It is equal to adding functions of automated negotiation to the original negotiation support system. Figure 1 shows that there are five layers in the software architecture of the automated negotiation system. It derives from the three layers (data layer, business logic layer and presentation layer) architecture of JEE technology. Starting from the bottom:

Data layer consists of database, decision models and negotiation strategy models, which are used by the negotiating agent and the whole system in the process of negotiation. Database plays an important role in the whole system. It administrates and maintains not only the runtime data of automated negotiation process, but also the data of every system function module supporting the process of negotiation, such as data of user information and production information etc.

Message transport layer is in charge of transporting FIPA ACL message to the negotiating agent in the system. At the same time, it can also communicate with other agent platform via HTTP, IIOP and SOAP, making the local negotiating agent can interact with agents on other remote agent platforms.

Agent management layer is made up of two components, Agent Management System (AMS) and Directory Facilitator (DF). It implements the communication facilitator used in most abstract automated negotiation communication models. AMS performs basic operations on the negotiating agent, such as creating and deleting a negotiating agent, modifying the agent’s description, and monitoring agents migrating among different platforms. Any negotiating agent resides in the current agent platform must register on the AMS. AMS assigns an ID to each negotiating agent. DF provides yellow page query service for all negotiating agents. It can provide timely, accurate and complete listed information about negotiating agents in current system. Every negotiating agent has to publish the detailed information by themselves via DF, such as their names, addresses, services and so on. Other negotiating agents can find other agents by searching DF. AMS and DF together constitute the core of agent management system. Their function definition comes from FIPA specification SC00023J [FIPA, 2009], and can be implemented with the help of JADE platform.
The negotiating agent provides services for the automated negotiation module in the business layer. The business layer is divided into two function modules, automated negotiation module and negotiation support module. The automated negotiation module mainly contains the automated negotiation protocol algorithms. The negotiation support module is a legacy system that includes functions supporting the negotiation process online. The whole business layer is deployed in the EJB (Enterprise Java Bean) container and can provide data persistence and transaction management for the whole negotiation system, and can ensure good security and stability.

The presentation layer uses the technologies of JSP, Java Servlet and HTML to produce web page dynamically on the browser, to help the interaction between the system and the users. This layer runs on the web container, whose function is responding to the web request based on HTTP protocol.

5. **Abstract Architecture for Service-oriented Automated Negotiation System**

From the abstract viewpoint, the service-oriented automated negotiation system includes the following four participants: service registration centre, negotiation service requester, negotiation service provider, and protocol.

Service registration centre is a database of service providers’ information. The negotiation service provider follows a standard service API so that customers can use negotiation services from different service providers. Registration service centre supports all customers and services in the open system, of which negotiation is just one. Service registration centre provides customers with other services such as security and auditing in addition to negotiation service registration.

Negotiation service requester discovers and calls negotiation service. It provides business solutions to enterprises and individuals using the negotiation service. In general, the service requestor does not directly interact with service registration centre, but through an application portal system. The benefit for doing so is the application portal can provide users with access to a wide range of services.
Negotiation service provider, managed by the negotiation software vendors, advertises negotiation service to the service registration centre, including the registration of their functions and access interfaces. It responds to the service request. At the same time, it must ensure that any modifications to the service will not affect the requester.

Protocol is an agreement between negotiation service requesters and providers. It standardizes service request and response and ensures mutual communication.

Figure 2 illustrates a simple negotiation service interaction cycle, which begins with a negotiation service ADVERTISING itself through a well-known service registration centre. A negotiation requester, who may or may not run as a separate service, queries the service registration centre to DISCOVER a service that meets its need of negotiation. The service registration centre returns a (possibly empty) list of suitable services, and the service requester selects one and passes a request message to it, using any mutually recognized protocol. In this example, the negotiation service responds either with the result of the requested operation or with a fault message. Then they INTERACT with each other continuously.

6. Architecture for Automated Negotiation System Based On SOA

SOA is an architectural style, while Web services are an implementation technology. Service-oriented automated negotiation system requires the following technical specifications based on SOA [HP, 2009]: XML (Extensible Mark-up Language), SOAP (Simple Object Access Protocol), WSDL (Web Service Description Language), UDDI (Universal Description, Discovery and Integration) and so on. WSDL is programming interface for describing Web Services used by the programmers. Web Services can register their properties on the UDDI, and other applications can find their needed web services through the UDDI. SOAP provides means of communication between applications and web services. WSDL, SOAP and UDDI are based on the XML technology.

UDDI service registration centre is the core of integrated application services and is a technical specification for querying web services. Negotiation service providers can issue negotiation service on the services registration centre and negotiation service requestor can find the desired services in the registration centre. There are two possible types of UDDI registration centres in the architecture: one is privately established by the negotiation service provider itself (not showed in Figure 3), located within the automated negotiation system, mainly for service-oriented negotiation. It can query and discover web services inside the system. Another UDDI is public (as showed in Figure 3), built up by a third party e-commerce platform. It mainly provides services for the clients visiting the web site and can also be a popular public registration centre. Here, we assume that the automated negotiation system, as a service node, is deployed outside the third party e-commerce platform.

SOAP router supports interaction between entities of the web services. It provides a lightweight mechanism for exchanging structured information in the Internet. It provides the interoperability between heterogeneous applications. When the service requestor chooses a negotiation service, it will use the WSDL description to find how to access to the negotiation services; once found, WSDL description will be used to generate SOAP request message sent to the application server. Application server plays the role of negotiation service provider.

In Figure 3, the automated negotiation system is deployed in the SOA-based architecture of enterprise application integration. In the architecture, the clients (buyers and sellers) visit a third party e-commerce platform
through Internet, and request a service. Of course, the clients can request for varieties of services provided by the platform. The application server of the e-commerce platform accepts the service request in the abstract architecture. It sends a request to the public UDDI Registration Server to find a required service. We assume the service here is negotiation and is already registered in the public UDDI Registration Server.

Besides providing the function of publishing and querying product information to the client, the third party e-commerce platform can also provide varieties of trade services, such as negotiation service, auction service, direct trade service, and so on in order to meet different trading requirements. All the services information and service interface specification are posted on a directory server: the Public UDDI Registration Server. The service interface specification is described by the universal WSDL. Users visit the third party e-commerce platform through Internet. Sellers publish product information on the e-commerce platform. Buyers query the product information and then determine the trade object. Buyers and sellers query the UDDI Server for suitable transaction and obtain the Web Services’ invocation information. Assuming the negotiation service is discovered and invoked; the system binds the buyers and sellers to the negotiation service and activates the automated negotiation system that is deployed in another server somewhere. The buyers and sellers start their negotiation on the automated negotiation system using SOAP messages.

WSDL plays an important role for building the architecture. Because the negotiation service must have the characteristic of loosely couple and automatic integration, which requires that the description of interface can be identified automatically by machines. Before a service requester invokes the negotiation service, it must know the call interface. WSDL describes the service interface of negotiation. It is defined in XML Schema style and able to describe service interfaces realized by a variety of language. The negotiation service is defined as a collection of network endpoints in WSDL, in which some elements, such as type, message and ports, are used to describe the service interface. Requestors know the data type, message structure and transferring protocol required by the service. So it can call the negotiation services.

![Figure 3: Architecture for automated negotiation system based on SOA](image)

To illustrate the registration service description of service-oriented automated negotiation system in Figure 3, a prototype service example is listed below. Using WSDL and UDDI document in brief XML.
Developing a Multi-Agent Automated Negotiation Service

1) WSDL Service Implementation

//the implementation document for the description of negotiations service
<definitions name = "NegotiationService"

targetNamespace = "http://www.xxx.com/NegotiationService" />
<import namespace = "http://www.xxx.com/NegotiationService - interface"

location = "http://www.xxx.com/wsdl/ANS-interface.wsdl" />
<service name = "NegotiationService">

<port name = "SingleSymbolService"

binding = "interface: SingleSymbolBinding"

<soap:address location = "http://www.xxx.com/AutomatedNegotiationService" />

</port>
</service>
</definitions>

2) WSDL Service Interface

//the interface document for the description of negotiations service
<definitions name = "NegotiationService - interface"

targetNamespace = "http://www.xxx.com/NegotiationService - interface" />
<message name = "SingleSymbolRequest">

<part name = "symbol" type = "xsd:string" />
</message>
<message name = "SingleSymbolResponse">

<part name = "symbol" type = "xsd:string" />
</message>
<portType name = "SingleSymbolNegotiationService">

<operation name = "startNegotiation">

<inputmessage = "tns:SingleSymbolRequest" />

<outputmessage = "tns:SingleSymbolResponse" />
</operation>
</portType>
...}

<binding name = "SingleSymbolBinding"

type = "tns:SingleSymbolNegotiationService" }
...}
</binding>
</definitions>

3) UDDI Registry

//UDDI business service created according to the description of negotiations service, which is invoked directly from
//the implementation document for the description of negotiations service
<businessService serviceKey = "...">

<name>NegotiationService</name>

<bindingTemplates>
...
<overviewDoc>

<overviewURL>http://www.xxx.com/services/ANS.Wsdl</overviewURL>
</overviewDoc>
...
</bindingTemplates>
</businessService>

Our prototype system was developed using JWSDP (Java Web Service Developer Pack) provided by Sun Microsystems. In particular, the JAXP (Java API for XML Processing) and JAXB (Java Architecture for XML Binding) were used to manipulate the XML documents. The JAXM (Java API for XML Messaging) was used to send messages over the Internet in a standard format using the SOAP method calls. Essentially, this is how the offer information is communicated between the negotiation server and the client application programs. The JAXR (Java API for XML Registries) was used by a client program to query a service registry to identify the appropriate negotiation services.
7. The Process Model of the Automated Negotiation System

The design of the proposed automated negotiation service is based on a sound B2B E-Commerce model call Business-to-Business Transaction Model (BBT) [HE, Jennings and Leung, 2003]. With reference to the BBT model, B2B business process can be divided into six important stages [Raymond and Lau, 2007]. The first stage is “Partnership Formation” that involves finding the business partners that provide products or services in a supply chain as well as forming a virtual enterprise; the second stage is “Brokering” which is the process of matching sellers who supply goods or services to the buyers who require them; the third stage is “Negotiation” where the traders aim to reach an agreement about what actions should be performed under what conditions; the fourth stage named “Contract Formation” marks the end of negotiation and involves the agreed terms being put into a legally binding contract; the fifth stage named “Contract Fulfillment” refers to the parties executing the agreed transactions according to the terms specified in the contract; “Service Evaluation” is the final stage where traders evaluate their satisfaction with the transactions so as to prepare for another partnership formation in the future. The Service-Oriented Automated Negotiation System (SOANS) can directly support “Partnership Formation”, “Brokering”, “Negotiation” and “Contract Formation”. Moreover, it can be easily extended to support the last two stages of the BBT model. A real-world scenario of applying the SOANS platform to streamline B2B E-Commerce can be illustrated as follows (see Figure 4):

![Figure 4: Operation process of automated negotiation system](image)

According to the architecture of automated negotiation system proposed above, we can see that the whole running process of the system includes the negotiation support process and automated negotiation process. Product providers, automobile manufacturers for example, publish their product information on the third party e-commerce platform.

When a wholesale car dealer wants to purchase a batch of family cars from some automobile manufacturers (i.e., a B2B scenario), the car dealer queries the product information to find a car manufacturer. Then, the buyer and seller go into the match stage, and they apply a negotiation service to the e-commerce platform, and wait for binding to the negotiation service.

Before the application submitted to the automated negotiation system, the buyer and seller should choose whether they want to use agent to delegate their negotiation. If they choose to use agent, then the negotiation will go into automated negotiation process. If they choose not to use agent, then the negotiation will go into negotiation...
support process. Of course, there could be a case that one side chooses to use agent but the other side does not. In order to cope with this kind of situation, our system is designed to have the ability to process negotiation between human and agent.

In the case of automated negotiation, users first create their negotiation agent. This step includes some preparation for initiating negotiation agent, such as setting basic parameters, setting preferences, selecting negotiation strategies and so on.

The created negotiation agents register on the Agent Management System (AMS), and then the agents of the two sides query the Directory Facilitator (DF) to find one or more agents to make a match and start the automated negotiation.

If the two sides reach an agreement, the agent will return the result to the human user to make the final decision. If the users accept the result, the two sides will go into the stage of “Contract Formation”. After that, the negotiation agent will be released to quit the whole automated negotiation process.

If the negotiation doesn’t reach an agreement, or the user reject the agreement made by the agent, the process can return back to the stage of automated negotiation to build another negotiation, or users can choose to free the agent to quit the system.

8. Conclusions and Future Work

Web services and service-oriented style of architecture are widely seen as the basis for a new generation of distributed applications and system integration tools. This manuscript introduced the key concepts, relationships and benefits of these two technologies and showed how they can be combined to develop automated negotiation application systems that can be integrated in an enterprise application, regardless of the hardware and software platforms deployed in business partners. This manuscript showed how to use software technology to develop a multi-agent automated negotiation system and how to integrate the system with the service-oriented architecture. The result is meaningful for the development and application of e-commerce oriented intelligent trading system.

On the other hand, while the combination of SOA and Web services will clearly be the focus of development in the near future, it must be recognized that they still have a number of pervasive issues to overcome, including immature technology, conflicts and some confusion in the area of standards, and the significant performance overhead of serializing, de-serializing and parsing SOAP messages and XML documents on each message exchange. Therefore, these problems will also affect the development and application of the automated negotiation. They are to be resolved in the future.

Acknowledgement

The authors are grateful for a grant (2009J05162) provided by China Fujian Province Natural Science Foundation, and a grant (70902042) provided by National Natural Science Foundation of China for one of the authors to present an earlier version of the paper at the International Conference of Service Systems and Service Management (ICSSSM), June 2009. We also would like to thank anonymous reviewers for their constructive comments on the earlier version of the manuscript.

REFERENCES


