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

Tsunenori Mine[1], Kousaku Kimura[2], Satoshi Amamiya[1], Ken'ichi Takahashi[3], and Makoto Amamiya[1]

{Faculty[2], Graduate School[1]} of ISEE., Kyushu University
744 Motooka, Nishiku, Fukuoka 819-0395, Japan
{mine,kimura,roger,amamiya}@al.is.kyushu-u.ac.jp
Institute of Systems & Information Technologies/KYUSHU[3], 2-1-22 Momochihama, Sawara-ku, Fukuoka 814-0001, Japan
takahashi@isit.or.jp

Abstract. Business matching and collaboration support systems are useful, in particular for small-and-medium 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.

1 Introduction

Nowadays, various sorts of matching systems such as business matching(e.g. [9, 10]) or human resource matching(e.g. [7]) 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 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[1]. 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.

In this paper, we present 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 [5] as the former retrieval method and the node lookup and routing method based on the Ordered-Tree-with-Tuft (OTT) shaped overlay network as the latter retrieval[4, 3].

[1] http://www.jxta.org/

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

The system is being implemented with Multi-agent Kodama framework[11]. Kodama provides user community creation function, and secure and robust one-to-one and one-to-many communication in and between communities. Matching between supplier and demander is carried out by peer-to-peer communication between Kodama agents.

The rest of the paper is organized as follows. Section 2 describes the characteristics of the system and Section 3 discusses an experimental system. Section 4 concludes and describes our future work.

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 [11]. Kodama comprises hierarchical structured agent communities based on a portal-agent model. A 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 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 one 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 P2P key which is a session key based on SKC is used. The key is generated by exchanging random numbers between two peers. Exchanging the random numbers is done based on 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. Fig. 1 depicts one-to-many communication with the group key in the left-hand side, and peer-to-peer communication with the P2P key in the right-hand side.

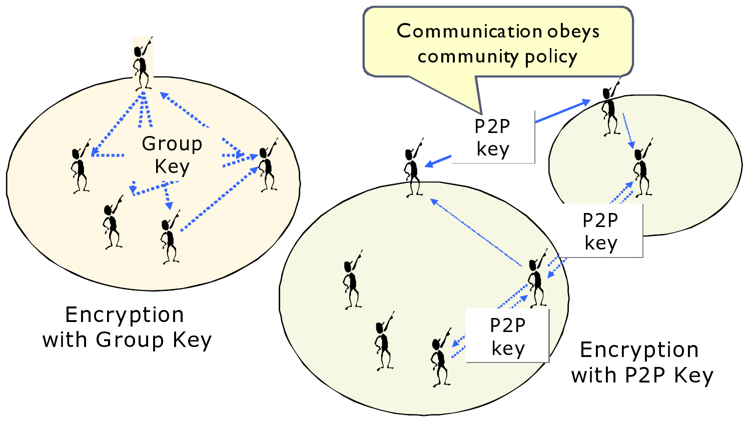

Communication obeys community policy
Group Key
P2P key
P2P key
P2P key
Encryption with Group Key
Encryption with P2P Key

Fig. 1. Communication with Group Key(left) and P2P Key(right)

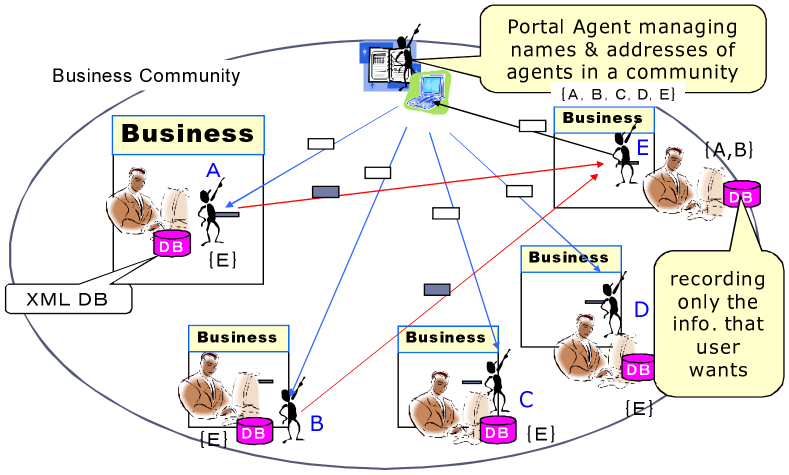


Fig. 2. Matching between a query and an acceptance condition in a business community

Although a community is a closed environment, all the members do not always have collaborative relationships with one another, rather they might have 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[2]. 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.

Fig. 1 shows communication images with the group key and the P2P key. Although each of the characteristics 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 [8].

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

[2] This would strongly depend on a community structure.

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. 2, 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.

2.4 Look-up of Delivery Address from Business Trading History

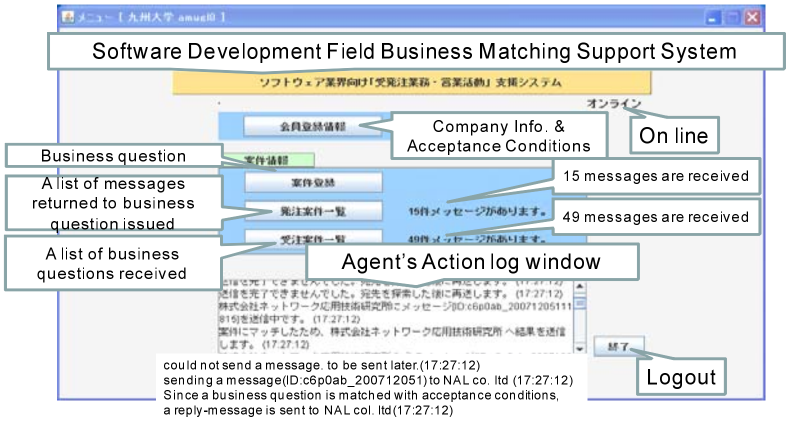
Business dealings generally prefer known and reliable clients to new ones. An agent in our system accordingly keeps all its trading records in its XML database so that it can easily search for the clients and their detail of the transaction. The information registered in the database is a set of pairs of business query and its answer. When a business query is created, business transaction relevant to it will be searched from the database with XQuery consisting of a business item name and content in the query. The searched results are displayed in a ranking list according to their score calculated for the relevancy. Users will choose addresses for delivery in the list and send the query to the addresses. If the users can not find any address that they want to send or if they want to find new clients that are not in the history, they will issue the query to all the members of the communities specified in the query. According to the query and answer operations using the trading history, we believe that business to business or person to person relation networks will gradually be created. This basic idea is based on the knowledge of the ACP2P method [6].

3 Experimental System

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 nowerdays supports an encryption function, any user can use it with ease. Fig. 3 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.

Fig. 4 depicts a list of issued business questions (top), a condition matched between a business question and an acceptance condition (middle), and an inquiry window (bottom). With the inquiry window, a user that issued the business question can make free communications related to the matched condition with the user that returned the message. Both users can send a message with attached


Software Development Field Business Matching Support System
ソフトウェア業界向け「受発注業務・営業活動」支援システム
オンライン
On line
会員登録情報
Company Info. & Acceptance Conditions
Business question
案件情報
案件登録
A list of messages returned to business question issued
発注案件一覧
15件メッセージがあります。
15 messages are received
受注案件一覧
49件メッセージがあります。
49 messages are received
A list of business questions received
Agent's Action log window
終了
Logout

```
送信を完了できませんでした。宛先を探索した後に再送します。 (17:27:12)
株式会社ネットワーク応用技術研究所にメッセージ(ID:c6p0ab_20071205111
815)を送信中です。 (17:27:12)
案件にマッチしたため、株式会社ネットワーク応用技術研究所 へ結果を送信
します。 (17:27:12)
```

```
could not send a message. to be sent later.(17:27:12)
sending a message(ID:c6p0ab_200712051) to NAL co. ltd (17:27:12)
Since a business question is matched with acceptance conditions,
a reply-message is sent to NAL col. ltd(17:27:12)
```

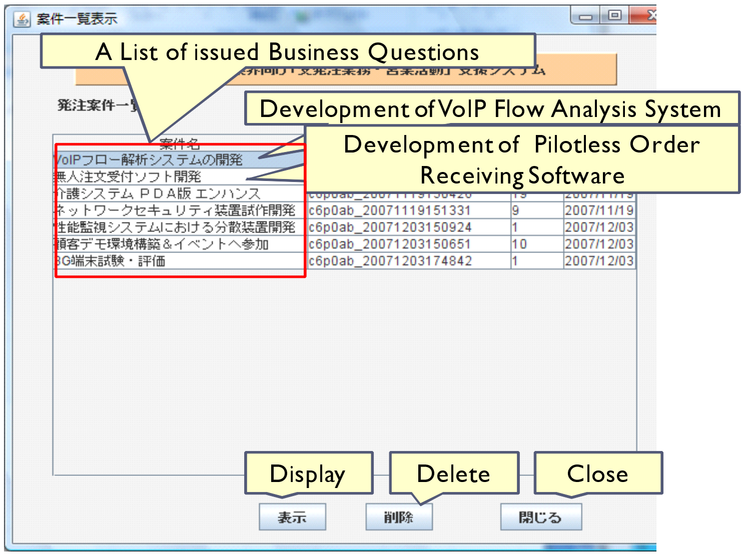
Fig. 3. Menu Window

files. As well as the packet transmission system, the attached files are splitted into some numbers of small-sized objects, and issued to a target agent. Thus this does not cause trouble even if we take a star-shaped network toppology to be mentioned later in Section 3.3.

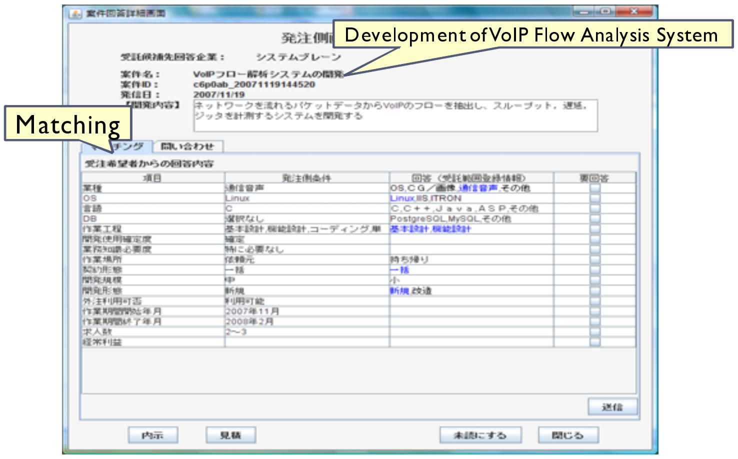
3.2 Internal Structure of the System

The structure of the system is shown in Fig. 5. 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.


案件一覧表示
A List of issued Business Questions
発注案件一覧
Development of VoIP Flow Analysis System
Development of Pilotless Order Receiving Software
Display
Delete
Close
表示
削除
閉じる

案件名			
VoIPフロー解析システムの開発			
無人注文受付ソフト開発			
介護システム PDA版 エンハンス	[illegible]	[illegible]	[illegible]
ネットワークセキュリティ装置試作開発	c6p0ab_20071119151331	9	2007/11/19
性能監視システムにおける分散装置開発	c6p0ab_20071203150924	1	2007/12/03
顧客デモ環境構築＆イベントへ参加	c6p0ab_20071203150651	10	2007/12/03
3G端末試験・評価	c6p0ab_20071203174842	1	2007/12/03

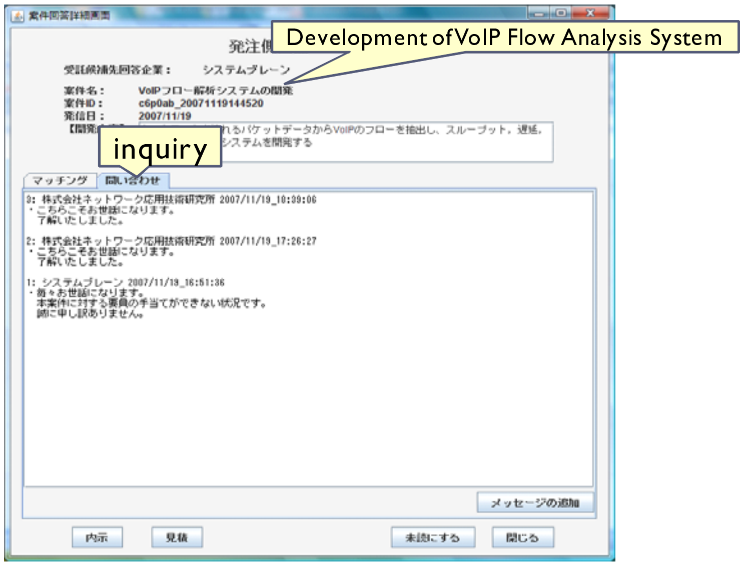


受託候補先回答企業： システムブレーン

案件名： VoIPフロー解析システムの開発
案件ID： c6p0ab_20071119144520
発信日： 2007/11/19
【開発内容】 ネットワークを流れるパケットデータからVoIPのフローを抽出し、スループット，遅延，ジッタを計測するシステムを開発する

受注希望者からの回答内容

項目	発注側条件	回答（受託範囲登録情報）	要回答
業種	通信音声	OS,CG／画像,通信音声,その他	☐
OS	Linux	Linux,IIS,ITRON	☐
言語	C	C,C++,Java,ASP,その他	☐
DB	選択なし	PostgreSQL,MySQL,その他	☐
作業工程	基本設計,機能設計,コーディング,単	基本設計,機能設計	☐
開発使用確定度	確定		☐
業務知識必要度	特に必要なし		☐
作業場所	依頼元	持ち帰り	☐
契約形態	一括	一括	☐
開発規模	中	小	☐
開発形態	新規	新規,改造	☐
外注利用可否	利用可能		☐
作業期間開始年月	2007年11月		☐
作業期間終了年月	2008年2月		☐
求人数	2〜3		☐
経常利益			☐



受託候補先回答企業： システムブレーン

案件名： VoIPフロー解析システムの開発
案件ID： c6p0ab_20071119144520
発信日： 2007/11/19
【開発内容】 れるパケットデータからVoIPのフローを抽出し、スループット，遅延，システムを開発する

```
3: 株式会社ネットワーク応用技術研究所 2007/11/19_18:39:06
・こちらこそお世話になります。
  了解いたしました。

2: 株式会社ネットワーク応用技術研究所 2007/11/19_17:26:27
・こちらこそお世話になります。
  了解いたしました。

1: システムブレーン 2007/11/19_16:51:36
・毎々お世話になります。
  本案件に対する要員の手当てができない状況です。
  誠に申し訳ありません。
```

Fig. 4. A list of issued Business Questions (top), A message matched between Business Question and Acceptance Condition (middle), Inquiry Window (bottom)

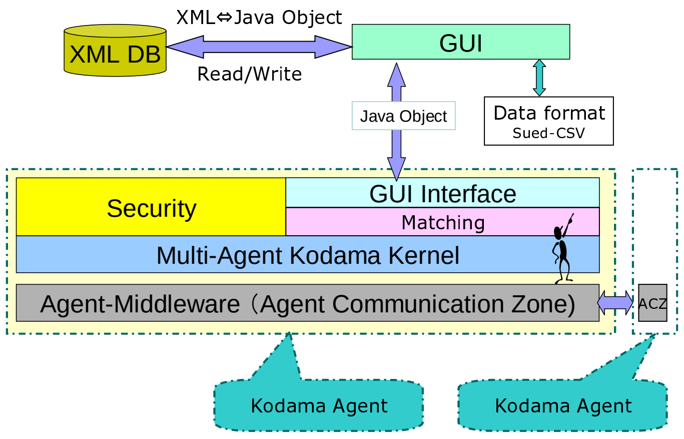


Fig. 5. System Structure

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. Since the two templates are similar, we will show the business question window of the semiconductor development field in Fig. 6. A category dependent item consists of a set of choices with a check box or a radio button and a free description field. Each choice is composed of a header and a description field for describing comments.

By having question templates selectable, matching between various industries gets to be possible.

When making matching between a business question (Q) and an acceptance condition (C), the system calculates a similarity score between them by summing the total number of matched choices in all the items considering the weight of each item. If the item gives a free descriptive sentence field, descriptive sentences in it are analyzed morphologically and summing the average number of every matched terms in the sentences for every item between Q and C. All the combination of items including descriptive sentence field in Q and C are tried for matching as corresponding ones. The matched data of Q and C are handled as one object data, which are composed of a set of pair of items. Each data has an

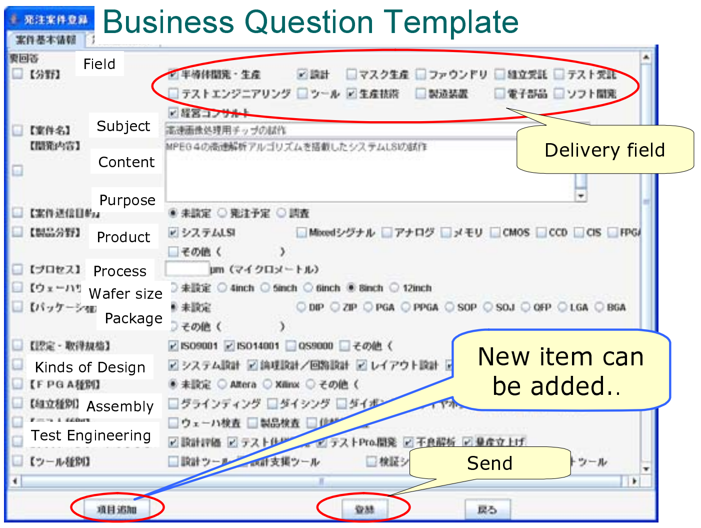


Fig. 6. Demander's Business Question Window

identifier (ID) to distinguish each other. The ID consists of its issued date and a requested condition counter, whose value is set to 0 when a business question is created. It is incremented by one for every transmission to the other agent. Those exchanged data are transformed into an XML format and are stored into an XML native database[3]. When receiving a reply-message, a system should consider not only the similarity score, but also the frequency exchanged with the agent returning the message in order to rank received messages. In particular, the messages returned by whom the user continues talking with should be way up on the high level so that they can be easily looked up.

3.3 Routing

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 at the sender agent's database. This is because the message is sent by peer-to-peer communication. Is it why ? If the message would be kept by a portal 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,

[3] We currently use eXist. http://exist-db.org/index.html

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. Consequently it 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 will log 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 a menu window shown in Fig.3.

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 routing via 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 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, 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 and 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 now currently used by personnel-service business companies.

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.

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. An improvement and evaluation of routing method based on OTT-shaped overlay network (in Japanese). In *Joint Agent Workshops and Symposium 2007(JAWS2007)*, pages (CD–ROM), 10 2007.
3. K. Kimura, S. Amamiya, T. Mine, and M. Amamiya. A Semi-structured Overlay Network for Large-scale Peer-to-peer Systems. To be submitted to *the Seventh International Workshop on Agents and Peer-to-Peer Computing (AP2PC2008).*
4. 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*, J90(9):2388–2397, 9 2007.
5. 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.
6. T. Mine, D. Matsuno, A. Kogo, and M. Amamiya. Design and implementation of agent community based peer-to-peer information retrieval method. In *The eighth International Workshop CIA 2004 on Cooperative Information Agents (CIA2004)*, volume LNAI 3191, pages 31–46, 9 2004.
7. RikuNavi. http://www.rikunabi.com/, 2006.
8. 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.
9. The Business Mall. http://www.b-mall.ne.jp, 2006.
10. JETRO TTPP http://www3.jetro.go.jp/ttppoas/indexj.html, 2006.
11. 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.