Comments and Suggestions on FIPA 98 Specification Part 12 Ontology Service [*]

Ryusuke Masuoka [†]
NetMedia Laboratory
Fujitsu Laboratories Ltd.

Fujitsu is currently devoting considerable efforts in research and development of agent technology. These efforts range from agent platforms to ontologies used in agent systems.

We are applying our agent system to many real-world applications such as CALS (Commerce At Light Speed), inter-company EC, online database integration, knowledge management, etc.

From such experience, we have been making comments, suggestions, and proposals to FIPA since the Palo Alto meeting.

The comments and suggestions in this document fall on FIPA 98 Specification Part 12 Ontology Service ([1]). We can see a lot of efforts have been put into this specification. This specification is quite a practical one even though discussions on ontologies can be very conceptual.

Still we believe we can give several comments and suggestions on the specification from a very pragmatic point of view of interoperabilities between agents and applicability of agent technology to real-world applications.

First we will introduce you to our research and development activities in agent technology shortly with emphasis on facilitators. Then we will give comments and suggestions.

1 Fujitsu research and development activities

Fujitsu research and development activities can be categorized as follows.

- Agent platform
  - Mobile agents
  - Service brokerage
- Intelligent Agents
  - Facilitators

[*] Revision: 1.6

[†] masuoka@flab.fujitsu.co.jp

– Ontologies

For agent platform, mobile agents and service brokerage are the main R & D areas. We aim at seamless and transparent agent collaboration with the technology.

R & D of intelligent agents are being done under the SAGE (Smart AGent Environment) project with emphasis on seamless integration of heterogeneous information distributed over networks. The main topics of the SAGE project are facilitators and ontologies. Those and other research areas for the SAGE project include:

- Agentification of users and legacy applications
- Facilitation of interoperation between agents by facilitators
- Ontologies used in agent systems
- Message formats and transactions
- Libraries and tools for the above
- Real-world applications

The project utilizes software agent technology, especially of those conversational agents which communicate by ACL (Agent Communication Language) [3]. Currently we use KQML (Knowledge Query and Manipulation Language) [4] for the message format and KIF (Knowledge Interchange Format) [2] for content language in our agent system, but use of FIPA-ACL is under consideration.

A facilitator is an agent which realizes content-based brokerage services. Since facilitators depend on contents of messages to provide their services, ontologies are of crucial importance. We will give more details on R & D efforts on facilitators in the rest of this chapter.

The architecture of SAGE has a 3-tier structure where facilitators are placed between user agents and database agents. User agents represent users as agents to SAGE and database agents represent databases and other information sources as agents to SAGE. Facilitators stand between them to provide brokerage, translation, merge and other services to help those agents interoperate.

The system works as follows.

1. When a database agent starts, it advertises categories, message formats and ontology it can handle to the facilitator.
2. When a user makes a query through the user interface, the user agent turns the query from the user into an ACL message and sends it to the facilitator.
3. The facilitator chooses the appropriate database agents based on advertise/unadvertise messages from the database agent and translates it for each database agent if necessary. [1]
4. The facilitator sends out the messages to the chosen database agents and wait for the reply messages.

[1] We allow database agents to have different ontologies for KIF terms in KIF content of the messages.

5. The database agent changes the ACL message from the facilitator into a SQL query and consult the database.

6. The result from the database is composed into an ACL message and the ACL message is sent back to the facilitator.

7. The facilitator merges the messages from the database agents into one and sends back to the user agent.

8. The user agent displays the result.

The facilitators play an essential role as a broker in the scenario above. What facilitators do are summarized as follows: [2]

- Selects appropriate database agents for the query message based on the stored advertise messages.
- Forwards the query message to the selected database agents.
- Translates messages from one ontology to another using translation knowledge if necessary.
- Merges the reply messages from the database agents and sends the merged (and sorted) results to the user agent.

2 Comments and suggestions

In this section, we will give comments and suggestions on FIPA 98 Specification Part 12 Ontology Service. These comments and suggestions are produced from our efforts in applying our agent system with facilitators to the applications.

2.1 An ontology for advertise/unadvertise

We believe an ontology is necessary to advertise and unadvertise agent capabilities and its properties.

This ontology can be appended to the document [1] as an annex like Annex D "Ontology of FIPA 97 Agent Management." Or this might be considered as a comment on ACL (Agent Communication Language).

FIPA 97 Part 1 introduces DF (Directory Facilitator) which provides yellow page service for software agents. FIPA 97 Part 3 introduces a mechanism where ARB's (Agent Resource Brokers) provide yellow page service for non-agentised softwares.

But what we have learned is that a more generic ontology is needed to communicate agent capabilities and properties. In our agent system, a facilitator receives advertise/unadvertise messages from database agents (information provider agents) and the facilitator provides brokerage services for querying agents. In this system, the efficiency and the quality of such services depend on how accurately agents can communicate their capabilities and properties to other agents.

[2] This is not an exhaustive list of facilitator's functions.

Here we list the identified categories of agent capabilities and properties that should be communicated.

Message contents the agent can handle: Since agents are realized as entities which send and receive messages, their capabilities are described by kinds of message contents they can handle. Even if agents use the same ontology, agents differ from one to another in their knowledge. The differences in their knowledge reflect on what kinds of message contents they can handle. Message contents an agent can handle maps to the extent of VKB (Virtual Knowledge Base) of the agent in case of KQML agents. (c.f. [4])

Agent properties: Agent properties such as access control list of other agents which can access the information in the agent or the agent's specialties. This information can be used for efficient brokerage.

Protocol/Flow: Protocol handling capabilities. Protocols such as in FIPA 97 Part 2, Chapter 7.

Retraction: Retraction of above information. Retraction of (all of or part of) above information is a "must have" to communicate the changes in the information provider agents.

2.2 Versions of Ontologies

The world is dynamic. Changes are ways of life.

Some japanese large retail store changes its category structure of their products every year even for its large cost. This is because they need to see the products from a different point of view to reflect the current trend of consumers.

Some companies do BPR (Business Process Reengineering). BPR is essentially continuous introduction of changes in points of view for how they see tasks are done in their companies.

New views introduce new ontologies. Of course, the new ontology retains relationships to the old one. These old and new ontologies should be referred as the same entity but as having different versions. The difference between the old and new should be addressed by the difference in the version, since both are used to describe same things.

There can be the option of always giving a different name for every ontology. But it is more economical to use versions to differentiate ontologies for slight changes.

2.3 Combination of multiple ontologies

It is very difficult and often impossible to conjure up a single, comprehensive, and coherent ontology all at once.

The inheritance of ontologies is a possibility, but not an efficient mechanism for constructing a necessary ontology used in an application, since the inherited ontology will tend to be large.

Therefore the idea is "divide and conquer." The ontology can be thought as consisting of small components. The examples of small components are ontologies for basic concepts such as space, time, matter, object, event, action, etc., ontologies for generic tasks such as diagnosing, selling, etc., ontologies for generic domains such as medicine, automobiles, manufacturing, finance, etc., and ontologies for describing application specific concepts. We can combine these ontologies to produce an ontology used in an application.

Dynamic combinations of chosen ontologies are preferable. If the system integrators of an application can pick up the ontologies they really need and can combine them, they can have a terse, efficient, but functional ontology for their application.

The idea of combination of small ontologies is also discussed in Scalable Knowledge Composition ([6]).

In order to deal with this dynamic combination of ontologies, the following issues have to be considered.

Naming and referring Ontologies: "6.2 Naming and referring Ontologies" in [1] might need some modifications.

- Do we allow multiple ontologies dynamically combined in the first place?
- If so, do we always give one name for each specific combination of ontologies?
- If we do give one name, who will be responsible for the combination. (The candidate is the OA (Ontology Agent).)
- If we do not give one name, there should be a system to refer to such anonymous ontology combinations.

Introduction of namespaces: Introduction of namespaces is necessary. It should be something like namespaces in XML ([8]). Expression without namespaces should be allowed as default where only one ontology is used. But when multiple ontologies are used, namespace information should be added to each term to discern which ontology the term comes from.

Ontology for ontology combinations: An ontology to describe ontology combinations is necessary.

2.4 Same name for different meanings

The document [1] mentions "different names with same meanings" in several places. But there is no mention of "same name for different meanings."

It seems it is taken for granted that the word in one ontology has the same meaning as the word of the same spelling in another ontology. [3] Our experiences show that the word of the same spelling has deviation in its meaning from one domain to another.

Therefore, we believe the document [1] should mention "same name for different meanings" in its text. This might lead to the introduction of namespaces in section 2.3,

[3] We are not talking about examples such as *bank* of financial institute and *bank* of land along river. Even *bank* of financial institute has different meanings from one domain (context) to another.

since namespaces can be used to discern the words of the same spelling in different ontologies.

2.5 Ontology translation information format

The document [1] introduces ontology translation mechanism. The next thing in the line is sharing of ontology translation information.

If ontology translation information format can be produced, it can be used in messages between OA's (Ontology Agents) to share ontology translation information. It can also be used as file format for OA's to use. Both ways increase the portability of ontology translation information.

Namespaces in section 2.3 might be used in the format since ontology translation information involves terms from at least two ontologies.

References

[1] FIPA,
FIPA 98 Specification, Part 12 Ontology Service, Version 0.2 - 16/7/98.

[2] M.R.Genesereth and R.E.Fikes,
"Knowledge Interchange Format Version 3.0 Reference Manual," Technical Report Logic-92-1, Computer Science Department, Stanford Univ., 1992/6.

[3] M.R.Genesereth and S.P.Ketchpel,
"Software Agents," Comm. ACM Vol.37 No.7, 1994.

[4] Tim Finin and et al.,
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`

[5] R. Masuoka, et al., "SAGE and Its Application to Inter-company EC", Proceedings of PAAM98, pp.123 - 135.

[6] Scalable Knowledge Composition:
`http://www-db.stanford.edu/SKC/index.html`

[7] T. Sugasaka, et al., "An application of multi-agent technology to electronic commerce - SAGE:Francis -", Proceedings of I'MEDIAT98, `http://context.mit.edu/imediat98/proceedings.htm`

[8] Namespaces in XML,
`http://www.w3.org/TR/WD-xml-names`