

SABPO: A Standards Based and Pattern Oriented Multi-Agent Development Methodology

Oguz Dikenelli, Riza Cenk Erdur

Ege University, Department of Computer Engineering, 35100, Bornova, Izmir, Turkey.
{erdur, oguzd}@staff.ege.edu.tr

Abstract. Organizational metaphor, in which each agent plays a specific role to achieve organization's global goal(s), seems as the most suitable approach to model multi-agent systems. In this paper, a new methodology, which uses organizational metaphor as its basis and integrates this metaphor to the FIPA standards and well-known interaction protocols in a systematic way, is introduced. In addition to the proposed methodology, a new interaction pattern, which is called as "ontology inception", is introduced. This pattern illustrates how the methodologists can mine new agent interaction patterns from the requirements and how these patterns can be inserted into the FIPA standards to create a pattern catalogue. Naturally, interactions of each interaction pattern add new tasks to the participant agents. We believe that the requirements of these new tasks and how they are inserted into the agent's internal architecture must be documented as part of the pattern knowledge. Thus, internal requirements of the "ontology inception" pattern are discussed from the perspective of the participating agents. By this way, understandability of the pattern will be improved.

1. Introduction

Multi-agent systems (MASs) have been widely recognized as a new paradigm to develop complex, intelligent and distributed applications. Based on this acceptance, developers need new methodologies to construct MASs in a robust and repeatable fashion. In the last few years, several methodologies, which take agents as high-level abstract entities and develop the methodology around the agent concept, has been proposed. Most of these proposals either extend Object Oriented (OO) methodologies using agent abstraction instead of object or focus on the internal design of the individual agent architecture [11] [12] [13] [16]. However, MASs behave like social organizations where each agent plays a specific role within the organization to satisfy the organization's global goals. Hence, we need a different approach that takes organizational abstraction as a key concept and builds the methodology around this concept.

Gaia [17] is one of the well-known methodologies that uses the organizational view to construct the development methodology. In Gaia, an agent organization is defined by collection of roles, each role's responsibilities, and interactions between those roles. All of these entities try to satisfy the organization's global goals all together. Hence, Gaia assists the developers to systematically discover the abstract

entities of the organization (such as roles, responsibilities, interaction protocols and activities) and to map these entities to the concrete ones (such as agents, services and acquaintances). But, Gaia has been initially developed for modeling only the closed systems, where organizational structure of the system is static and do not change at run-time. It is a very critical shortcoming since MAS developers are mainly interested in open environments such as Internet.

This shortcoming of the Gaia methodology has been addressed by Zambonelli, Jennings, Wooldridge [19] and by Omicini [14]. Zambonelli, Jennings and Wooldridge introduced three additional organizational concepts over the Gaia: organizational rules, organizational structures and organizational patterns. Organizational rules are used to define global requirements of the organization. Hence, new agents can learn organization's global rules when they join the organization in an open environment. Organizational structure defines the architecture of the organization, which is selected among many possible alternatives according to the specified requirements. Organization patterns express pre-defined organizational structures that can be re-used from system to system.

Omicini's methodology, which is called SODA, explicitly takes the agent environment into account. Environment is modeled as services and agents use these services to work properly within the open environment according to the system's requirements. SODA also separates the role's tasks into individual and social ones. Social tasks are used to model the organizational aspects of a MAS.

It is certain that any methodology, which aims to model MASs must have a method to define organizational rules and must explicitly take the environment and organizational structures into account. However, neither SODA nor the methodology proposed by Zambonelli, Jennings and Wooldridge take standardization efforts of the agent community into account. FIPA organization was established to develop standards for supporting interoperability among agents and agent-based applications. FIPA standards define the required services to construct MASs working in open environments and define lots of interaction patterns to build robust organizational structures. We believe that any attempt of methodology development must take the FIPA standards as the basis. By this way, each attempt will refine the shortcomings of the previous ones and agent community will reach a unified methodology, which becomes the FIPA methodology specification. So, in this study a new methodology is proposed based on FIPA standards to illustrate our vision. After reaching a unified methodology, agent methodologists should focus on the identification of new interaction patterns and integration of these new patterns to the FIPA specification.

To illustrate the mechanism of FIPA compliant pattern identification, we introduce a new pattern called as "ontology inception". This pattern can be used to construct adaptable multi-agent organizations working in the environment, where ontologies change at run-time and new ontologies may be added to the system at any time.

The rest of the paper is organized as follows: Section 2 gives an overview of the SABPO methodology, especially discussing how the FIPA standards are used during the modeling process. The use of the methodology is illustrated by means of a case study in section 3. The "ontology inception" pattern is introduced in section 4. This introduction includes the motivation behind the "ontology inception" pattern, its interaction mechanism and its affect to the participating agent's internal architecture.

Last section concludes the study and outlines main contributions over the previous studies.

2. Overview of the SABPO Methodology

Our approach puts the FIPA's abstract architecture specification [7] to the center of the methodology as a basic organizational structure and tries to create a concrete FIPA architecture that satisfies system requirements. Before discussing the details of the methodology, we need to introduce the elements of the abstract architecture and the relationship between these elements. Figure-1 outlines the basic relationship between the key elements of the FIPA abstract architecture.

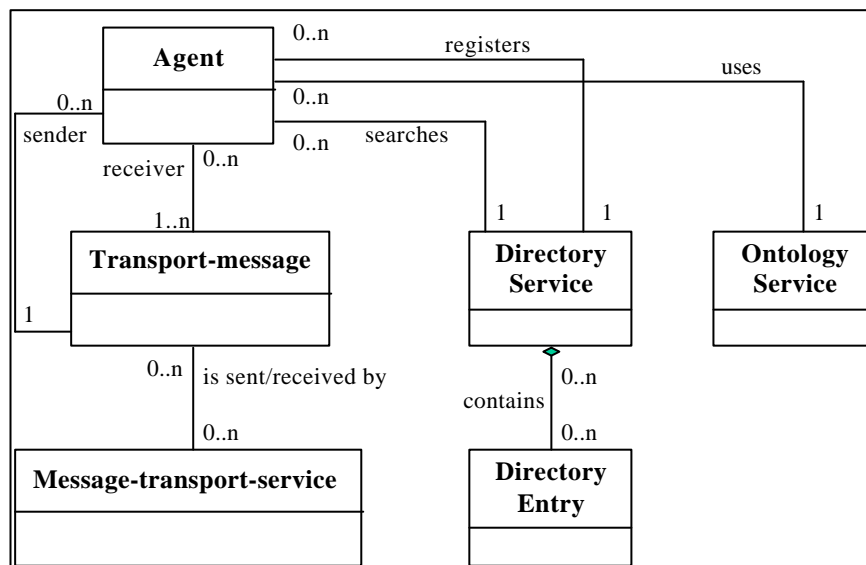


Fig 1. Basic agent relationships

As seen from figure-1, agents communicate using transport-messages, which are sent or received by message transport service. Each transport-message includes a message, which is written in FIPA agent-communication-language (ACL) and the content of the message is expressed in one of the FIPA content languages. Message-transport-service is not critical from the modeling methodology point of view, since any FIPA-compliant agent framework implements this service as a part of an agent's internal architecture [3] [15]. But defining interactions between the agents is one of the most critical activities of the MAS development. In FIPA based agent systems, agent interactions are specified using the pre-defined FIPA interaction protocols. Each interaction protocol defines a pre-agreed message exchange between the agents. For example, FIPA Query Interaction Protocol, which defines how one agent may query another agent, is shown in figure 2. Like all FIPA interaction protocols, this protocol uses FIPA ACL communicative acts (such as query-if, query-ref, inform etc.) within a well defined semantic. Hence, each interaction protocol can be considered as a

generic pattern where each participating agent plays a specific role within the defined semantic to solve a particular recurring problem. Since we accept interaction protocols as generic patterns, we use the concepts of the interaction protocol and the interaction pattern interchangeably within the paper.

SABPO tries to identify required interaction protocols based on the system requirements during the analysis phase and then fulfills all HPA ACL messages within the protocol during the design phase. Interaction patterns are considered as an extension point for SABPO methodology. When a MAS developer faces a new situation that can't be handled with existing patterns, she/he has to define a new pattern using the FIPA standards. As a result, SABPO drives the extension of FIPA standards by defining pattern identification activity within the process.

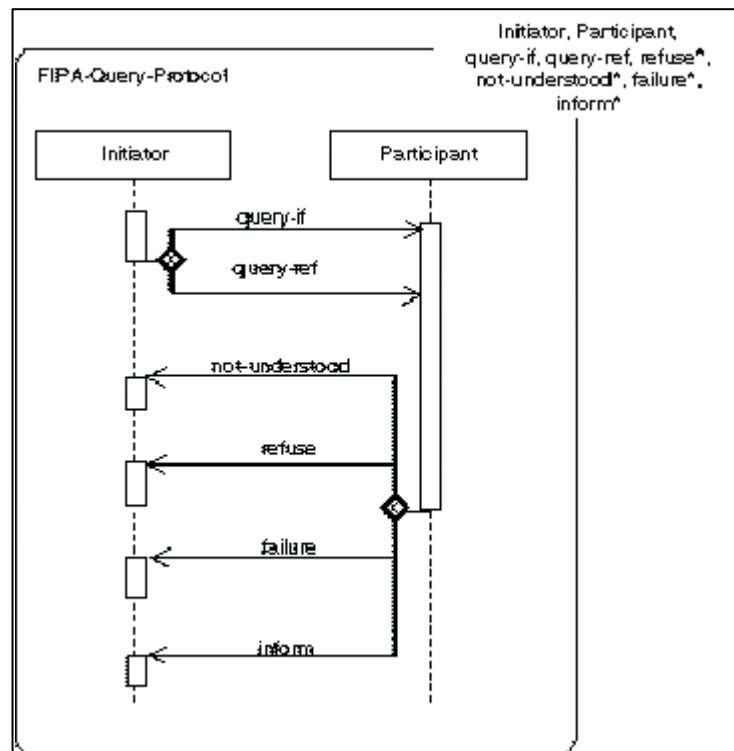


Fig2. FIPA Query Interaction Protocol.

Directory service, which is a mandatory component of FIPA abstract architecture, must be analyzed carefully during the modeling process, since this service is the heart of any multi-agent organization running in an open environment. A directory service provides a shared information repository, in which agents may publish their capabilities and in which they may search for other agents of interest. A directory service provides the following standard services:

- * An agent may register a directory-entry with the directory service.
- * An agent may modify a directory-entry that has been registered with a directory service.
- * An agent may delete a directory-entry from a directory service.
- * An agent may query a directory service to locate an agent of interest.

In addition to these mandatory elements, FIPA abstract architecture specification defines some optional elements. Ontology is one of these elements. Ontologies provide a vocabulary for representing and communicating knowledge about some topic and a set of relationships and properties that hold for the entities denoted by that vocabulary. Although ontologies are defined as optional elements in the FIPA specification, they are one of the most important issues to be considered during the development of MASs running in open environments. Because, in an open environment, agents can join to or leave from the agent system unpredictably; hence, newly joined agents learn how to communicate with other agent of the organization from the declaratively represented explicit ontologies. Hence, SABPO accepts the ontology service as a mandatory element. That is why ontology service is defined as a part of abstract architecture in Fig1.

Usage of explicit ontologies are defined in another FIPA specification [8]. This specification introduces a new agent called as Ontology Agent (OA). OA provides some ontology service to an agent organization. As well as the other agents, the OA registers its services with the directory service. The OA also registers the list of the maintained ontologies and its translation capabilities, in order to allow other agents to query the directory service for the specific OA that manages a specific ontology. OA provides the following services:

- * Helping a FIPA agent in selecting a shared ontology for communication.
- * Creating and updating a specific ontology, or only some terms of the ontology.
- * Responding to queries for relationships between terms or between ontologies. (Six relationships are defined in the specifications: ontologies can be identified, equivalent, extension of the other, weakly-translatable, strongly-translatable or approximately-translatable. OA stores the defined relationships between the ontologies and agents in the organization can query this knowledge.
- * Translating expressions between different ontologies (of course, ontologies must be translatable and translation rules must be defined by an authority.)

Agents query the OA to get the desired service using the FIPA interaction protocols where content of the each ACL within the protocol is written in FIPA-Meta-Ontology.

As an conclusion, SABPO identifies tasks of all abstract elements of the architecture and FIPA based interaction patterns between these elements, to realize a concrete architecture, which satisfies the system's requirements. SABPO only defines the analysis and design phases of the development process. Models created and activities involved in these phases are described below.

2.1 Analysis Phase

SABPO analysis phase exploit two different models; the role model and the interaction model as in the Gaia methodology. But HPA abstract architecture and FIPA interaction patterns are taken into consideration during the modeling process.

Role Model mainly defines the roles of the organization and responsibilities of these roles to satisfy the organization's global goals. These roles can be considered as abstract agent elements of the FIPA's abstract architecture specification. Then, SABPO introduces two new roles based on the abstract architecture specification. These roles are "Directory Service Provider" and "Ontology Service Provider". Services given by the directory service and OA of the abstract architecture are accepted as those roles' responsibilities. In SABPO, role model is documented using the Gaia's modeling style.

Interaction Model defines the interaction protocols between the agents. To create a FIPA-compliant architecture, "Directory Service Provider" and "Ontology Service Provider" roles participate in interactions using the interaction patterns defined in the FIPA specifications. These interactions are documented using Agent-UML [2] notation and FIPA's communication acts are explicitly indicated on the interaction diagrams. Inclusion of a directory service provider and an ontology service provider forces the developers to make same decisions in the analysis phase such as:

- * Name and general objectives of the global ontology must be identified.
- * Relationships between the global ontology and if there are any other known ontologies in the domain, then they must be identified.
- * If more than one ontology is identified for the organization, then the specific ontology used for each role must be identified.
- * Services given by the directory service provider and the ontology service provider roles during the interactions must be specified according to the related FIPA standards.

2.2 Design Phase

In SABPO design phase, a FIPA based concrete architecture is constructed by transforming the abstract entities identified in the analysis phase to the concrete ones using FIPA specifications. Design phase generates three models which are named as, the ontology model, the agent model and the detailed interaction model.

Ontology Model extends the ontology knowledge derived in the analysis phase. The global ontology can be modeled using any of the well known ontology modeling languages such as DAML [10], extended UML [1] [4], etc. Then, the mappings between this model and FIPA-Meta-Ontology specification are defined to make the global ontology, which is managed by the OA, accessible from outside. If there are any known ontologies in the domain, the mappings between these ontologies and the global ontology are also defined to make OA capable of providing translation service to the organization. At this point, organisational rules are defined using the

Zambonelli, Jennings and Wooldridge's study's approach and stored as a part of the ontology knowledge.

Agent Model defines the agent types and assigns the roles defined in the analysis phase to the agent types. This model also defines the services of each agent according to the responsibilities of the roles assigned to the agents. Hence, this model combines the "Agent Model" and "Service Model" of the Gaia methodology. "Ontology Service Provider" role and "Directory Service Provider" role and their defined services are assigned to both the ontology agent and the directory facilitator agent, which are the agents that are used for these roles in FIPA specifications. Since each agent must adapt itself to the ontologies used in the organization, mapping between agent's internal knowledge and system's ontologies are defined for each agent. This mapping differs for different types of agents within the organization. For example, agents playing the role of information providers need a mapping between its local knowledge and the ontologies they support. On the other hand, agents playing the role of information requesters just need proper user interface definitions based on the ontologies they support. These mappings are documented using the extensible style sheet language for transformations (XSLT) [18] standard and stored for each agent.

Detailed Interaction Model maps the interaction patterns identified in the analysis phase to the FIPA specifications. For this purpose, each message between the agents is rewritten using the FIPA ACL specification. The content of each message is also written in one of the content languages supported by FIPA such as FIPA-RDF.

At the end of the design phase, all of the concrete entities of the organization are ready to be implemented using any of the FIPA-compliant agent frameworks, since they totally conform the specifications of the FIPA.

3. Applying the SABPO Methodology

To illustrate how SABPO is applied to engineer a real system, we will use a part of the realistic B2B application as an example. This application is used by a developer company, which gathers required parts from different suppliers. Hence, the application should support the following functionalities:

- * It should support the developer company users to define the desired part's knowledge
- * It should find the potential suppliers of the desired parts.
- * It should send a request to collect the information about the desired parts to the suppliers and shows the results to the developer company's interested users.

Because of the space limitation, only analysis phase of the SAPBO is illustrated for the requirements listed above.

Role Model "Developer Company" role and "Supplier" role can be identified easily from the requirements. Following the Gaia methodology "Developer Company" role must have a "part definition" activity, which helps the users to define the desired parts. A "request" protocol must also be added to the "Developer Company" role to send the part request to the proper suppliers. On the other hand,

“Supplier” role must have a “get part knowledge” activity and an “inform” protocol to return desired part knowledge to the “Developer Company” role.

However, there are some open questions such as how the "Developer Company" role will identify the right suppliers and how it will create the part knowledge that is understandable by suppliers. This is the point where "Directory Service Provider" and "Ontology Service Provider" roles come in to the stage. To locate the right supplier, "Developer Company" role must request this knowledge from the "Directory Service Provider" role since each supplier should advertise itself to the directory and the directory service provides such a service according to the FIPA abstract architecture specification.

To create understandable part knowledge, it is certain that this organization must have a global explicit ontology. Let's call this ontology as "part-ontology". "Supplier" role must support this ontology and advertise it to the directory as the supported ontology. "Developer Company" role must also support this ontology to create sharable part knowledge. As a summary, "Developer Company" role must use a "request" protocol to get the right suppliers that support the "part-ontology" from the directory service and each agent must have the standard "advertise" protocol to advertise itself (including the ontology it supports) to the "Directory Service Provider".

Interaction Model In the interaction model, all protocols identified in the role model are defined using the Agent-UML[2]. Also, the general interaction mechanism is visualized using the collaboration diagram of the Agent-UML. Figure-3 shows the collaboration diagram of the role's interaction.

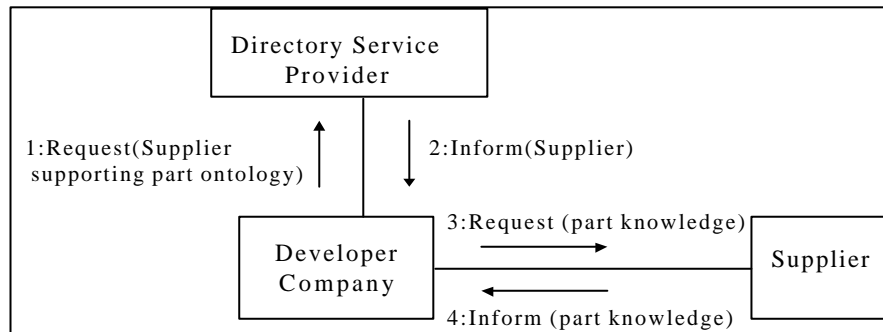


Fig 3. Collaboration diagram of the role's interaction.

The information flow is given in detail below:

1. "Developer Company" requests the possible suppliers supporting the "part ontology" from the "Directory Service Provider".
2. "Directory Service Provider" informs the "Developer Company" by returning the possible suppliers.
3. "Developer Company" requests from the supplier for the desired part using the "part ontology".
4. Suppliers return the requested part information to the "Developer Company".

Now, we will change the situation and illustrate how SABPO handles different conditions using the FIPA specifications. Let us assume that there is another ontology in this domain supported by some suppliers and call this ontology as "part1 ontology". "Developer Company" supports only the "part ontology", but it also knows that there are some suppliers around supporting different ontologies. To handle this situation, there must be a relationship between these ontologies and the "Ontology Service Provider" should be able to perform the translation service between any two ontologies as defined in the FIPA Ontology Service Specification [8]. The collaboration diagram to handle the defined situation is shown in figure-4.

Explanation of the information flow is given below:

1. "Developer Company" requests all suppliers in the domain from the "Directory Service Provider".
2. "Directory Service Provider" returns all possible suppliers.
3. "Developer Company" understands that there are some suppliers, which support "part1 ontology" and ask the "Directory Service Provider" if there is any "Ontology Service Provider", which provides a translation service between these ontologies.
4. "Directory Service Provider" returns the "Ontology Service Provider" which supports the translation service.
5. "Developer Company" requests for the translation of the query from "part ontology" to "part1 ontology".
6. "Ontology Service Provider" returns the translated query.
7. "Developer Company" requests for the part knowledge from suppliers using the proper ontology for each supplier.
8. Suppliers return the part knowledge.

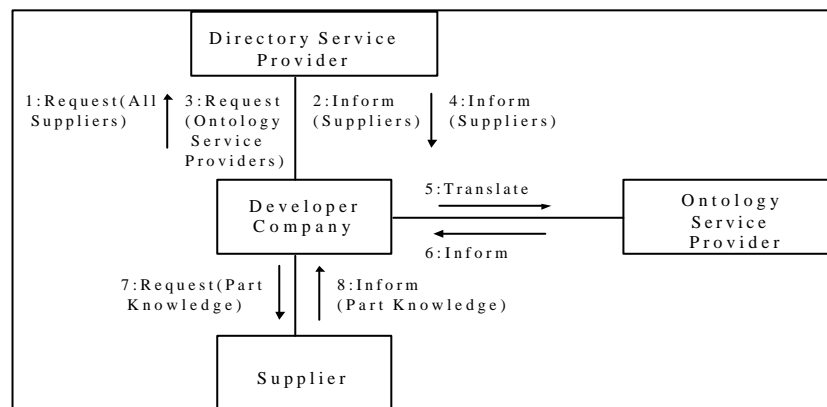


Fig 4. Collaboration diagram for the role's interaction for a different situation.

This new situation, of course, creates new activities for each agent and new interaction protocols as shown in figure-4.

As a summary, SABPO first defines the roles of the organization and their responsibilities in the analysis phase. These individual roles are accepted as the abstract agent element of the FIPA's abstract architecture. Then, it fully exploits the abstract architecture by taking the directory and the ontology services as roles and defining the interaction protocols between the roles to satisfy system's requirements.

4. Ontology Inception Pattern

MASs aim to model complex applications running on complex environments. Hence, MAS developers may always face new situations, which have not been defined in the FIPA specifications yet. If the solutions found for these new situations are general and applicable to any MAS facing similar situations, then these solutions should be added to the FIPA specifications. By this way, an extensible catalogue of solutions patterns will be formed as part of FIPA standards. This approach has been applied successfully by the software community in creating a pattern catalogue for solving recurring design problems in an extensible way [9]. We believe that pattern identification must be taken as one of the most important research direction of the agent community. In this section, a new interaction pattern is introduced for multi-agent systems and a documentation style is proposed to define the identified patterns.

4.1 Motivation and Interaction Mechanism

To document the MAS interaction patterns, we define the pattern from three different perspectives. First, motivation behind the pattern identification is introduced. Then, interaction mechanism of the participating agents is defined according to the FIPA standards. Finally, guidelines are defined to make the pattern FIPA-compliant.

Motivation: To understand the motivation behind the "Ontology Inception" pattern, let us look at the open environment from the ontological perspective and assume the following characteristics:

- * There are a large number of domain dependent ontologies, which may change at run-time.
- * At any time, new ontologies may be added to the MAS. Also, some existing ontologies may decided not to be used any more.

In such an environment, some questions easily come to mind. For example, one question is about how agents provide the information related with newly added ontologies. In the ontology service specification, OA only provides translation service, which may be used for this situation. But, any agent that supports different ontologies can join the organization at any time. Who will define the mappings between these new ontologies and existing ones? How will individual agents begin to support these new ontologies? It is clear that translation service is not enough to support such an environment. Hence, "Ontology Inception" is a solution pattern that makes agent organization adaptable to the open environment.

Interaction Mechanism: Interaction mechanism defines the interaction protocol between the agents that collaborate to solve the problem at hand. Also, activities of the participating agents are identified and modeled as services using Gaia style. The interaction mechanism is illustrated in figure-5 using a collaboration diagram.

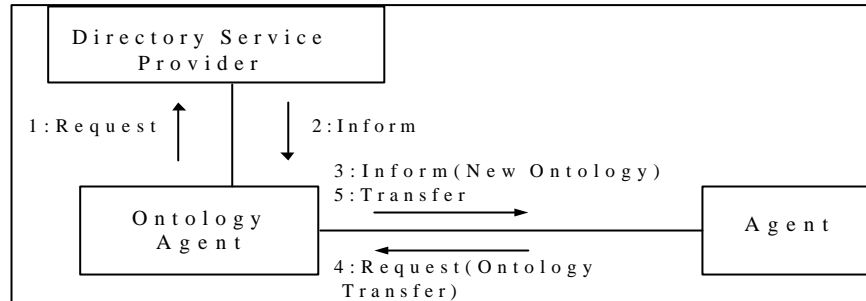


Fig 5. Interaction mechanism of "Ontology Inception" pattern.

Information flow between the participating agent is described below:

1. Ontology agent requests the directory facilitator agent to find the agent within the organization.
2. Directory facilitator sends the list of active agents to the ontology agent.
3. Ontology agent informs each agent that there is a new ontology in the system.
4. Interested agents send a request to the ontology agent for ontology transfer.
5. Ontology agent transfers the ontology to the interested agents.

It can be seen from the interactions that the agent role needs an activity to define the required mappings between the transferred ontology and its local knowledge. This activity is called as "localize" and defined as the service of the agent role. Also, ontology agent needs a new communication act to transfer a specific ontology to the interested agent. This new communication act is named as "transfer" and defined in Agent-UML.

Guidelines for FIPA-compliance: The following guidelines are defined to conform the "Ontology Inception" pattern to the FIPA specification:

- * A new service should be defined for the ontology agent to transfer the required ontology.
- * FIPA-Meta-Ontology specification should be extended with required semantic to support the transfer service.
- * FIPA standards committee should decide either to define a new communicative act for transfer protocol or using inform act inserting the required knowledge to the content of the message.

4.2 Run-Time Behavior of the Ontology Inception Pattern

Sometimes it can be difficult to understand the internal behaviour of participating agents when they involve in a complex interaction pattern. "Ontology Inception" also requires a definition of how the participating agents perform a "localize" activity. The "localize" activity needs different behaviour for different types of agents. For example, an information provider type of agent implements it by defining a mapping between the transferred ontologies and its local knowledge. On the other hand, an information requester type of agent implements it to create user interfaces for the transferred ontology. Hence, sometimes it can be helpful to discuss the details of

internal behaviour of the participating agent to make the pattern more understandable for the developers.

To transfer ontologies, first of all, it must be decided what representation to use for modeling the ontologies. In agent literature, description logic based languages [6] [10] are widely used for ontology representation. Also, there are some work which use UML for agent based systems, present an ontology representation language based on a subset of UML together with its associated Object Constraint Language (OCL) or propose mechanisms for reasoning on UML class diagrams using description logic [1] [4]. Similarly, we use UML for representing ontologies [5].

After deciding which ontology modeling language to use, the ontologies are represented in the selected ontology modeling language. Then, the represented ontologies are transferred by the ontology agent. Hence, developers have to decide how to import the ontologies in the content of FIPA ACL messages. One possible solution is to use FIPA RDF content language, since most of the ontology modeling languages are based on RDF and use RDF/XML as a syntax to transport the modeled ontologies.

Each participating agent can issue a transfer request for a specific ontology to the ontology agent, whenever needed. When the requested ontology is transferred from the ontology agent to the participating agent, the participating agent has to extract the content of the arrived FIPA ACL message and parse the incoming ontology. If the participating agent is an information provider type of agent, then it should visually show the ontology model and its local information model so that a mapping can be defined. This mapping will then be saved as part of the wrapping knowledge. If it is an interface type of agent, then it should generate a user interface so that the users can enter queries related with a specific ontology and can see the results

5. Conclusion

In this paper, a new methodology, which uses organizational metaphor as the basis and integrates this metaphor to the FIPA standards, is introduced. In addition, a new interaction pattern, which is called as "ontology inception", is defined. The proposed methodology and the "ontology inception" pattern have been used in the development of a multi-agent system infrastructure for software component market-place [5], which has all the basic the characteristics of open environments.

6. References

- [1] Baclawski, K., Kokar, M.K., Kogut, P.A., Hart, L., Smith, J., Holmes, W.S., Letkowski, J. and Aromson, M.L. Extending UML to support ontology engineering for the semantic web. The Unified Modeling Language, Modeling Languages, Concepts and Tools, 4th International Conference, Toronto, Canada, LNCS 2185 Springer, 342-360, 2001.
- [2] Bauer, B., Müller, J.P. and Odell, J. Agent UML: A formalism for specifying multiagent interaction. Agent Oriented Software Engineering, Paolo Ciancarini and Michael Wooldridge eds., Springer-Verlag, Berlin, pp.91-103, 2001.
- [3] Bellifemine, F., Poggi, A., and Rimassa, G. Developing multi-agent systems with a FIPA-compliant agent framework. *Software Practice. and Experience*, 31:103-128, 2001.
- [4] Cranefield, S., Haustein, S., and Purvis, M. UML-based ontology modelling for software agents. Proceedings of the Workshop on Ontologies in Agent Systems held at the 5th

International Conference on Autonomous Agents, 2001, Montreal, Canada. Available on line at <http://cis.otago.ac.nz/OASWorkshop>.

- [5] Erdur, R. C. and Dikenelli, O. A multi-agent system infrastructure for software component market-place: an ontological perspective, *ACM Sigmod Record*, Vol:31, Number:1, March, 2002.
- [6] Fensel, D., Horrocks, I., Van Harmelen, F., Decker, S., Erdmann, M. and Klein, M. OIL in a nutshell. In the proceedings of the workshop on applications of ontologies and problem solving methods, 14th European Conference on Artificial Intelligence, Germany, 2000.
- [7] FIPA(a). FIPA XC00001I: FIPA abstract architecture specification. <http://www.fipa.org>.
- [8] FIPA(b). FIPA XC00086C: FIPA ontology service specification. <http://www.fipa.org>.
- [9] Gamma, E., Helm, R., Johnson, R. and Vlissides J. *Design patterns*. Addison Wesley, Reading (MA), 1995.
- [10] Hendler, J. and McGuinness, D. The DARPA agent markup language. *IEEE Intelligent Systems*, 15, No.6:67-73, 2000.
- [11] Iglesias, C., Garijo, M. and Gonzales, J. A survey of agent-oriented methodologies. In A.S. Rao J.P. Muller, M.P.Singh, editor, *Intelligent Agents IV (ATAL98)*, LNAI. Springer- Verlag, 1999.
- [12] Kendall, E.A. Agent software engineering with role modeling. In *Proceedings of the 1st International Workshop on Agent Oriented Software Engineering*, volume 1957 of LNCS. Springer Verlag, 2000.
- [13] Kinny, D. and Georgeff, M. A methodology and modeling technique for systems of bdi agents. In *Workshop on Modeling Autonomous Agents in a Multi-agent world*, LNAI 1038, pages 56-71. Springer Verlag, 2000.
- [14] Ominici, A. SODA: Societies and infrastructures in analysis and design of agent-based systems. In Ciancarini, P. and Wooldridge, M. (eds). *Proc. 1st Int. Workshop on Agent-Oriented Software Engineering (AOSE 2000)*, Limerick, Ireland, June, 2000. Volume 1957 of LNCS, Springer-Verlag, Berlin, 2001.
- [15] Poslad, S., Buckle P., and Hadingham, R. The FIPA-OS agent platform: open source for open standards. in the *Proceedings of the 5th International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-agents*, 2000 UK. Available at <http://www.fipa.org/resources/byyear.html>
- [16] Wood, M., DeLoach, S.A. and Sparkman, C. *Multiagent system engineering*. International Journal of Software Engineering and Knowledge Engineering, 2001.
- [17] Wooldridge, M., Jennings, N.R. and Kinny, D. The Gaia Methodology for agent oriented analysis and design. *Journal of Autonomous Agents and Multi-Agent Systems*, 3(3):285-312, 2000.
- [18] XSLT specification, <http://www.w3.org/TR/xslt>.
- [19] Zambonelli, F., Jennings, N., and Wooldridge, M. Organisational rules as an abstraction for the analysis and design of multi-agent systems. *Int. Journal on Software Engineering and Knowledge Engineering*.