A Research on the Architecture of ERP for Small & Medium-Sized Enterprise Based on Agent and SOA

Ruixue Fu[1], Zhanhong Xin[2] and Jianzhang Wu[3]

[1,2]School of Economics and Management, Beijing University of Posts and Telecommunications, Beijing 100876, P.R. China brucefrx@gmail.com
Key laboratory of Information Management and Economics, MII, P.R. China
xinzhanhong@263.net
[3] School of Management Science and Engineering, Shijiazhuang University of Economics, Shijiazhuang 050031, P.R. China

Abstract. With the rapid development of information technology and the gradual extension of information technology to enterprise, ERP systems become more and more complex and some new requirements that focus on both manufacturing activities and the supply chain are brought forward. To address these problems, a MAERP system architecture based on agent and SOA has been developed. To illustrate the architecture, an experiment system has been proposed in detail. This simulation implied that the architecture provides a very efficient method to design ERP systems for small & medium-size Enterprises with the purposes of flexibility to achieve the business agility, reusability of the intelligent component, and cooperation between application systems to assure a global optimization. The objectives of this paper is to illustrate the architecture of ERP systems based on agent for small & medium-size Enterprises, and the approach of how a web services-based SOA supports our MAERP system.

Keywords: *Software architecture, Enterprise resource planning (ERP), Service-oriented architecture, Web services, Agent*

1. INTRODUCTION

ERP (Enterprise Resource Planning) systems reflect the most advanced management theory of enterprise nowadays and supply for the best strategy of CIMS [1]. It is a famous conception put forward by Gartner Group, the famous IT Analysis Company in America, in the 20 century [2]. ERP systems that developed from Material Requirement Planning integrate and automate core corporate activities from inventory control, to sales, production, and supply chain [3, 4].

ERP systems become more and more complex and some new requirements that focus on both manufacturing activities and the supply chain are brought forward. They require ERP for some advantages, such as integration, flexibility and reengineering to respond to changing business requirements and complex transaction processing to extend quickly, more informed decisions making, communication directly with suppliers and customers, etc.

For this reason, the agent has introduced as one of the solutions in future software environments. Since the agent had introduced from AI community, it has extended to various applications, such as e-mail filtering, and Air-traffic Control. Moreover, in distributed and heterogeneous environments such as Electronic Commerce applications, the concepts of agent are widely applied [5]. In the past years, researchers engaged in applying agent technology to the development of ERP systems, and presented various architectures. For examples, Bih-Ru Lea etc. introduce an architecture, called MAERP that includes four types of basic agents [6]; Ye Bin etc. propose an architecture called DIERPS [1].

Previous researches and developments pave a fundamental basis for the application of agent technologies in ERP systems. Although there are many literatures for agent technology, web services-based SOA, and ERP systems respectively, it seems that a lack of literature discusses applying agent technology and web services-based SOA to implement ERP systems for small & medium size enterprises with the purposes of scalability, extensibility, flexibility and reusability of the intelligent component. Based upon multi-agent technology, this paper proposes the architecture aimed at providing a practical solution for ERP systems. Different to the literatures mentioned before, this paper has mainly been concerned with applying agent technology and web services-based SOA to implement ERP systems for small & medium size enterprises with the purposes of scalability, extensibility, flexibility to achieve the business agility, reusability of the intelligent component, and cooperation between application systems to assure a global optimization.

The paper is organized as follows. In section 2, we introduce the concept of Agent, MAS. In section 3, we present our MAERP system architecture consists of five types of agents and web service-based architecture of agents consists of three types of agents. In section 4, we give the communication method between Agents in detail. In section 5, we introduce our simulation of the architecture we proposed in detail. Finally, in section 6 we conclude the present problems and future directions.

2. AGENT AND MAS

The concept of agent was started from John McCarthy in the mid-1950's and established by Oliver G. Selfridge several years later. In the early days, many researchers have been studied about agent in boundary of AI. Since 80's the agent has been widely applied [5]. There is a general agreement that an agent is a reusable component that exhibits a combination of the six characteristics: Autonomous, Adaptable, Mobile, Knowledgeable, Collaborative and Persistence [7, 8]. So in software system, agent means software component that has inference capability, and can interacts autonomously as a surrogate for its user with its environment and other agents to achieve the predefined goal, and reacts to changes in the environment. There are three categories of agents based on the function of agents [7, 8]: Personal agents, Mobile agents, and Collaborative agents.

Inspired by distributed artificial intelligence, a MAS (Multi-Agent System) consists of autonomous, generally heterogeneous and potentially independent agents which work together to solve special problems. As described by Brennan,

autonomous, cooperative, and scalable are the typical characteristics of a MAS that has the following capabilities [9]: Independent decision-making, Interacting with other agents and humans, Perceiving changes in their environment and acting as a consequence, and Taking initiative to reach certain objectives.

These distinctive characteristics of a MAS can facilitate ERP systems with effective means to integrate various software systems over a net work in distributed manner. It is appropriate to adopt the MAS technology in a distributed ERP systems to resolve the distributed project scheduling and management problems.

3. THE SOFTWARE ARCHITECTURE

3.1 The SOA

To achieve business agility and IT flexibility, Service-Oriented Architecture (SOA) is becoming the mainstream of system integration [10]. SOA is an architectural framework that supports integrating business tasks as linked services that can be accessed when needed over a network. It allows the user to place multiple service applications into a process or processes. Each application function needs to be transformed into services which are then assembled into processes. Figure 1 presents the relationship between service, processes and applications in SOA. Within SOA architecture, all functions, such as check service inventory, and software distribution are defined as services [11].

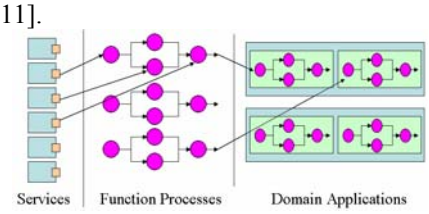

Figure 1. Three Layers of the SOA

Services are the building blocks of the system. These services have well-defined interfaces that let consumers know how to interact with them. They are grouped together to form complex, integrated processes that define the sequence in which services will be invoked. A function process provides the means for coordinating services in a specific order. The processes are then combined into domain applications that fit specific business requirements.

3.2 The MAERP System Architecture

A group of agents can form a MAS to achieve the common global goals by connecting with each other through a LAN, the Intranet or Internet. We assume there is a MAS within each functional area such as department, factory and workroom.

To achieve flexibility of system and business agility, we apply Service Oriented Architecture (SOA) concept to design the system. As indicated in Figure 2, the framework for ERP system consists of five types of agents with different functionalities which are discussed next. Figure 2 illustrates the abstract level of MAERP system architecture with coordination agents communicating with each other over the company's network [6], [12].

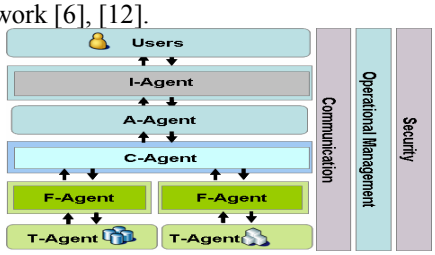

Figure 2. The MAERP System Architecture

A-Agent is used to realize an application of the whole ERP systems which will be divided into three application levels: operation level, management level and decision level. Operation level is the basis of the whole ERP system, its major function is to deal with business and collect the data. Management level is responsible for helping middle-level manager understand work condition in his department in time and make assistant strategy in tactic problem. Decision supporting level is responsible for providing high-level decision staff with information support of all kinds of data and knowledge and some best solving-problem plans. According to the above analysis, we design three types of A-Agents: Operation A-Agent, Management A-Agent and Decision A-Agent to realize the above three applications.

F-Agent is used to realize a function or business process which is the important base for operation of enterprise, to define the sequence in which T-Agents will be invoked, and to be further combined into domain applications that fit specific business requirements.

C-Agent is the heart of this MAERP architecture and is the controller of the other agents within an application. An application can have one or many C-Agent depending on the nature of task complexity. The C-Agent can communicate and collaborate with other agents, react to various requests, assign tasks to F-Agents, receive instructions and report to user through A-Agent, assign data collection to and receive data from F-Agent, assign tasks and receive feedback from F-Agents, as well as communicate with and provide request data for other C-Agents.

T-Agent usually can carry autonomously out some specific functions in terms of their own domain knowledge without the intervention of coordination agents. We look upon T-Agents as services, which are the building blocks of the MAERP and have well-defined interfaces that let other agents know how to interact with them. They are grouped together to form complex, integrated processes that define the

sequence in which services will be invoked. Some T-Agents possess the ability to collect data, perform data analysis, query specific databases within the application, retrieve information requested by F-Agent, and perform data warehousing and prepare dataset upon request from F-Agent.

I-Agent can communicate between users and MAERP system, prepare reports for users, and interpret results for users. It can monitor and inform users when tasks have been completed without the inquiry of users, learn and store preferences of users, and record the user's disposition to usage of the MAERP system.

All agents in the system have the ability to communicate through SOAP and KQML (see Agents communication). Each agent has a defined type (denoting a set of messages which are accepted and understood by the agent) and it is identified within the network by its distinct name and a given identification number [9], [4], [13].

3.3 Web Services

By using the service-oriented approach, the MAERP system will have a flexible infrastructure, which can easily adapt to user requirements. We apply web services to implement the MAERP system based on SOA. Web services provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks.

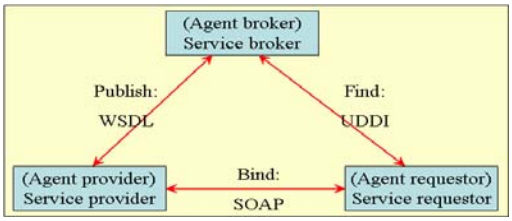

Figure 3. Web Service Based Architecture of Agents

The conceptual architecture of Web services mainly includes three roles and three operations as Figure 3 shows. Each service provider can publish service descriptions to the service broker, which is the description of service interface and of implementation details with WSDL (Web Service Description Language) including data types and operations of Web service, binding and invoking information and its location. To catalog service descriptions, we use a standard registry definition called Universal Discovery, Description and Integration (UDDI). It registries contain white pages for each registered provider. When a service requester needs a service, it will query and find the suitable service in the catalog. After it acquires the suitable service's information, the requester will directly bind and access the service [10].

Web services rely on the functionalities: publish, find, and bind. The equivalent agent-based functionalities depicted in Figure 3 are shown in parentheses, and all interactions are via an agent-communication language that we use KQML. The Web service based architecture of agents consists of three types of agent [14], [15].

Agent provider publishes service descriptions to the Agent broker, asks Agent broker to register their capabilities and physical areas, and provides other agents with the service. Agent requester asks Agent broker to discover the information about Agent provider and bind to the Agent provider to obtain the service. Agent broker registers yellow pages for each registered Agent provider and plays match-maker between Agent providers and Agent requesters by considering their locations, capabilities and requirements.

4. AGENTS COMMUNICATION

To express communication and negotiation required and organize communications between agents, we use the KQML. The basic assumption is that agents can be located on separate machines or at least separate processes within one machine. Hence the communication between them must be constituted by some kind of network protocol-in our case it is TCP/IP. But this protocol represents only the lowest layers of the communication which is extended by five other protocols on the top two levels of the OSI model-they are SOAP, WSDL, UDDI, KQML and the content language itself [10, 3, 14], as shown in Figure 4.

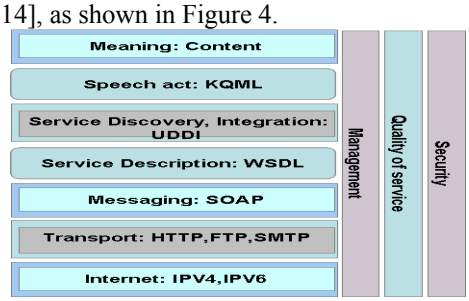

Figure 4. The Communication Layers

The simple object access protocol (SOAP) provides the common protocol systems need to communicate with each other so that they can request services, such as to schedule appointments, order parts, and deliver information. The Web Services Description Language (WSDL) describes the agents in a machine-readable form, where the names of functions, their required parameters, and their results can be specified. Finally, Universal Description, Discovery, and Integration (UDDI) gives Agent requestor a way to find needed agents by specifying a registry or "yellow pages" of agents.

As shown in Figure 4, the inter-agent message-transporting layer is constituted by SOAP [10, 15, 3]. Above UDDI layer which facilitates the connection between agents, there is a KQML layer. The Knowledge Query and Manipulation Language (KQML) language defined by KSE (Knowledge Share Effort) is based on the linguistic theory of the speech act. The KQML is a language and protocol exchanging information and sharing knowledge, which provides basic format of expressing and processing messages and supports sharing information among agents.

So we design architecture of communication based on KQML and SOAP, as shown in Figure 5. We realize knowledge expression and cooperation requests among agents with KQML. Agent sends message to Communication Switch Agent (CSA), CSA wraps the message of KQML to the access format of SOAP message. We realize communication and information exchange between agents (based on KQML and SOAP) by means of Communication Switch Agent [15]. The communication among the agents that used in our architecture is described as follows: The sender agent sends a KQML request message to CSA, the CSA converts the semantics of KQML to the access format of SOAP message, and the CSA sends the message to the receiver agents in terms of the same method over transport network [15, 3].

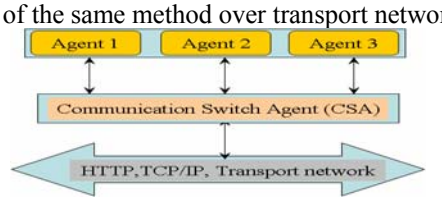

Figure 5. The Architecture of Agent Communication Based on SOAP

5. SIMULATION STUDY

We apply the architecture we proposed to an experiment system as Figure 6 shows to estimate the architecture. We assume that a user in a planning department needs to make the production planning. We assume for simplicity that there are six functions or business processes: sales, transport, planning, materials, purchasing, and distribution centers in the XX Company. Each function has its own information system, database, and data architecture [6, 16].

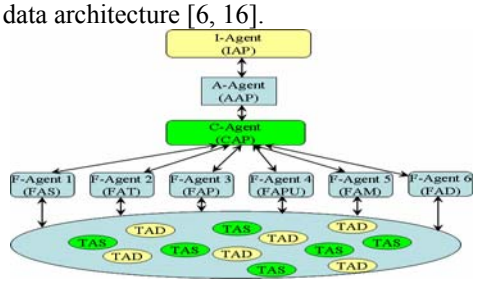

Figure 6. The Architecture of Simulation System

We assume that one Operation A-Agent is assigned to the planning department (i.e., AAP), one C-Agent (i.e., CAP), and one I-Agent (i.e., IAP). We further assume that one function agent is assigned to the sales (i.e., FAS), transport (i.e., FAT), materials (i.e., FAM), purchasing (i.e., FAPU), planning (i.e., FAP), and distribution centers (i.e., FAD). Further, for each function agent, one data collection agent (TAD),

and several tasks agents (TAS) [17, 6, 18]. Figure 6 provides an overview of MAS which will be used to answer the above production planning in six steps as follows.

In the planning application, IAP communicates the question about production planning to AAP. And then AAM starts to deal with this question as follows:We realize the production planning through lists of goals for this week and some future weeks. These plans go upstream through the internal supply chain, and come back downstream as plans of delivery. On the way upstream each agent contributes with its own knowledge. To explain the conversation plans and rules, we begin with looking at the issuing of demand-forecasts, which give the expected number of units ordered for this or coming weeks and start production planning.

Step 1:

By exercising domain knowledge, FAS organizes four tasks concurrently by means of C-Agent CAM. Create a demand-forecast-conversation; Compute the demand-forecast by means of T-Agent; Prepare the data for sending, and send the message to the distribution center agent (FAD); and Monitor the status of requested information from various agents.

Step 2:

When FAD receive the demand-forecast message form FAS, the FAD create a demand-plan-conversation which use knowledge of the DC's inventory levels. DC-demand-plans, which define the targeted quantity of each product arriving at the DC at the end of this and coming weeks, are made and send to the Transport agent which creates a corresponding conversation.

Step 3:

Transport agent (FAT) knows how much is expense to the DC and make ship-plans, which define the quantity of each product that should be shipped from a plant to a given DC at the end of this week and coming weeks. The ship-plans are sent to the planning agents of the plants concerned.

Step 4:

The aim of a plant's planning agent (FAP) is to convert the incoming ship-plan (if it has external customers) and materials-demand-plans from the next downstream plants (if it has internal customers) to the plant's own materials-demand-plans that define the number of units of a given product the plant needs this week and coming weeks for all internally supplied parts. To calculate the materials-demand-plans the Planning agent will use data from the other agents in the plant. These plans are sent to the next agent upstream.

Step 5:

FAP will make delivery-plan that defines the number of units the plant will deliver this week and coming weeks for each customer. The delivery-plan includes the total demand and is limited by part availabilities and production capacities. And the planning agent (FAP) decides the actual-production-plan of the plant according to domain knowledge, which is the production goals for this and coming weeks.

Step 6:

From the actual-production-plan, the materials agent can calculate a materials-order-plan for parts. The plans are sent to the purchasing agent, in which they are converted to part orders for the suppliers. When receiving acknowledgment messages from the supplier, FAM update their order data base and inventory data base.

Upon notification of FAM, AAP will inform the user through IAP about the new plan. IAP will also record the user's decision, which will be used to predict the preference in the future.

6. CONCLUSIONS

With the assumption that there is a MAS in each functional area such as department and factory, one ERP architecture is established to meet some new requirements with five types of agent, and present Web service based architecture of agents that is based on three types of agents.

To estimate the architecture, we propose an experiment system. This simulation implies that the architecture provides a very efficient method to design and implement ERP systems for small & medium-size Enterprises to assure a global optimization, flexibility of system, business agility, and the reusable ability of sub-systems or legacy systems in distributed and heterogeneous environments. However, there are many limitations of this paper. One of them is that we have not provided monitoring mechanism to supervise the process when agents communicate with each other. Secondly, the security issues are not resolved.

ACKNOWLEDGEMENTS

This work is supported by Key Laboratory of Information Management and Information Economics, MII, P.R.C, Grant No. F0607-35.

REFERENCES

1. B. Ye, Z. Ma, and X. Tu, Research on the Architecture of ERP System Based on Intelligent Autonomous Decentralized System, in *Proc. of The ISADS 2005 Proceedings* (2005), pp.616-619,
2. Q. Chen, *ERP-Step forward from Internal Integration* (Publishing House of Electronics Industry: Beijing, 2005).
3. C. Wu and Z. Gong, A Study of Web Service and Agent Technology, *Microprocessors*. Volume 4, pp.28-32, (2006).
4. N. Gibson, C.P. Holland, and B. Light, Enterprise Resource Planning: a Business Approach to Systems Development, in *Proc. of the 32nd Annual Hawaii International Conference, Volume 7* (1999), pp.9-13.
5. M. Kim, S. Lee, I. Park, J. Kim, and S. Park, Agent-Oriented Software Modeling, in *The Proc. of Software Engineering Conference (APSEC) Sixth Asia Pacific* (1999), pp.318-325.
6. B. Lea, M.C. Gupta, and W. Yu, A Prototype Multi-Agent ERP System: An Integrated Architecture and a Conceptual Framework, *Technovation*. Volume 25, Number 4, pp.433-441, (2005).

7. G. Pour, Integrating agent-oriented enterprise software engineering into software engineering curriculum, *Frontiers in Education*. Volume 3, pp.8-12, (2002).
8. M.L. Griss and G. Pour, Accelerating Development with Agent Components, *Computer*. Volume 34, Number 5, pp.37-43, (2001).
9. S. Wu and D. Kotak, Agent-Based Collaborative Project Management System for Distributed Manufacturing Systems, in *Proc. of The Man and Cybernetics of IEEE International Conference, Volume 2* (2003), pp.1223-1228.
10. F. Liu, L. Yao, W. Zhang, H. Liu, and H. Zhang, A Conceptual Model of Agent Mediated Web Service, in *Proc. of The Services Computing Proceedings of IEEE International Conference* (2004), pp.638-642.
11. I. Chen and C. Huang, An SOA-Based Software Development Management System, in *Proc. of The Web Intelligence of IEEE/WIC/ACM International Conference* (2006), pp.617-620.
12. Y. Huang, J. Zhang, Q. Zhang, and S. Wang, Intelligent Resource Planning of Testing Lab Based on CORBA and Multi-agent, in *Proc. of Machine Learning and Cybernetics*, *2002 International Conference, Volume 1* (2002), pp.492-495.
13. M.L. Griss and G. Pour, Accelerating Development with Agent Components, *Computer*. Volume 34, Number 5, pp.37-43, (2001).
14. M.N. Huhns, Agents as Web services, *The Internet Computing (IEEE)*. Volume 6, Number 4, pp.93-95, (2002).
15. A. Sashima, N. Izumi, and K. Kurumatani, Location-Mediated Coordination of Web Services in Ubiquitous Computing, in *Proc. of The Web Services Proceedings of IEEE International Conference* (2004), pp.22-823.
16. Z. Xu and H. Wang, Research on Service Selection-Oriented Web Service Architecture Based on Agent, *Computer Technology and Development*. Volume 16, Number 9, pp.59-61, (2006).
17. M. Barbuceanu, R. Teigen, and M.S Fox, Agent Based Design and Simulation of Supply Chain Systems, in *Proc. of The Enabling Technologies: Infrastructure for Collaborative Enterprises of Proceedings Sixth IEEE workshops* (1997), pp.36-41.
18. X. Liu, Y. Sun, Y. Hao, and J. Xu, Research on Group-Oriented Enterprises Resource Planning: A Solution to Multiregional, Heterogeneous and Distributed Group Enterprises Application, in *Proc. of the Intelligent Control and Automation of WCICA, Volume 2* (2006), pp.7186-7190.