ICEB 2004 Proceedings

International Conference on Electronic Business (ICEB)

Winter 12-5-2004

A Multi-Agent System for E-Business Processes Monitoring in a Web-Based Environment

Xinyu Zhao

Cen Wu

Runjie Zhang

Chen Zhao

Zuoquan Lin

A Multi-Agent System for E-Business Processes Monitoring in a Web-Based Environment

Xinyu Zhao, Cen Wu, Runjie Zhang, Chen Zhao, Zuoquan Lin

Department of Information Science, Peking University, Beijing 100871, China
{xinyua, wucen, zrj, zchen, lz}@is.pku.edu.cn

ABSTRACT

In this paper, we present a multi-agent system MAGS for the e-business processes monitoring in a web-based environment. We classify the types of agents in MAGS by their monitoring capabilities. An algorithm is given to explain the mechanism of supervising and controlling the execution of business processes. An abstract model of alerts, which can give warnings of infringement on business policies, is proposed. Access control can also be realized by MAGS, which manifests in delivering different view of the business process to different roles participate in it. Being successfully adopted in a customer service management system, MAGS has been proven flexible and practical.

Keywords: Business Processes Monitoring, BDI, Agent, Capability, Web Services

1. INTRODUCTION

A trend of e-Business is to wrap legacy applications with web services technologies, by allowing enterprises to integrate internal existing processes with the trading partners' in a manageable manner. As such, a business process specifies a potential execution order of operations from a collection of web services involving multiple organizations. With the number of complex business processes increasing, automated monitoring technology is in great demand. However, the traditional passive, centralized business process management system is not applicable for fully distributed applications in a web-based environment. For example, without initiative supporting and monitoring, it is hard to response timely and accurately to numerous events from other applications inside and outside the enterprise. Moreover, the detailed information about distributed business processes should be gathered by the requirements of different users and the relationships with the particular business process instances. Nowadays most traditional systems can not deal with the above issues well.

Software agent is an object which has autonomous actions of accomplishing specific task [1]. In multi-agent system, an agent interacts with other agents or environments to achieve its goal by communication, coordination and cooperation. Of various agent architectures which have been presented, BDI model [2] [3] [4], is probably the most mature and has been adopted in a number of research and industrial applications. There are some works introducing multi-agent into the execution monitoring of business processes [5] [6] [7]. The rationale-based monitors [5] aim at planning and removing the conventional assumptions in static and determinate. Classifications of monitors are proposed to suggest plan transformations. Execution Assistants [6] is implemented to assist human in monitoring robots' behaviors. A top-level categorization of alerts is presented, but the next lower level of the ontology is not provided. Execution Assistants emphasize the VOI (Value Of Information) and VOA (Value Of Alerts) and analyses the relationships between the two domains. Continuous Planning and Execution Framework [7] defines a monitor to be an event-response rule. Taxonomy of monitor classes derived from the types of events is believed to enable simpler and more modular specifications of monitors. Considering in the new web environments, in which multiple services and applications interact with each other, richer monitoring capabilities, such as alert technology and access control, are in demand for the monitoring management of the business processes.

This paper presents a multi-agent system MAGS (*Monitoring AGent System*) for the monitoring management of the business processes in a web-based environment. Conventional BDI agent model is extended to BDIC (Belief, Desire, Intention and Capability) model with monitoring capabilities, which are induced from a real application. MAGS is assigned to supervise and control the business processes following the profile of requirements of administrators. Also, MAGS is able to continually monitor the incoming system events, and then carry out alerts to the corresponding service providers accurately and timely via multiple channels. MAGS is able to cooperate with other agents to get the role of the users involved in the business process instance and give the detailed information about the instance. We will demonstrate how the implemented MAGS is integrated in a customer service management (CSM) system and show some of its novel features.

This paper is organized as follows. In Section 2, an example drawn from a distributed call center is depicted to show our motivation throughout this paper. Section 3 presents the architecture of MAGS and the BDIC agent model. In Section 4 the implementations of MAGS are discussed. Finally, conclusions are drawn and future works are discussed in the concluding section.

2. MOTIVATING EXAMPLE

We will illustrate our motivations with a distributed call center example. Internet–Enabled Customer Services [8] have become the most important channel, through which customers can talk directly to the corporations. Customers can call and state their requests to the distribution corporation. When the call center receives a call from a customer, the broker of call center divide the requirements into different service types, e.g., advisement, repair and so on. The broker should find a business partner, for example Service Net, from which customers can get the services, to fulfill the customer's request. When the proper business partner is found, the broker will make an appointment with the customer. Meanwhile, the customer's information and requirement are dispatched to the Service Net via multiple channels, such as short message service (SMS), email, web and fax, etc. After the requested service is finished, the Service Net submits a service report to the call center. To keep a satisfactory service level, the call center will collect the customers' feedbacks and track the Service Net's performance measure daily. If there are customers making complaints, their requirements will be re-accepted. At the same time, the department of service quality control will evaluate the quality of the whole business process. A typical business process of dealing with the customer incoming call is shown in Figure 1.
Due to the distributed properties of the system, the business processes might not be executed as we expected. Without automated support, the call center cannot deal with numerous unexpected events correctly and timely. Usually, it might be required that 95% of received calls should be answered within 10 seconds. Automated alert technology is required to give the warning of exceptions. Different roles participating in the business processes, such as call center, Service Net and service quality control department, have different information requirements. The details of running and completed business processes should be given based on the roles of the requester. All above motivate us to make an investigation on the monitoring technology for cross-organizational business process in web environments.

3. MAGS ARCHITECTURE

3.1 BDIC Monitoring Agent Model

A belief-desire-intention-capability (BDIC) architecture includes an explicit representation for an agent's beliefs, desires, intentions and capabilities. In this paper, it's the capability component that determines agent's types and monitoring functions. We concentrate on monitoring mechanisms here and the details of agent capabilities can be found in [9]. We give the definition of BDIC Monitoring Agent first.

Definition 1 BDIC Monitoring Agent is a structural Monitoring Agent = (*B*, *D*, *I*, *C*, *Se*, *Ef*, *Pr*), where (see Figure 2.):

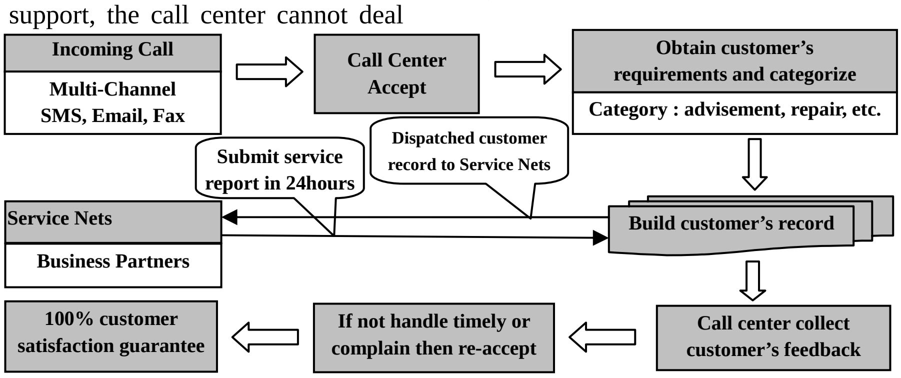


Figure 1 Closed loop diagram of an incoming call business process in call center

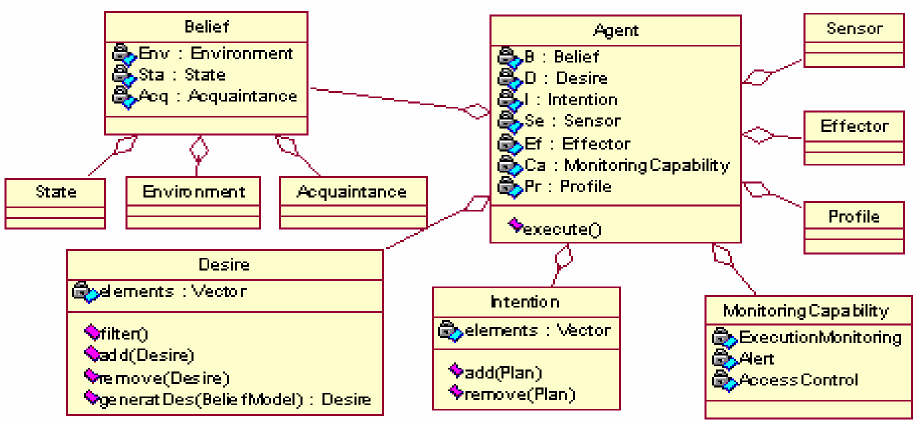


Figure 2 UML diagram of monitoring agent

B is the beliefs of monitoring agent, that is, the informational state of the business process in which the agent is monitoring. The beliefs include State, Environment and Acquaintance.
D is its motivational state, that is, what the monitoring agent is trying to achieve.
I is the intentions of monitoring agent, that is, the planning for eventual execution.
C is monitoring capability including categories of Execution Monitoring, Alert and Access Control. It will be further explained in the next sections (see Figure 3).
Se is the sensors of monitoring agent. It sensors information from environments or other agents by the standard message format e.g. FIPA, ACL.
Ef is the effectors of the monitoring agent by which the monitoring agent pursues its intentions.
Pr is the information requirements profile of system users.

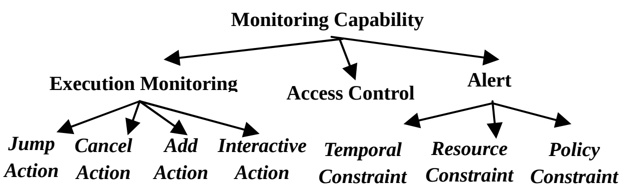


Figure 3 Monitoring capabilities

Monitoring agent is a typical BDI agent whose sensor perceives the information of business processes and revises beliefs. Desires is formed based on the current mental states *(B, D, I)*. Profiles are customized by the system users. Monitoring agent pursues intentions by effectors. The type of agent is determined by its monitoring capabilities.

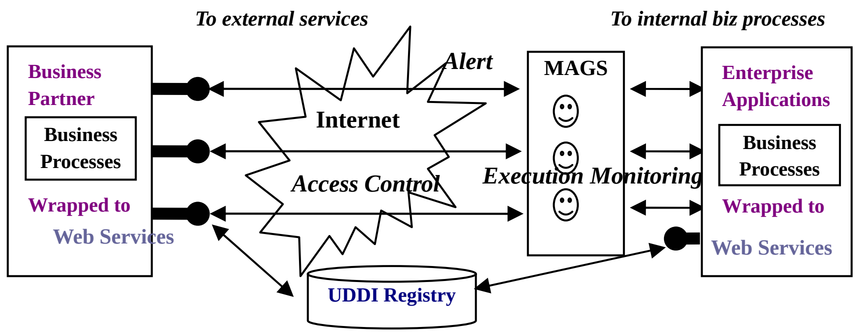


Figure 4 Integrated architectures

3.2 Integrated Architectures

In order to make the following explanations explicit, we give a kind of definition for the business processes here. A business process is a set of one or more linked procedures that collectively realize a business objective. By packaging business processes as services that are accessible over the internet, enterprises achieve new and better means to utilize their own and other's applications (see Figure 4). In general, a business process includes preconditions, effects and execution body as the following definition.

Definition 2 A business process is a structural BP = {precondition, effect, body}.

3.3 Execution Monitoring Agent

Below is an agent monitoring-interpreter, which explains the mechanism of the execution monitoring of business processes.

```
Monitoring–interpreter
Initialize-state();
repeat
    if all of currentBP.effects are true
    then currentBP = getNextBP()        //     Jump Action
    end if
    if one of currentBP.precondition is false
    then tempBPList = getBPList()
    // Get a list of business processes that will make the
  false preconditions of the current business process true.
            if tempBPList == null
            // Cancel Action: no business process will
make the false preconditions of the current business process
true.
                then reportToAdmin()
                        currentBP = getNextBP ()
            else    currentBP = getBestBP(tempBPList)
                        //Add Action and Interactive
Action: interact with human to get the best business process
that can achieve the goal in high quality.
            end if
     end if
execute(currentBP.body)
end repeat
```

According to the monitoring-interpreter, the execution monitoring agent has the following actions:

Jump Action: If all effects of current business process are true in the dynamic environments, it is not necessary to execute current business process in normal sequence. Whenever the effects of current business process are already true at the beginning of the execution, the execution monitoring agent will perform a jump action which changes the business process engine's focus of attention to the next process.

Considering the case in our application, when the broker has accepted the calls, yet not dispatched it to the Service

Net, the customer calls again to tell that he has solved his problem. It means that the goal of the Dispatch process has been achieved, thus the execution monitoring agent leads the Feed-Back process to be the current process.

Add Action: Changes of the environments may make the current business process preconditions false so that it could not be executed. The execution monitoring agent watches the preconditions continually. If one of preconditions fails, the execution monitoring agent will add a new process to execute which can make the false precondition true. Thus current business process can keep going on.

Cancel Action: During the execution of current business process, its preconditions may become infeasible due to the unpredictability of environments. Going on business processes with infeasible preconditions, the system may throw unwanted exceptions. The execution monitoring agent will watch the preconditions of current business process. If one of precondition cannot be made true through any Add-Actions, the execution monitoring agent will report to administrators and cancel current business process.

Interactive Action: In real-life domains, there may be multiple processes available for achieving a certain goal. It requires human experience and judgment for the business processes to execute in a high quality. Whenever multiple choices appear, the execution monitoring agent will interact with system users to get their knowledge and choose the most suitable business process to execute.

3.4 Alert Agent

A business process is usually constrained by policies or rules that must not be violated during its execution. Alert agents give warnings of infringing on business policies and try to avoid the failure of business process.

Definition 3 Alert is a structure *Al = (Id, Re, Ty, Le, Ch, Co)*, where:

Id is the unique identifier of the alert.
Re is the receiver of alerts, which commonly are services provided by business partners.
Ty is the type of alerts, including temporal, resource and policy constrains.
Le is the emergent level of alerts.
Ch is the channel of alert sent through.
Co is the content of alert.

When the business policy is infringed during the business process execution, the alert agent sends an alert to the specific service provider. An alert has not only the identification information but also an emergent level of indicating its urgency. MAGS support alert agents to send alerts through different channels. Alerts are classified into three types by temporal, resource and policy constrains.

Temporal Constraints: The business processes are commonly demanded to execute under temporal constraints. Each temporal constraint consists of a sequence of actions and one or more pairs of temporal constraint ontology and time. The temporal constraint ontology and its meanings are shown in Table 1. Temporal constraint has the following form:

BPName TC_Ontology Time [&TC_Ontology Time…] action1 [; action2…]

It indicates that the business process with the name of *BPName* must not violate the temporal constraints; otherwise the alert agent will perform corresponding actions. The alert will also be composed and sent by alert agent.

Table 1. Temporal ontology

TC_Ontology	Meaning
Start-at	Start time must be at given time
Start-lt	Start time must be later than given time
Start-et	Start time must be earlier than given time
End-at	End time must be at given time
End-lt	End time must be earlier than given time
End-et	End time must be earlier than given time

In our domain, CSM provides interfaces of getting the entire view of the business processes for system users. Generating the entire view of a business process is a complex and time-consuming process and it is restricted by time constraints. The following is a temporal constraint in the profiles of alert agent:

GetBPView Start-et 8:00:00&Start-lt 20:00:00 reportToAdmin; rejectRequest

According to the constraint, the business process *GetBPView* should start earlier than eight and later than twenty everyday. It makes the *GetBPView* process stagger the rush hour with other business processes and ensure CSM safety. If a system user attempts to invoke the *GetBPView* process in the interval of 8:00:00 and 20:00:00, the alert agent will compose an alert to administrators and reject the request.

Resource Constraints: The business processes in a resource-bounded environment should not overspend the system's resources. Resource constraint has the following form:

BPName resource threshold [&resource threshold…] action1 [; action2…]

It indicates that resources spent by the *BPName* process must not exceed by the threshold; otherwise the alert agent will perform corresponding actions to give an alert.

Policy Constraints: The business processes must maintain the business policies that must not be violated. Alert agent gets policies from its profile and continually monitors the incoming events unsatisfying conditions in the policy constraints. If any constraint is violated, the alert agent will perform corresponding actions to give alert. Policy constraint has the following form:

BPName condition [&condition…] action1 [; action2…]

(RBAC) system and can cooperate with MAGS in offering the desired security. The access control agents communicate with WebDaemon through Authentication and Authorization Protocol (AAP). Based on the user's role returned by WebDaemon, the access control agent can give the corresponding view of running or completed business processes to the particular user.

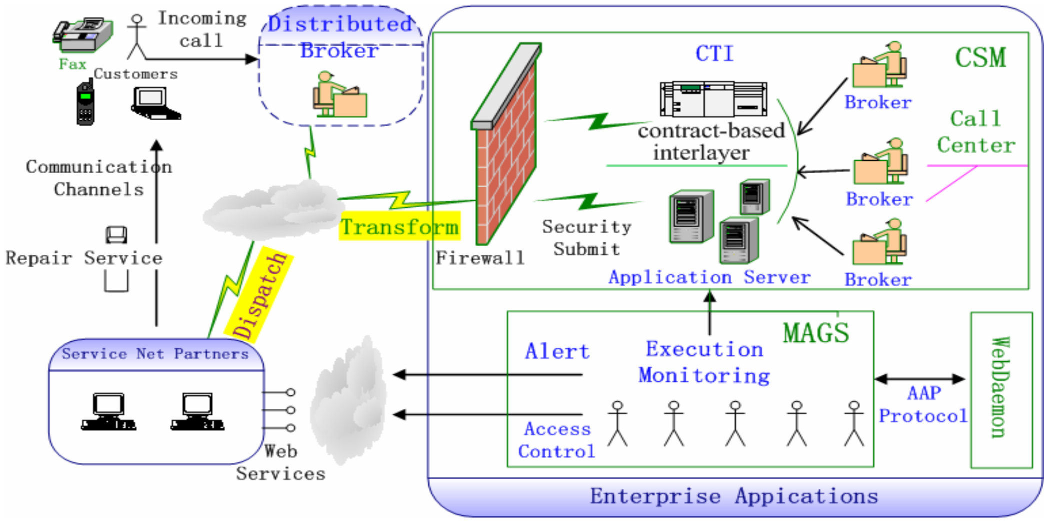
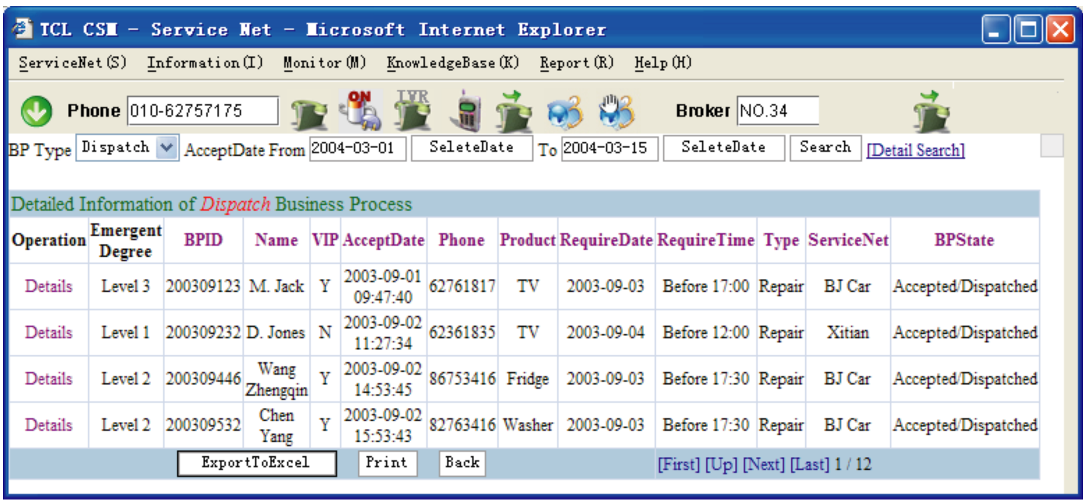


Figure 5 Overall implementations of enterprise applications



Operation	Emergent Degree	BPID	Name	VIP	AcceptDate	Phone	Product	RequireDate	RequireTime	Type	ServiceNet	BPState
Details	Level 3	200309123	M. Jack	Y	2003-09-01 09:47:40	62761817	TV	2003-09-03	Before 17:00	Repair	BJ Car	Accepted/Dispatched
Details	Level 1	200309232	D. Jones	N	2003-09-02 11:27:34	62361835	TV	2003-09-04	Before 12:00	Repair	Xitian	Accepted/Dispatched
Details	Level 2	200309446	Wang Zhengqin	Y	2003-09-02 14:53:45	86753416	Fridge	2003-09-03	Before 17:30	Repair	BJ Car	Accepted/Dispatched
Details	Level 2	200309532	Chen Yang	Y	2003-09-02 15:53:43	82763416	Washer	2003-09-03	Before 17:30	Repair	BJ Car	Accepted/Dispatched

Figure 6 Web pages of detailed Dispatch process information reported by MAGS

3.5 Access Control Agent

From the integrated MAGS architecture (Figure 4), we know that there are different kinds of requirements of business process information from inside the enterprise, e.g., supervisor, and from outside the enterprise, e.g. the business partners. There are problems if everyone can access no different information from the same process. In order to solve the security problem, the access control agents in MAGS expose interfaces to provide this function. WebDaemon [10] is a role-based access control

4. IMPLEMENTATION

The multi-agent system presented here has been successfully implemented and seamlessly integrated in a Customer Service Management (CSM) system of a company [1] (see Figure 5.). Agents act as intelligent session Enterprise Java Beans (EJB) within MAGS. Under the J2EE framework, session beans are components containing business logics associated with a

[1] TCL Group Corporation, http://www.tcl.com.

particular client session or task. MAGS runs on a J2EE application server, namely Weblogic, and adds intelligence in the sense of intention-based monitoring behaviors. CSM adopts a correlative work, called the contract-based interlayer [11], to integrate the layer of hardware-related functions (SMS, FAX) with the layer of business-related functions. With the contract-based interlayer, MAGS can send out alerts to business partners' services. MAGS cooperates with WebDaemon in providing access control functions. The statuses of the whole business processes can be monitored by web browsers based on the J2EE framework (as Figure 6 illustrates). We have got reliable and effective results in practice.

MAGS targets at providing a mechanism for business process management in a web services environment. MAGS is loosely coupled with other applications in architecture in the sense that it can be plugged into any potential systems, including e-commerce, web services and web-based applications. There is no implementary obstacle for these systems if only they hold out the specification of web services.

5. CONCLUSIONS

In this paper, MAGS, a monitoring multi-agent system for the management of business process in a web services environment, has been presented. MAGS can be integrated with applications in the enterprise and web services provided by cross-organizational business partners in architecture. Capabilities are added to BDI agent in MAGS to offer mechanisms of monitoring process execution, carrying out alerts and access control, which is described in detail with the examples of a typical business process in a customer service management system.

In the future, we will focus on developing richer monitoring capabilities and introducing MAGS into more domains, such as Semantic Web Services [12]. Semantic Web Services, in essence, is a kind of business processes. Its Semantic Markup OWL-S [13] should provide declarative descriptors for the state of execution of services. In fact, versions of OWL-S developed so far have not ventured into this area. In our view, introducing semantics into MAGS is a promising approach to make functionalities of automatic monitoring possible.

ACKNOWLEDGEMENT

This work is supported in part by the National Science Foundation of China under the contract numbers 60373002 and 60496323, and in part by TCL group corporation. The authors thank for all the members of the CSM project.

REFERENCES

[1] Woodridge M, Jennings N.R., "Intelligent Agent: Theory and Practice", The Knowledge Engineering Review, Vol.10, No.2, pp115-152, 1995.
[2] Michael Bratman, Intentions, Plans, and Practical Reason, Cambridge: Harvard University Press, 1987.
[3] P.R.Cohen, H.J.Levesque. "Intention is choice with commitment". Artificial Intelligence, Vol. 42, No.2-3, pp213~261,1990.
[4] Rao, A.S. & Georgeff, M.P. "BDI Agents: From Theory to Practice". proceeding of the First International Conference on Multi-Agent Systems (ICMAS'95), 312-319.
[5] Veloso, M., and Cox, M., "Rationale-based monitoring for planning in dynamic environments". Proceeding of the 1998 International Conference on AI Planning Systems, pp171-180. AAAI Press, Menlo Park, CA.
[6] David E. Wilkins, Thomas J. Lee and Pauline Berry, "Interactive Execution Monitoring of Agent Teams", Journal of Artificial Intelligence Vol.18, pp217-261, 2003.
[7] Kaminkaa, G., Pynadath, D., & Tambe, M. "Monitoring deployed agent teams". proceeding of Autonomous Agents'01, pp. 308-315, Montreal, Canada.
[8] The Gartner CRM Vendor Guide: 2004, http://www4.gartner.com 5 April 2004
[9] Xinyu Zhao, Zuoquan Lin, "An Agent-based Supply Chain Modeling", Computer Science(In Chinese) Vol. 8, 2004
[10] Yanfeng Gui, Zuoquan Lin, "WebDaemon: A Role-based WEB Security Access Control". System Computer Research and Development(In Chinese), Vol.8 pp20-30, 2003
[11] Wu Cen, Zhao Xinyu, Zhao Chen, Lin Zuoquan, "Contract-based Interlayer: A Two-way Approach to Integrate Call Center With J2EE Framework", International Symposium on Distributed Computing and Applications to Business, Engineering and Science (DCABES 2004).
[12] S.A. McIlraith,T.C. Son, and H. Zeng, "Semantic Web Services," IEEE Intelligent Systems, Vol. 16, No. 2, pp46–53, Mar./Apr. 2001
[13] DARPA Agent Markup Language. DAML-S/OWL-S 1.0 Draft Release. 2003. Http://www.daml.org/services/owl-s/1.0/