UDC 519.681:681.3.068

Integrating Information and Knowledge with Software Agents

●Hironobu Kitajima ●Ryusuke Masuoka ●Fumihiro Maruyama

(Manuscript received June 9, 2000)

This paper describes research and development activities for integrating distributed information and knowledge with software agent technology. First, we give an overview of our software agent architecture. Then, we describe the application of our software agent technology in a system that integrates distributed databases and in a document-oriented information integration system which integrates search engines. Our agent-based approach is suitable for dynamic information sources.

1. Introduction

The advent of the Internet has brought about what is called an "information flood." There is an enormous amount of information and knowledge digitally available through networks. Although such information and knowledge can be accessed on an individual basis, it is not an easy task for users to acquire appropriate information and knowledge.

The sources of information and knowledge have platform-level, system-level, and representation-level differences. Platform-level differences include the differences between hardware and operating systems, for example, differences between Solaris, Windows NT, and mainframe computers. System-level differences include the differences between database management systems such as Oracle, SQL Server, DB2 and proprietary applications on mainframe computers. Representation-level differences come from the different database structures and different vocabularies used in databases, which include different field names for corresponding fields and different field values with the same meaning. These three types of differences make it difficult for users to retrieve information and knowledge simultaneously from distributed sources and to make consecutive retrievals of associated information and knowledge.

Distributed and disparate information sources need to be integrated. This is where software agent technology comes into play. In this paper, a "software agent" is a computer system that is situated in an environment and is capable of autonomous action in this environment to meet its design objectives.1) A system of software agents hides platform-level, system-level, and representation-level differences and realizes a virtual integration. The areas where virtual integration can be applied include Supply Chain Management (SCM), Enterprise Application Integration (EAI), Enterprise Information Portals (EIPs), and Knowledge Management (KM).

This paper describes our activities in this area under the Smart AGent Environment (SAGE) project at Fujitsu Laboratories. Chapter 2 gives a general overview of our software agent architecture and introduces its application to the integration of distributed databases for SCM. Chapter 3 describes a system for document-oriented information integration which has been in practical use by Fujitsu's systems engineers since October 1998. Chapter 4 describes the standardization efforts and future directions of our project. Chapter 5 concludes the paper.

2. Integrating distributed databases

In this chapter, we first introduce the Smart AGent Environment (SAGE) project and its intelligent agent systems, which enable users to virtually integrate distributed information sources. In the rest of this chapter, we mainly discuss SAGE: Francis, which is an Electronic Commerce (EC) application of the SAGE project in which distributed and disparate Relational Databases (RDBs) are virtually integrated. Section 2.2 describes the architecture of SAGE: Francis, Section 2.3 describes its main features, and Section 2.4 describes our mediator agents called "facilitators." Then, in Section 2.5, we explain how SAGE: Francis works by describing an example application in a Supply Chain Management (SCM) prototype system. Finally, in Section 2.6, we discuss centralized-type and decentralized-type solutions for the integration of distributed information sources.

2.1 SAGE

We are conducting research on intelligent agent systems under the SAGE project. Under this project, we have developed our agent system and several agent system applications, which include SAGE: Francis[2),3)] and SAGE: Anthony. In both SAGE: Francis and SAGE: Anthony, the agent system is used for virtual integration of distributed databases and other disparate information sources. SAGE: Francis is an application of our agent system to Electronic Commerce (EC) settings where RDBs are virtually integrated to realize virtual catalogs. SAGE: Anthony, details of which are given in Chapter 3, is an application of the agent system to a virtual integration of document search engines. In both applications, the agent system enables the system integrators to give end users a virtually integrated view of distributed and disparate information sources without changing the way in which information sources are operated.

This chapter mainly describes SAGE: Francis. (SAGE: Francis lead to the development of an agent system software product called AGENTPRO.[4),5)] Fujitsu released AGENTPRO in August 1999.)

2.2 Architecture

Figure 1 shows the architecture of SAGE: Francis. It consists of user agents (UAs), facilitators (FAs), and database agents (DBAs).[note 1)] All agents communicate with each other via Agent Communication Language (ACL) messages, the syntax of which is defined by the Knowledge Query and Manipulation Language (KQML)[6)] and Knowledge Interchange Format (KIF).[7), note 2)]

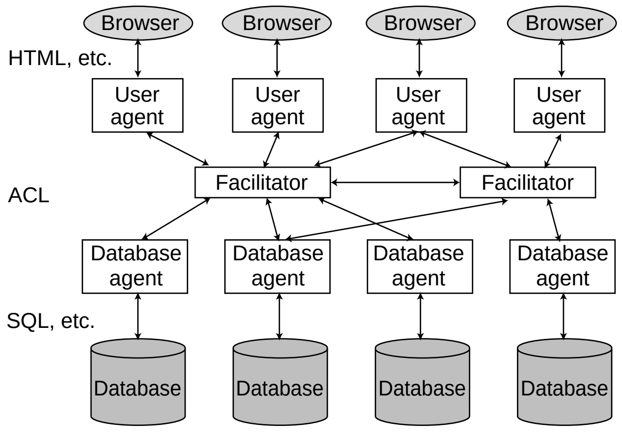


Figure 1
Architecture of SAGE.

note 1) There are also application agents (AAs) in AGENTPRO, which are created as wrappers for general applications. AAs are very close to DBAs. Essentially, they are DBAs in which database information sources are replaced with general applications.

note 2) KQML and KIF are often called DARPA ACL since they are outcomes of Knowledge Sharing Efforts (KSE), which is a DARPA project. Future versions of AGENTPRO will be FIPA-compliant.[8)]

Every ACL message in SAGE is sent and received asynchronously. Agents can automatically match ACL messages such as a query message and its resultant message. This is important because agents can send the messages when appropriate and operate freely.

The two main functions of UAs are to process conversions between ACL messages and expressions on user interfaces such as Web pages and to offer a support service for users. One of the main functions of DBAs is to process conversions between ACL messages and SQL queries: SQL is the query language of database management systems (DBMSs) such as Oracle and Microsoft Access. Two other key functions include converting DB data into virtual knowledge to be used in ACL messages from DBAs and informing FAs of the capabilities of DBAs (a function called "advertising"). The main functions of FAs, which act as mediators between UAs and DBAs, are forwarding messages to suitable DBAs, merging and sorting (if specified) results from multiple DBAs, replying accordingly if there are no suitable agents, and translating terms in ACL messages as necessary.

2.3 Main features

As a complete agent system, SAGE: Francis has the following main features:

1) Agentification of legacy information sources

SAGE: Francis provides DBAs to agentify legacy information sources such as databases. After the agentification, these information sources are made into agents which communicate via ACL messages using CORBA Object Request Broker (ORB). This hides platform-level and system-level differences between information sources.

2) Dynamic system configuration by advertise messages

DBAs can communicate knowledge about themselves to FAs by sending advertise messages. This knowledge includes their addresses as agents, their information processing capability, information on what kind of information they have, and which ontology they use. There are also "unadvertise" messages, which enable DBAs to deny partially or entirely the knowledge sent by previous advertise messages.[note 3)]

Based on such knowledge, FAs can route and translate ACL messages appropriately. Hence, the system configuration is changed according to the advertise messages from DBAs. These advertise messages enable a decentralized method of system configuration, which allows DBAs to have autonomy over their operations. This advertisement mechanism insulates UAs, and therefore the users, from the locations and other details of information sources. The transparency provided by this mechanism is essential for virtual information integration.

3) Ontology and ontology translation

SAGE is based on ontologies. An ontology is a set of definitions of terms and the relationships between those terms. An ontology can be roughly defined as a vocabulary used by communicating agents.

Since distributed and disparate information sources are developed independently, they usually use different sets of terms. Even though agents use the same syntax for ACL messages, for example, KQML and KIF, ontologies used in ACL messages can be different. Therefore, FAs provide an ontology translation service. With this service, the agent system hides vocabulary-level difference from users. This is another essential element for virtual information integration.

FAs also provide flexible routing of ACL messages based on the messages' ontology, which uses its own category information. This is one of the merits in having explicit ontologies.

4) Ontology Alignment Tool (OAT)

This support tool provides users with a visual interface for manipulating ontology and ontology translation information (**Figure 2**). It was created because only humans can provide

note 3) DBAs and AAs can edit an FA's knowledge about themselves with advertise and unadvertise messages.

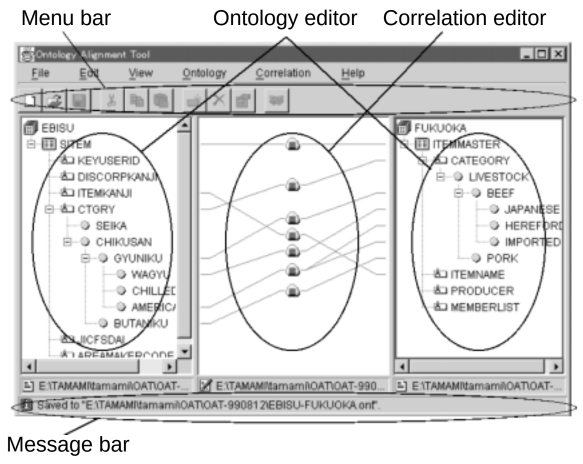


Figure 2
Ontology Alignment Tool (OAT).

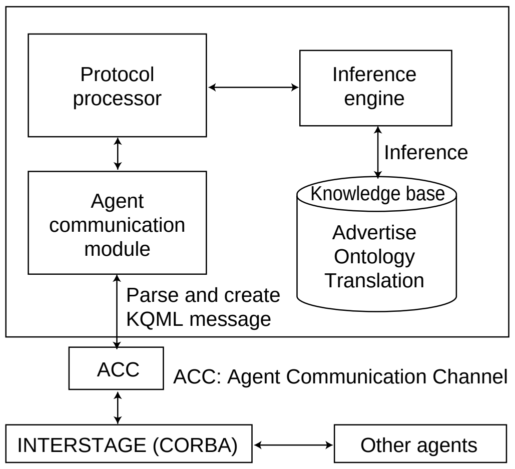


Figure 3
Architecture of a facilitator (FA).

correct meanings, relations, and correspondences of terms. We are currently considering adding higher-level support functions to alleviate human tasks related to ontology manipulation.

5) Merging and sorting of ACL messages

FAs merge the reply messages from multiple DBAs for an original query message and sort them if so specified. Because of this function of FAs, a user sees the result of a query as a single list, even though the results may have come from several DBAs.

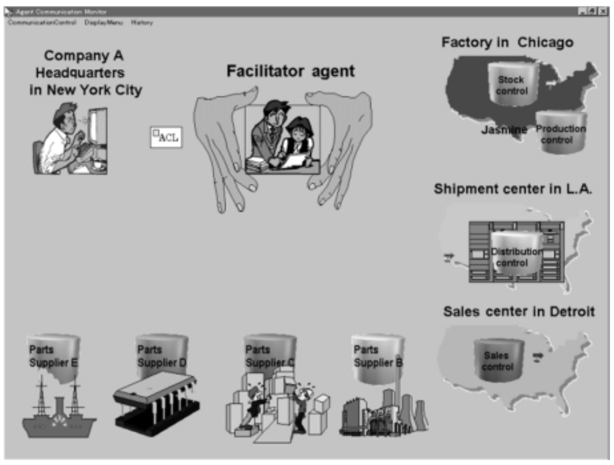
2.4 Facilitators (FAs)

In this section, we give some details about facilitators (FAs), which constitute the core of our agent system. Other agents, for example, UAs and DBAs, differ from FAs in that they have connections to outer resources such as users and databases. Another major difference between FAs and other agents is in their implementation languages. Common Lisp is used for the implementation of FAs, while Java is used for the implementation of UAs and DBAs.

Figure 3 shows the architecture of an FA. One of the most important modules is the protocol processor. FAs need to manage multiple and complex sessions of message exchanges. To realize this management, a protocol object is created for each session at the receipt of a session-initiating message. A protocol object is a Finite State Machine (FSM) that keeps the state of the session. For each protocol object, the protocol processor checks, at fixed intervals, whether a timeout occurs and whether the conditions required by the protocol are satisfied. If a timeout occurs or the conditions are satisfied, the actions specified in the protocol object are executed. This mechanism enables FAs to be involved in multiple, complex sessions at the same time.

Other important modules are the inference engine and the knowledge base. The knowledge base keeps ontologies and ontology translations loaded at the startup of the FA. It also keeps advertise information, which is dynamically inserted by advertise messages from DBAs and is dynamically deleted by unadvertise messages. The inference engine is used to match the query messages from UAs and advertise information. Ontologies and ontology translations are utilized during this matching process.

FAs communicate with other agents through the agent communication module. The module stores all received messages and allows other modules to read them by function calls. It provides the functions to parse and create ACL messages. The module, in turn, uses the Agent Communication Channel (ACC) to send and receive the messages. The ACC supports asyn-



This prototype was built for an imaginary TV manufacturer and associated parts suppliers. By agentifying databases and applications at a factory, shipment center, sales center, and parts suppliers distributed in a country, mediation by the facilitator enables the prototype to answer a query by a user at the headquarters of the TV manufacturer regarding the delivery date for a specified amount of product.

Figure 4
SCM prototype by agent system.

chronous messaging between the agents. This agent communication mechanism is built on INTERSTAGE,[9] which is Fujitsu's CORBA middleware.

2.5 How it works: an SCM system prototype

In this section, by using an SCM system prototype, we describe how SAGE: Francis is applied to virtually integrate various information sources and how it works. This SCM system prototype was developed for a demonstration of AGENTPRO. It uses application agents (AAs) along with DBAs which agentify general applications. The agent system used in this prototype is somewhat more advanced than the currently released version of AGENTPRO. The advanced features will be provided as a part of the future versions of AGENTPRO.

One of the purposes of this SCM prototype system is to provide an estimate of the delivery time of products by virtually integrating distributed information sources with the agent system. The scenario of the prototype, which was built for an imaginary TV manufacturer and associated parts suppliers (**Figure 4**), goes as follows.

The TV manufacturer has a factory, a shipment center, a sales center, and parts suppliers distributed throughout a country. Each department or supplier has its own databases. Since they are developed more or less independently, they use different Database Management Systems (DBMSs), for example, Oracle and Microsoft Access, and they have different applications on their mainframe computers. They also use different ontologies for their field names and data in their databases.

The manufacturer introduced the agent system to create an SCM system that will enable the sales persons to provide estimated delivery dates. First, they agentify the distributed databases into the DBAs. Then, they use a visual tool, OAT, to create ontology files and ontology translation files for translation between different ontologies. These files are provided for the FA, a mediator agent, which provides ontology translation and other ontology-related services.

When the DBAs and AAs are started, they send out advertise messages to the FA. The advertise messages describe the DBAs' and AAs' capabilities and what kind of information those agents have. They also create UAs, which interpret the users' intentions into agent messages and which communicate received agent messages to users. The users' side of the agents is realized by Web interfaces. Through the Web interface of the UA, a sales person at the headquarters makes a query about the estimated delivery date of a specific product. Then, the UA sends out the query in an ACL message to the FA.

Based on the information provided by the advertise messages from the DBAs and AAs, the FA queries the DBAs and AAs about the stocks at the production line, the shipment center, and sales center. The FA sends out to each DBA or AA a query message in the ontology used by the agent. Then, the FA waits for the asynchronous replies from the DBAs and AAs.

If there are not enough stocks, the FA queries the DBA at the factory about the products on the production line. If there are not enough products on the production line to make up for the shortfall, the FA further queries the DBA at the factory for the parts needed for the necessary additional production. Then, the FA asks the DBAs and AAs of the parts suppliers for the delivery dates of those parts. With the obtained information, the FA comes back to the DBA at the factory and queries about the estimated date for completion of the additional production. Then, the FA summarizes all the information and sends a message containing the result to the UA, which in turn displays the result to the user.

A new parts supplier can join the system by agentifying its database and by sending an advertise message to the FA when it is ready. There are also unadvertise messages that remove the information of the senders from the knowledge base of the receiver. This advertisement mechanism enables dynamic configuration of the total system, allowing departments and suppliers to join and leave at their convenience and to have autonomy over their operations.

2.6 Centralized-type versus decentralized-type solutions

In this section, we compare centralized-type and decentralized-type solutions for integrating distributed information sources.

Centralized-type solutions retrieve data from distributed information sources and send it to a central server using robots or FTP. The central server usually consists of one or more very high performance computers. Then, indices are given to the collected data. The central server provides services such as a query service for the collected data. Our agent system is an example of a decentralized-type solution.

Both types of solutions have their own merits and demerits. Our argument is that recent rapid changes in the corporate environment and the need to interoperate with other enterprises favor decentralized-type solutions.

With centralized-type solutions, the performance is independent of the performance of the software and hardware used for distributed information sources. It is also independent of the communication environment between the central server and distributed information sources. Those factors can work against decentralized-type solutions. On the other hand, decentralized-type solutions work better when changes and varieties are abundant. Changes include addition and deletion of distributed information sources. Varieties include those of platforms, systems, and vocabularies.

Sometimes an open environment prohibits the use of centralized-type solutions, because the participating distributed information sources are operated independently and autonomously. These sources will not or cannot submit all their information to the central server, but only accept one-time queries.

Now, there is a new situation which makes centralized-type solutions difficult. That is, more and more information is being created or composed dynamically, especially for Web pages. This situation prevents centralized-type solutions from collecting all the data from distributed data sources efficiently since often an indefinite number of pages can be created by the combination of the original data held there. In order to deal with such cases, all of the distributed information sources have to be queried each time a user makes a query and, therefore, decentralized-type solutions are the only options.

3. SAGE: Anthony

3.1 Overview

We have already developed an intelligent agent environment named SAGE. It uses a kind of mediation agent, called a facilitator (FA), as an essential part of the technology. SAGE has proved its usefulness in the SAGE: Francis project, the target of which is the virtual integration of multiple RDBs over a network. As a next step, we started another project to build a new information system, called SAGE: Anthony,[10] for systems engineers at Fujitsu.

The systems engineers of our company had been using a lot of isolated information systems which are mainly based on WWW servers and are administrated independently by different departments. Since most of the systems have no relation to each other and their authentication processes were not unified, it was a very laborious task to search the network for information. SAGE: Anthony is an integrated document search engine system which can address this issue, even over a very large scale network.

In this chapter, we present a brief description of the system from the architectural and functional points of view. Compared with conventional document search engine systems, SAGE: Anthony has the following features.

1) Dispersiveness

Documents to be searched and index-files to be used by search engines in their keyword-matching processes are not centralized. Document indexes are stored in multiple local sites equipped with search engines. Each site is managed independently and is only responsible for searching its locally owned documents.

2) Standardization of information sources

We defined a document database equipped with a search engine as a basic unit for information sources and transformed it into a DBA of SAGE: Anthony. If there were differences in the designs of the search engine programs, we standardized them through their agentifying processes. The DBA usually holds index-data about documents only; the documents themselves may be accumulated in other sites such as WWW servers.

3) Efficient searching with facilitation

SAGE: Anthony has a federated architecture[11] so that the facilitation services of its FA can perform efficient searches. These services are based on the FA's internally stored knowledge about other agents and play a key role in SAGE: Anthony's economical distributed searching of documents.

4) Dynamic integration of information sources

The DBAs are dynamically integrated by sending advertise messages to the FA. Such messages can keep the FA informed of knowledge about DBAs and ready for providing facilitation services.

5) Unified authentication

The user authentication processes of multiple information sources are unified into the one-time login step of SAGE: Anthony's entry point. This service is brought by an authentication agent, which is a special kind of DBA that has information about all users and groups.

6) Scalability

SAGE: Anthony's natural dispersiveness provides a certain degree of system scalability. Since we allotted a portion of the document-searching task to each DBA and kept its index-file size smaller than that of the conventional centralized search engine system, the time efficiency is higher and updating of indexes and files can be done more frequently.

Efforts to integrate document-searching services for enterprise information systems are already underway. One of these systems is Domino Extended Search (DES).[12] DES provides distributed heterogeneous searches across the network using the user's explicit selection of information sources. Compared with this, SAGE: Anthony can automatically determine the recommendable DBAs by analyzing the previously acquired knowledge about DBAs.

SAGE: Anthony is a kind of three-tier system whose middle tier is a facilitator agent. The middle tier stores information to be used for the virtual integration of information sources. There has been some research into heterogeneous searches with three-tier systems;[13)-15)] however, the middle tiers of these systems are rather static. SAGE: Anthony's FA is so dynamic that the system can realize on-the-fly modification of searching policies by real-time updating of stored knowledge in the FA.

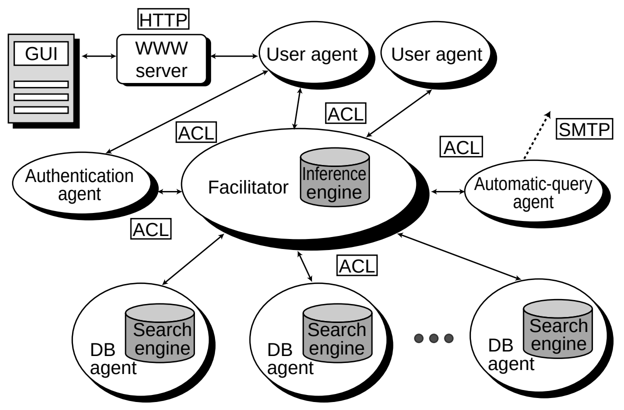


Figure 5
SAGE: Anthony.

3.2 Architecture

SAGE: Anthony inherited its federated architecture from the design concept of SAGE: Francis. **Figure 5** shows a schematic of the agent system. As can be seen, the architecture is very similar to that of SAGE: Francis. Actually, there is little difference between these two systems from the architectural point of view. The principal difference is in the content of the facilitation services.

A WWW browser is used for the GUI. The browser can access SAGE: Anthony via a WWW server using a UA and can also display the results of document searching for human users. The UA, which is an entrance of SAGE's agent world, can translate the user's operational actions into appropriate ACL messages.

Each document database has a keyword-matching-based search engine for its own use. The database is agentified to a DBA that can only be accessed from an FA with ACL messages.

The FA provides facilitation services based on the content of the message from the UA, utilizing previously acquired knowledge concerning other agents. Since there is only one FA in the system, it may become a bottleneck that reduces the searching performance. We therefore enabled mirroring of the FA in order to further enhance the scalability of this system.

This system features an authentication agent and an automatic-query agent. The former is a special kind of DBA that maintains information about all users and groups for the unified authentication process. The latter is a substitutional agent that executes scheduled queries on behalf of human users. The automatic-query agent uses the SMTP protocol for returning answers of automated queries to the users.

3.3 Facilitation services

A typical document search proceeds through message-passing routines which are very similar to those of SAGE: Francis. For example, when a UA of SAGE: Anthony issues a query by sending a KQML's ask-all message to the FA, it redirects the message to the recommendable DBAs based on its knowledge about DBAs. After receiving the ask-all message from the FA, each DBA executes an exhaustive document search and returns the answer to the FA by sending reply messages. The message includes information about searched documents such as URLs, titles, and file-sizes. The FA collects such answers and returns a merged reply message to the UA that originated the query task.

In other aspects, however, there are differences between SAGE: Francis and SAGE: Anthony. Whereas SAGE: Francis' target of virtual integration is an RDB, SAGE: Anthony integrates search engines. The contents of the facilitation service, therefore, needed to be designed uniquely.

The FA of SAGE: Anthony provides intelligent facilitation services. It determines recommendable DBAs based on various kinds of internal knowledge and redirects received query messages to them. The knowledge can be classified into the following three domains.

1) Database category

DBAs are classified by a pre-defined taxonomy concerning their contents and owners. Some example classifications are "Technical Information," "Product Information," "Industrial System Department," and "Medical System Division." Such classifications, which are usually organized

```
(ask-all
 :content
   (and
     (category ?database "finance")
     (has ?database ?doc)
     (and (keyword ?doc "keyword1")
          (or (keyword ?doc "keyword2")
            (keyword ?doc "keyword3"))
          (not (keyword ?doc "keyword4")))
     (url ?doc ?url)
     (title ?doc ?title)
     (byte-size ?doc ?byte-size)
     (registered-date ?doc ?registered-date)
     (summary ?doc ?summary)
     (ranking ?doc ?ranking))
 :aspect (?url ?title ?byte-size
              ?registered-date ?summary ?ranking)
 :id "fj900557"
 :language KIF
 :language-encoding x-euc-jp
 :ontology database.fujitsu.kif
 :reply-with abcdefg-3-0
 :sender UserAgent@flab.fujitsu.co.jp
 :receiver Facilitator@flab.fujitsu.co.jp)
```

Figure 6
Ask-all message.

```
(reply
 :content (
  ("http://www.fujitsu.co.jp"
   "Fujitsu's Homepage-1" 1345
    "19971205T000000000"
   "Fujitsu's pen computer is adopted by
    NASA's Space Shuttle project..."
    12.5)
  ("http://abc.www.fujitsu.co.jp"
   "Fujitsu's Homepage-2" 2012
    "19960303T000000000"
    "Here's very useful information about..."
    31.3)
  ...
  ...
 )
 :language KIF
 :language-encoding x-euc-jp
 :ontology database.fujitsu.kif
 :in-reply-to abcdefg-6-0
 :reply-with abcdefg-6-1
 :sender DBAgent@flab.fujitsu.co.jp
 :receiver Facilitator@flab.fujitsu.co.jp)
```

Figure 7
Reply message.

in a family-tree style, can be regarded as a special ontology to classify information sources. The FA knows as basic premises which category of document is indexed in which DBA. Users can specify such categories when they query the system.

2) Keyword information

The FA has a keyword-database to store records of keyword-related information consisting of keywords, names of DBAs, and frequencies of documents. The frequency indicates the number of documents indexed in the DBA that match the keyword. If we expect the information to be complete, the number of such records may be huge. Therefore, the keywords to be stored are automatically selected according to the frequency at which they appear in users' query conditions.

3) User profile

The user profile consists of attributive information about users such as the user's user name, password, work, and post. The authentication agent maintains a user-profile database and communicates its knowledge to the FA on demand. The FA possesses DBA profiles which contain similar knowledge to the user profiles. The FA utilizes these two kinds of profiles to check the user's accessibility to each DBA.

Most of the knowledge mentioned above must be acquired in advance of the searching task by receiving KQML advertise messages from other agents. The knowledge can always be updated dynamically by additional advertisement or by direct updating of databases. The FA has an inference engine to store and handle such knowledge, which can be applied for intelligent facilitation services.

3.4 Message examples

In this section, we show some examples of the ACL messages used in the conversation process among agents in order to present a clear image of ACL message passing.

Figure 6 shows an ask-all message sent to the FA from a UA. The UA is making a query about documents that are held by the DBAs of the finance category and are matched to four specified keywords by boolean operations. The part of the message tagged with ":aspect" prescribes that the answer message must be composed of document properties such as a URL, title, byte size, and so on. After receiving the ask-all message, the FA redirects it to the recommendable DBAs. In the example shown in **Figure 7**, a DBA is replying to the FA with the searched documents' properties that are specified in the ask-all mes-

```
(advertise
 :content
 ((database documentDB-1)
  (has-database DBAgent@flab.fujitsu.co.jp
    documentDB-1)
  (know-category documentDB-1 "finance")
  (know-category documentDB-1 "medicine")
  (ontology DBAgent@flab.fujitsu.co.jp
    standard.database.fujitsu.kif)
  (access_level documentDB-1 "0")
  (department_flag documentDB-1 "c10000")
  (post_flag documentDB-1 "ffffff")
  (place_flag documentDB-1 "ffffff")
  (allows-relational-db-query
    DBAgent@flab.fujitsu.co.jp)
  (table-definition documentDB-1
     "document database"
     '(description "database of documents"))
  (field-definition documentDB-1
    "document database"
    "keyword"
    '(type text description
       "keyword to be searched"))
  ...
  )
  :sender DBAgent@flab.fujitsu.co.jp
  :receiver Facilitator@flab.fujitsu.co.jp
  :reply-with abcdefg-4-1
  :language-encoding x-euc-jp
  :language KIF
  :ontology database.fujitsu.kif
    )
```

Figure 8
Advertise message.

sage shown in Figure 6.

Figure 8 shows an advertise message sent to the FA from a DBA. The DBA is advertising its name, database category, ontology name, DBA profiles, and other properties. These advertisements will be applied to the intelligent facilitation services.

3.5 Current status

After the successful prototyping stage, SAGE: Anthony evolved into a practical information system with industry-level robustness and with functionality for easy administration. The system, named FIND Future, has been running for 10s of thousands of Fujitsu's systems engineers since October 1998. **Figure 9** shows a screen-captured user's view of the system.

Currently, FIND Future integrates about 50 DBAs installed throughout Fujitsu's intranet. It boasts a huge number of accesses from every part of our company in Japan. In the very near future, FIND Future will be expanded to a worldwide enterprise information system by adding federated facilitation services among multiple independent FAs.

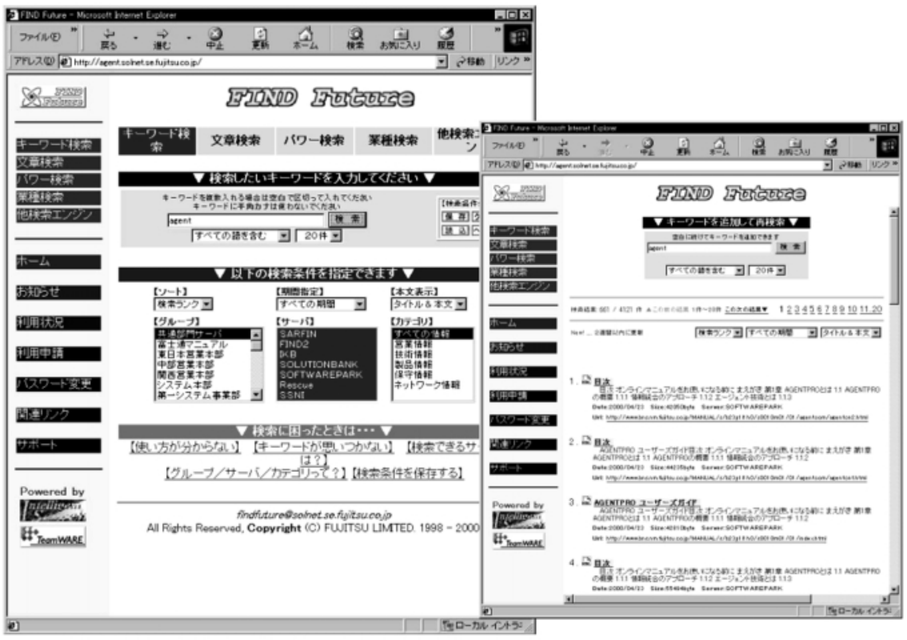


Figure 9
FIND Future.

4. Standardization efforts and future direction

In this chapter, we describe our standardization efforts and the future direction of our agent research.

Regarding our standardization efforts, we are committed to the Foundation for Intelligent Physical Agents (FIPA), which is an agent standardization organization. We have been active in the area of ontology and content language standardizations. Also, we have recently submitted a workplan called the "Agent Description Ontology." The Technical Committee (TC) of the Agreement Management of the FIPA has decided to include this workplan in its mission statement and to produce the specification under the name of the "Service Description Ontology." One of the authors (Hironobu Kitajima) has been appointed as the redactor of the specification. This workplan aims to specify ontologies to explicitly and formally describe agent services, agent capabilities, and agent needs. Realizing these aims will enable more accurate and efficient interoperation between agents. Furthermore, their realizations are the absolute prerequisites for content-based routing.

Almost all application domains may benefit from these specifications. In particular, those that involve loosely coupled processes, for example, database integration, electronic commerce, and CALS tools, will benefit.

We are also conducting interoperability tests with other agent platforms. We have succeeded in an interoperability test between a modified version of AGENTPRO and Comtec Agent Platform based on FIPA97 specifications.[16] We have also succeeded in a private interoperability test between a modified version of AGENTPRO and JADE Agent Platform.[17] We plan to provide a new agent platform in the future versions of AGENTPRO which will conform to a series of new FIPA specifications released in 2000.

Regarding the future direction of our agent research, we expect that it will be connected with Enterprise Application Integration (EAI) and Enterprise Information Portals (EIPs). For EAI and EIPs, both within and between enterprises, customized integration of disparate and distributed information sources is essential.

For enterprises to stay competitive in these application areas, it will be more and more important to adapt to changes. Enterprises need to change their information systems to introduce new services for customers and to distinguish themselves from their competitors. There is also an ever changing environment, and new laws and rules will continue to be introduced. Restructuring of enterprises, mergers, acquisitions, etc., will necessitate the accommodation of new information sources. Enterprises should be ready to deal with new users and new requests from users.

With such an enormous pressure from changes, enterprises will need to personalize and optimize their services for each user. To deal with this situation, we firmly believe that **automatic service integration** of disparate information sources and information systems is quintessential. Inside and outside changes force us to have externalized rules instead of knowledge programmed into code and to constantly reconfigure the services based on those rules to satisfy each user's request.

We consider that Artificial Intelligence (AI) technology will play a vital role in automatic service integration. We also consider that ontology will be the key to gluing multiple services together automatically and that the Service Description Ontology mentioned above provides the basic mechanism for sharing knowledge needed to realize automatic service integration.

5. Conclusion

We have described our research and development activities for integrating distributed information and knowledge with software agent technology. Applications of our integration technology include the integration of distributed databases such as SCM systems and document-oriented information integration systems such as information sharing systems. In August 1999, Fujitsu released a software product called AGENT-PRO based on our integration technology. Currently, we are working on extending our technology to advanced EAI and EIPs.

References

1) Gerhard Weiss, ed.: Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence. The MIT Press: Cambridge, MA, 1999.
2) R. Masuoka, T. Sugasaka, A. Sato, H. Kitajima, and F. Maruyama: SAGE and Its Application to Inter-company EC. Proceedings of PAAM98, pp.123-135, 1998.
3) T. Sugasaka, K. Tanaka, R. Masuoka, A. Sato, H. Kitajima, and F. Maruyama: A conversational agent system and its application to electronic commerce. Proceedings of the fourth & fifth world conference on integrated design and process technology (IDPT 2000), June, 2000.
4) INTERSTAGE AGENTPRO DataSheet. *http://www.interstage.com/images/AgentProDataSheet.pdf*
5) Dynamic Advertising – The Key to Intelligent Data Retrieval. *http://www.interstage.com/images/AGENTPRO%20Whitepaper1.pdf*
6) The DARPA Knowledge Sharing Initiative External Interfaces Working Group: Specification of the KQML Agent Communication Language, 1994/2/9. *http://logic.stanford.edu/papers/kqml.ps*
7) Draft proposed American National Standard (dpANS): Knowledge Interchange Format (KIF), NCITS. T2/98-004. *http://logic.stanford.edu/kif/dpans.html*
8) FIPA (Foundation for Intelligent and Physical Agents). *http://www.fipa.org/*
9) INTERSTAGE. *http://www.interstage.com/*
10) H. Kitajima, A. Kawamura, T. Yoshino, and F. Maruyama: SAGE: Anthony. (in Japanese), Computer Software, **17**, 1, pp.32-44 (2000).
11) M. R. Cutkosky et al.: PACT: An Experiment in Integrating Concurrent Engineering Systems. IEEE Computer, January 1993, pp.28-38.
12) Lotus Development Corporation, Domino Extended Search. *http://www.lotus.com/home.nsf/welcome/domsearch*
13) Y. Arens, Chee, Y. Chin, Hsu, Chun-Nan and Knoblock, and A. Craig : Retrieving and Integrating Data from Multiple Information Sources. International Journal of Intelligent and Cooperative Information Systems, **2**, 2, pp.127-158 (1993).
14) H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and Jennifer Widom: Integrating and Accessing Heterogeneous Information Sources in TSIMMIS. Proceedings of the AAAI Symposium on Information Gathering, Stanford, California, March 1995, pp.61-64.
15) A. Y. Levy, A. Rajaraman, and J. J. Ordille: Querying Heterogeneous Information Sources Using Source Descriptions. Proceedings of the 22nd International Conference on Very Large Databases. VLDB-96, Bombay, India, September 1996.
16) FIPA-related software developed by members and non-members. *http://www.fipa.org/fipa9706.pdf*
17) JADE (Java Agent DEvelopment Framework). *http://sharon.cselt.it/projects/jade/*

Hironobu Kitajima received the B.A.S. degree from the University of Tokyo, Tokyo, Japan in 1986. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1993 and has been engaged in research and development of simulated annealing and agent systems. He was a visiting scholar of Stanford University from 1996 to 1997. He is a member of the Information Processing Society of Japan (IPSJ).

E-mail: kitajima@jp.fujitsu.com

Ryusuke Masuoka received the B.S. and M.S. degrees in Mathematics from the University of Tokyo, Tokyo, Japan in 1985 and 1987, respectively. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1988 and has been engaged in research and development of neural networks, simulated annealing, and agent systems. He is a member of the Institute of Electrical and Electronics Engineers (IEEE), the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan, and the Information Processing Society of Japan (IPSJ). He received the Best Author Award from the IPSJ in 1995 and his Ph.D. degree in Mathematical Sciences from the University of Tokyo in 2000.

E-mail: masuoka.ryusuke@jp.fujitsu.com

Fumihiro Maruyama received the B.S. degree in Mathematical Engineering from the University of Tokyo, Japan in 1978. He joined Fujitsu Laboratories Ltd., Kawasaki, Japan in 1978 and has been engaged in research and development of computer-aided design and artificial intelligence. He received his Dr. of Engineering degree in Information Engineering from the University of Tokyo in 1991. He received the IPSJ 20th Anniversary Best Paper Award and Prof. Motooka Commemorative Award in 1980 and 1988, respectively. He is a member of the Institute of Electrical and Electronics Engineers (IEEE), the Information Processing Society of Japan (IPSJ), the Japanese Society for Artificial Intelligence (JSAI), and the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan.

E-mail: maruyama.f@jp.fujitsu.com