

# Agent-Community-Network-based Business Matching and Collaboration Support System

## (Short Paper)

Tsunenori Mine  
Faculty of ISEE, Kyushu  
University  
744 Motoooka, Nishi-ku,  
Fukuoka 819-0395 Japan  
mine@is.kyushu-u.ac.jp

Kosaku Kimura  
Graduate School of ISEE,  
Kyushu University  
744 Motoooka, Nishi-ku,  
Fukuoka 819-0395 Japan  
kimura@al.is.kyushu-  
u.ac.jp

Satoshi Amamiya  
Faculty of ISEE, Kyushu  
University  
744 Motoooka, Nishi-ku,  
Fukuoka 819-0395 Japan  
roger@al.is.kyushu-  
u.ac.jp

Ken'ichi Takahashi  
Institute of Systems &  
Information  
Technologies/KYUSHU  
2-1-22 Momochihama,  
Sawara-ku, Fukuoka  
814-0001 Japan  
takahashi@isit.or.jp

Makoto Amamiya  
Faculty of ISEE, Kyushu  
University, 744 Motoooka,  
Nishi-ku, Fukuoka 819-0395  
Japan,  
amamiya@is.kyushu-  
u.ac.jp

## ABSTRACT

Business matching and collaboration support systems are useful, in particular for small-and-medium sized companies. Most of them developed so far are based on the server-client architecture and provide their services on Web servers. They require special administrative facilities, ask users to upload their data for matching between business needs and seeds, and leave to themselves peer-to-peer communication or negotiation between matched companies. Considering these problems, we have been developing an agent-community-network-based business matching and collaboration support system. Our system requires neither any special administrative facilities nor uploading user data to a special server. It furthermore supports secure peer-to-peer communication between users. It is implemented with multi-agent Kodama framework.

## Categories and Subject Descriptors

I.2.11 [Distribute Artificial Intelligence]: Multiagent systems; J.7 [Computer Applications]: Computers in Other Systems—*Consumer products*

## General Terms

Security

## Keywords

Multi-Agents, Peer-to-Peer, Community

**Cite as:** Agent-Community-Network-based Business Matching and Collaboration Support System (Short Paper), T. Mine, K. Kimura, S. Amamiya, K. Takahashi, and M. Amamiya, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)- Industry and Applications Track*, Berger, Burg, Nishiyama (eds.), May, 12-16., 2008, Estoril, Portugal, pp.75-78.

Copyright © 2008, International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

## 1. INTRODUCTION

Nowadays, various sorts of matching systems such as business matching or human resource matching are getting popular and actively used. Since they are developed as Web-server-based systems, flexible access and useful interface are available. However, they have the following problems :

- (1) Special administrative facilities are required for the system administrators.
- (2) The users have to upload their data when updating it. The upload costs are neither cheap nor negligible.
- (3) As the number of users accessing a system increase, the load taken by the system becomes heavier.
- (4) Administrators of the system can easily eavesdrop the data stored or communications exchanged on it.

Considering these problems, we believe that the business matching and collaboration support system should be managed by a user him/herself so that his/her information used for matching and negotiation can be exchanged only with his/her negotiation partners. The systems accordingly should be constructed as distributed ones and support make one-to-one and one-to-many communication with one another. These problems are wedded to security issues. When making one-to-one or one-to-many communication by e-mail, we face lots of problems such as SPAM, phishing mail attack for public or targeted attack. In addition, since current e-mail systems are ever based on the server-client model, a server administrator can easily eavesdrop e-mail messages on the server. The appearance of this kind of attack makes an appeal of necessity of secure systems that support secure peer-to-peer communication between users.

Another issue is to create a community where users with the common interests or aims stay together and efficiently

communicate on the topics related to the community. The community can facilitate the secure exchange of privacy-related information by restricting communications within a community. Therefore business matching and collaboration support system should be constructed as a community-based system and be able to support secure peer-to-peer communication in and between communities. Such communities should have a hierarchical structure so that they can represent real societies. There is a work[1] which discusses a method constructing an e-market place based on a peer-to-peer network by combining some of existing technologies and methods, in particular, a method for realizing peer-to-peer communication based on technologies of JXTA Project<sup>1</sup>. However they still have an issue to realize one-to-many communications and do not discuss the problems for constructing a practical system, which are the main subject of this paper.

This paper presents Agent-Community-Network-based Business Matching and Collaboration Support System. We aim to develop an agent-oriented technique that is useful to realize a ubiquitous computing environment at a low cost. For our purpose, we are developing a business matching and collaboration support system for small-and-medium sized companies as a case study. Our target users are those who do not have enough knowledge on a mechanism of information exchanging between computer systems. We would like our system to enable the users get information they need without being aware of security mechanisms, places of information providers and mechanisms of exchanging information, and enjoy various services that are helpful for their business activities. Although there are several choices to meet these requirements, we choose a multi-agent-based system, in particular, Multi-agent Kodama framework because it originally supports a hierarchical community structure, and has flexible characteristics and configuration options. Moreover, Kodama separates agent name retrieval and agent physical address retrieval, which suggests availability of an efficient retrieval technique for the former retrieval according to an application of the system and an efficient node-lookup or message delivering technique for the latter retrieval. We currently adopt the agent-community-based peer-to-peer information retrieval (ACP2P) method [3] as the former retrieval method and the node lookup and routing method based on the Ordered-Tree-with-Tuft (OTT) shaped overlay network[2] as the latter retrieval. The system offers the following functions:

- Secure, robust and efficient one-to-one and one-to-many communication environment
- Automatic and semi-automatic matching between business supplier and demander
- Creating man-to-man and business-to-business communication networks based on business trading history

## 2. CHARACTERISTICS OF SYSTEM

### 2.1 Multi-Agent Kodama

The system is implemented with Multi-Agent Kodama (Kyushu university Open & Distributed Autonomous Multi-Agent) framework [5]. Kodama comprises hierarchical structured agent communities based on a portal-agent model. A

<sup>1</sup><http://www.jxta.org/>

portal agent (PA) is the representative of all member agents in a community and allows the community to be treated as one normal agent outside the community. A PA has its role limited in a community, and the PA itself may be managed as an agent by another higher-level portal agent. A PA manages all member agents in a community and can multicast a message to them. Any member agent in a community can ask the PA to multicast its message. All agents form a logical world which is completely separated from the physical world consisting of agent host machines. That means agents are not network-aware, but are organized and located by their places in the logical world. This model is realized with the agent middle-ware called Agent Communication Zone (ACZ for short). ACZ is primarily designed to act as a bridge between distributed physical networks, creating an agent-friendly communication infrastructure on which agents can much easily and freely be organized in a hierarchical fashion. One or more Kodama agents can act on one ACZ. ACZ is also designed to realize a peer-to-peer communication between agents. A Kodama agent consists of a kernel unit and an application unit. The kernel unit comprises the common basic modules shared by all Kodama agents. The application unit comprises a set of plug-in modules, each of which is used for describing and realizing a specialized or original function of agents. As described later, security, GUI and matching modules are plugged-into the application unit.

### 2.2 Security Functions

Even in a community, secure communication is indispensable because the community usually consists of other companies. The main characteristics of our security functions are as follows:

1. **Examination of applicants' qualifications and making a contract:** when joining a community, examination of applicants' qualifications is done by the representative of a community, which would be a community facility. User (applicant) registration will be done by making a contract with the facility. After its registration, the user will receive a pair of a login ID and a password to join the community. Both the login ID and the password are issued to an authority agent of the community which is called a portal agent. The portal agent authenticates them and permits the enrollment to the community. At that time, the portal agent gives the authenticated agent both a unique name of the community and a group key, which is based on the Symmetric-Key Cryptosystem (SKC). The authentication is performed according to the Public-Key Cryptosystem (PKC).
2. **Encrypted communication in a community :** in the community, two communications, one-to-one (or peer-to-peer) and one-to-many communications, are available. For the peer-to-peer communication, a session key based on the SKC is used. The key is generated by exchanging random numbers between two peers. Exchanging the random numbers is done based on the PKC. For one-to-many communication, a group key given by a portal agent is used. The group key is updated when any agent joins or leaves the community. Although a community is a closed environment, all the members do not always have collaborative relationships with one another, rather they might have

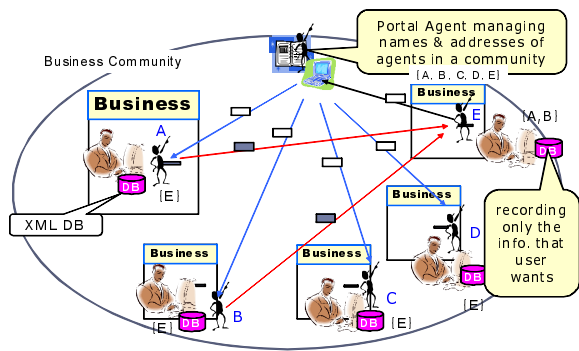


Figure 1: Matching between a query and an acceptance condition in a business community

competitive relationships. Even if all the community members belong to a common group sharing the benefits such as the same company, there would be a distinction of information according to the organization hierarchy of the group<sup>2</sup>. Encrypted communication in the community is consequently indispensable. The system does not employ anonymity of users because a target of the system is a business market. Therefore the company names or user names are attached on the query and responses. However users can keep other information private if they want.

Although each of the characteristics described above is not new, their combination is important for business matching and collaboration on this system. The detailed discussion of the security mechanisms adopted by the system is described in [4].

### 2.3 Matching Functions

When a user makes a business question, his/her agent will deliver it to the agents in the communities specified in the question. The agent that received the question checks it up with its acceptance condition created by its user. If there is one or more matched items in it, the agent creates and returns an answer to the query-sender agent.

With this matching, even though a user does not know the addresses of agents related to the user's query, the user can, by this matching function, get a list of their addresses in the communities specified in the query. This would be considered as a kind of "know-who".

In addition, receiving reply-messages from matched agents, a user can seamlessly continue to send another message to the users of the agents. In Fig. 1, agent E receives reply-messages from agent A and B. So, the user of agent E can continue to send another query to the user of agent A or agent B. On the other hand, in conventional systems, getting a set of agent addresses and sending messages by e-mail or something are separated. Actually, many people feel self-conscious about sending an e-mail message to a user whom they have no personal acquaintance with.

## 3. EXPERIMENTAL SYSTEM

<sup>2</sup>This would strongly depend on a community structure.

### 3.1 Overview

Our system is compact, easy to be moved with a USB memory and works on any computer system with a Java VM environment. Since a lot of USB memories nowadays supports an encryption function, any user can use it with ease. Figure 2 depicts the menu window of a user's agent system, which one of GUI modules. They have 4 menu buttons which are for making an acceptance condition, making a business question, showing a list of messages returned to business questions issued by a user and showing a list of business questions received.

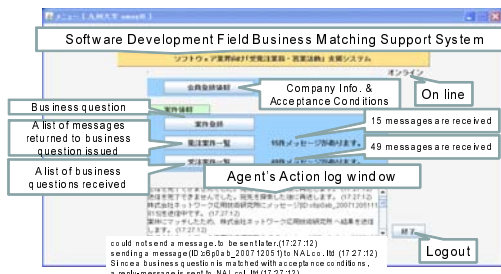


Figure 2: Menu Window

### 3.2 Internal Structure of the System

The system is composed of multi-agent Kodama and its applications. All applications such as GUI module, security module, matching module are plugged-into as Kodama application units. They are completely separated with the Kodama kernel unit. Every data, which is a Java object, is passed between modules through their interface. The security functions are defined as a set of Java classes so that it can flexibly be updated. A demander's business question and a supplier's acceptance condition are defined as a CSV-like template. They can easily be exchanged according to the business category of a company. The reason why we adopt a template-based query and an acceptance condition is that they would easily be described by anyone who are not good at operating computers. According to our interviews to some company people, our decisions were empirically acceptable. Since the system is composed of several modules that are completely separated and independent, we can individually develop each module. That means all we need is to aware the interface between the modules. This is one of the most important benefits of multi-agent-based software engineering.

The business question template is almost the same as the acceptance condition template except for a few items due to the difference of the viewpoint of their positions. These exceptions are coped with by a mapping table. The template represents a kind of small ontology of the business category. It consists of two types of items : the items independent and dependent to the category. The items independent to the category can be considered as upper ontology of the business field, which are business category, smaller business category, subject of order, detailed description of order, purpose of order, level of fixedness of development specification, start and end of development term, and certified standards such as ISO9001. A set of items dependent to the business category is domain ontology. In the case of a semiconductor

development field, they are “process”, “wafer size”, “product”, “package”, “design”, “assembly”, “test engineering” and so forth. In the case of a software development field, they are operating system, programming language, database, level of knowledge on category of business, work place, type of engagement such as package contract or detached service, development size, availability of subcontract and so on.

### 3.3 Routing

#### 3.3.1 Routing between Agents

When a query is issued to all the members of a community, the query is kept by the portal agent of the community during a given time interval so that any agent can get it even though it is not joining the community at that time. This function is also useful for agents that newly join the community and want to get such business questions that have already been delivered. When a message is directly sent to a target agent which is not logged in at that time, the message is kept by the sender agent. Is it why? If the sender agent asks a portal agent to keep and send the message to the target agent, the message will surely be reached to the target agent because a machine the portal agent resides will not be powered off and the portal agent can know when the target agent logs in. In addition, the message will not be decrypted by the portal agent because the message was encrypted by a peer-to-peer key only which the sender agent or the target agent has. However, the message may include a heavy attached files. As the number of such the messages increases, the load of the portal agent becomes heavier. It consequently causes the same problem of a client-server architecture. On the other hand, it is true that the sender agent can not know when the target agent logs in. In our current solution, the sender agent asks a portal agent to tell when the target agent logs in, although the computer the sender agent resides should not be powered off until the target agent logs in. This is trade off. In order to let a user know whether or not a message the user issues is reached to a target user (agent) or a message the user receives is matched with the acceptance condition of the user, every action an agent does is presented on the display of the action log window of a menu window shown in Fig.2.

#### 3.3.2 Routing between ACZs

We assume that a lot of users of our system use their computers on their private network and connect to the Internet via a NAT (Network Address Translation) or NAPT (Network Address Port Translation) system. Since our system provides one-to-one communication, we have to support the routing via the NAT or NAPT. When an agent is starting to join a community, the ACZ where the portal agent of the community resides keeps the connection with the ACZ where the community member agent resides. Thus, the ACZ of the community portal agent can make a communication with the the ACZs of community member agents. When another agent joins the community, the ACZ of the portal agent also keeps the connection with the ACZ of the agent. This constructs star-shaped connections between the ACZ of the portal agent and the ACZs of other member agents. Although this star-shaped routing may not be scalable, we made sure by empirical experiments using a cluster machine with 32 Personal Computers (PCs) that it could support 1000 agents. Supporting 1000 agents is enough for

our current objectives because the current targets are closed communities, each of where less than 1000 members belong to.

On these experiments, we also made sure that our systems are robust when agents of some PCs suddenly join and leave the network by powering on/off, getting started running or falling to the sleep mode by opening or shutting the display of a note PC the agents reside, and inserting/pulling their network cables to/from the computers.

### 4. CONCLUSION AND FUTURE WORK

This paper discussed Agent-Community-Network-based Business Matching and Collaboration Support System. We have just finished a beta test of our system. It suggests that this system is useful for determining business companies to be contacted. Without meeting contact persons of a target business company nor talking to them by face to face, business people can not give credit to the company. After meeting them, however, the system can also be used for exchanging important information to be led to making a contract. The system is currently used by a personnel-service business company as a trial.

Our system has many capabilities to be applied to various kinds of fields by changing the matching templates. In addition, not only business matching, but also distributed SNS services or dynamic mailing list services will be supported by the mechanism described in this paper. However we have a lot of things to do for modifying and improving the system to realize these things. One of our current concerns is to scale up the system. Another one is to improve the accuracy of matching between a business question and an acceptance condition although it requires a lot of real trials.

### Acknowledgment

This research was partly supported by the Strategic Information and Communications R&D Promotion Program under grant 052310008 from the MIC, Japan.

### 5. REFERENCES

- [1] D. R. Ferreira and J. J. P. Ferreira. Building an e-marketplace on a peer-to-peer infrastructure. *International Journal of Computer Integrated Manufacturing*, 17(3):254–264, 2004.
- [2] K. Kimura, S. Amamiya, T. Mine, and M. Amamiya. A new infrastructure construction method for building multi-agent-based system (in japanese). *The IEICE Transactions (Japanese Edition)*, J90(9):2388–2397, 9 2007.
- [3] T. Mine, A. Kogo, and M. Amamiya. Agent-community-based peer-to-peer information retrieval and its evaluation. *Systems and Computers in Japan*, 37(13):1–10, 11 2006.
- [4] K. Takahashi1, Y. Mitsuyuki, T. Mine, K. Sakurai, and M. Amamiya. Design and implementation of security mechanisms for a hierarchical community-based multi-agent system. In *the 10th Pacific Rim International Workshop on Multi-Agents*, 11 2007.
- [5] G. Zhong, S. Amamiya, K. Takahashi, T. Mine, and M. Amamiya. The design and application of kodama system. *IEICE Transactions INF.& SYST.*, E85-D(04):637–646, 4 2002.