Metamodeling of Semantic Web Enabled Multiagent Systems

Geylani Kardas[1], Arda Goknil[2], Oguz Dikenelli[2], and N. Yasemin Topaloglu[2]

[1]Ege University, International Computer Institute, 35100 Bornova, Izmir, Turkey
geylani.kardas@ege.edu.tr
[2]Ege University, Department of Computer Engineering, 35100 Bornova, Izmir, Turkey
{arda.goknil, oguz.dikenelli, yasemin.topaloglu}@ege.edu.tr

Abstract. Several agent researchers are currently studying agent modeling and they propose different architectural metamodels for developing Multiagent Systems (MAS) according to specific agent development methodologies. When support for Semantic Web technology and its related constructs are considered, agent metamodels should include meta-entities to model MASs which work in semantic web environment. In this paper, we introduce an agent metamodel to define the required constructs of a Semantic Web enabled MAS in order to provide semantic capability modeling and interaction of agents both with other agents and semantic web services. We first give a conceptual MAS architecture to identify new constructs in addition to constructs of a traditional MAS and then we propose a metamodel including the first-class entities required by such a conceptual architecture.

1 Introduction

A software methodology is typically characterized by a modeling language and a software process as mentioned in [1]. Same is valid for the Multiagent System (MAS) methodologies from the perspective of agent oriented software engineering (AOSE) paradigm. Again given in [1], we use agent modeling languages for the description of models, defining the elements of the model together with a specific syntax (notation) and associated semantics. On the other hand, software process defines development activities and their interrelationships. Each methodology naturally defines a specific metamodel based on artifacts of the process and its architectural constructs. However, a metamodel can be independent from a specific methodology and define architectural constructs and their relations for a specific application area or domain.

In MAS community, there are various studies which define metamodels of specific methodologies like Gaia, Adelfe, PASSI [3] and SODA [9]. Also, Foundation for Intelligent Physical Agents (FIPA) has an on-going effort for developing a notation to express relationships between agents, agent roles and agent groups in a MAS. Collaborating with Object Management Group's (OMG) Agent SIG, FIPA Modeling Technical Committee proposes a metamodel called Agent Class Superstructure Metamodel (ACSM) [7] which is based on - and extends - UML

2.0 superstructure. It presents a superstructure specification that defines the user-level constructs required to model agents, their roles and their groups [10]. However, we believe that a significant deficiency exists in those noteworthy agent modeling studies when we consider their support on Semantic Web [2] technology and its constructs.

Semantic Web evolution has doubtlessly brought a new vision into agent research. This *Second Generation Web* aims to improve WWW (World Wide Web) such that web page contents are interpreted by using ontologies. It is apparent that the interpretation in question will be realized by autonomous computational entities - so agents - to handle semantic content on behalf of their human users. Surely, Semantic Web environment has specific architectural entities and a different semantic which must be considered to model a MAS in this environment. Therefore, modeling techniques and development process should support this new environment by defining new meta-entities and meta-structures.

In this study, we introduce an agent metamodel aiming to define required constructs of a Semantic Web enabled MAS in order to provide semantic capability modeling and interaction of agents. The interaction in question involves both communications of an agent: agent-to-agent and agent-to-semantic entity (e.g. a semantic web service [14]). Hence, we first give a conceptual MAS architecture to identify new constructs in addition to constructs of a traditional MAS. Then we propose a metamodel including the first-class entities required by such a conceptual metamodel.

The paper is organized as follows: Section 2 discusses about the constructs of a conceptual architecture for a MAS which is capable to work on the Semantic Web environment. Section 3 introduces our proposed metamodel. This metamodel is the first step to incorporate a model driven approach to the development of MASs. Finally, conclusion and future work are given in Section 4.

2 A Conceptual Architecture for Semantic Web Enabled MASs

We need to define a conceptual architecture for semantic web enabled MASs in which autonomous agents can also evaluate semantic data and collaborate with semantically defined entities such as semantic web services by using content languages. Our proposed conceptual architecture for such MASs is given in Figure 1.

The architecture defines three layers: *Architectural Service Layer*, *Agency Layer* and *Communication Infrastructure Layer*. A group of system agents provides services defined in the Architectural Service Layer. Every agent in the system has an inner agent architecture described in the Agency Layer and they communicate with each other according to the protocols defined in the Communication Infrastructure.

Semantic web agents are agents which are initiated by using the platform architecture and able to use semantic services within the service layer. In Architectural Service Layer, services (and/or roles) of semantic web agents inside the platform are described. All services in the Architectural Service Layer use the capability of the Agency Layer. Besides domain specific agent services, yellow page and mediator services should also be provided.

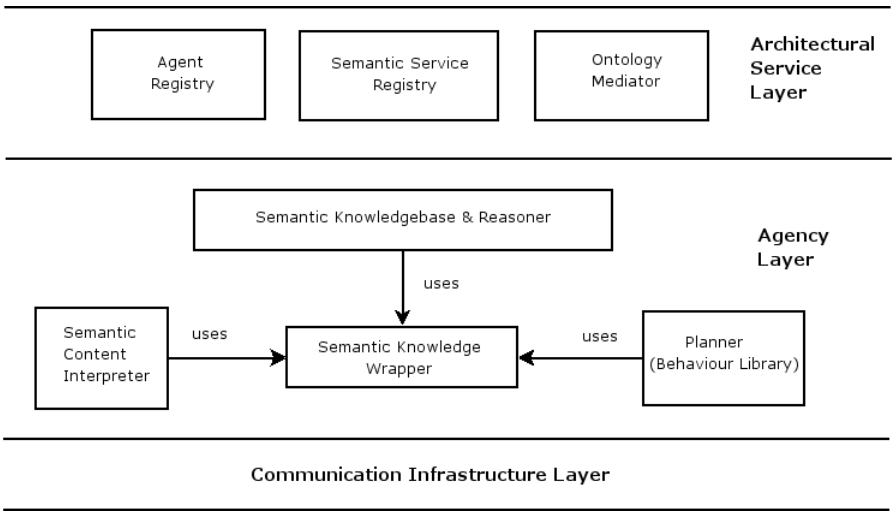

Fig. 1. The conceptual architecture for Semantic Web enabled MASs.

Agent Registry is a system facilitator in which capabilities of agents are semantically defined and advertised for other platform members. During their task executions, platform agents may need other agents' services. Hence, they query on this facilitator to determine relevant agents for interaction.

No matter it is FIPA-compliant [6] or not, a traditional MAS owns one or more registries which provides yellow page services for system's agents to look for proper agent services. Of course registries mentioned above are not simple structures and mostly implemented as directory services and served by some platform specific agents. For example there is a mandatory agent called *directory facilitator (DF)* in FIPA abstract architecture specification on which agent services are registered [6]. When an agent looks for a specific agent service, it gathers supplier data (agent's name, address, etc.) of the service from the DF and then it begins to communicate with this service provider agent to complete its task. However, capability matching becomes complex and has to be redefined when we take into consideration of MASs on semantic web environment. In case of agent service discovery in such systems, we should define semantic matching criteria of service capabilities and design registration mechanisms (directory services) of agent service specifications according to those criteria. That

makes matching of requested and advertised services more efficient by not only taking into consideration of identical service matching: New capability matching will determine type and degree of relation between two services (requested and advertised) *semantically*. Hence, the conceptual architecture includes an agent registry to provide capability matching on agent services.

On the other hand, agents of the platform may also need to interact with Semantic Web Services which are web services with semantic interface to be discovered and executed. To support interoperability and automatic composition of Web services, capability of services are defined in service ontologies such as OWL-S [12] and WSMO [15]. In our approach, those service capabilities should also be advertised on proper registries to provide dynamic discovery and execution of the services by agents. Hence, we define a conceptual entity called *Semantic Service Registry* in the proposed architecture. This registry can also be modeled as a service matchmaker in which semantic interfaces of the platform's semantic web services are advertised to be discovered by the agents. Considering OWL-S services, agents may query on this facilitator by sending its requested semantic service OWL-S profile to the facilitator. The facilitator (or matchmaker) performs a semantic capability matching between the given request and advertised profiles and informs the agent about suitable services. Then the agent may interact with those services to complete its task. Engagement and invocation of the semantic web service is also performed according to its semantic protocol definitions.

A semantic web enabled agent interacts with agents within the different organization(s) and semantic web services may use knowledge sources handled by the different knowledgebase(s) and/or peer system(s). In such environment, it is obvious that, there exist more than one ontology and different entities may use different ontologies. So, there should be another architectural service in which translation and mapping of different ontologies are performed. We call this service as *Ontology Mediator* and it may be provided by one or more agents within the MAS. Agents which provide this conceptual service, may handle the translation request(s) using the pre-defined mapping knowledge introduced through a specific user interface. Through the usage of the ontology translation support, any agent of the platform may communicate with MAS and/or services outside the platform even if they use different ontologies.

The middle layer of the architecture is the Agency which includes inner structural components of Semantic Web enabled agents. Every agent in the system has a *Semantic Knowledgebase* which stores the agent's local ontologies. Those ontologies are used by the agent during his interaction with other platform agents and semantic web services. Evaluation of the ontologies and primitive inference are realized by the *Reasoner*.

Semantic Knowledge Wrapper within the Agency provides utilization of above mentioned ontologies by upper-level Agency components. For example, during his task execution, the agent may need object (or any other programmatic) representation of a specific ontology individual. Or the content interpreter requests a query on one of the ontologies to reason about something. To meet up such

requirements, the Semantic Knowledge Wrapper of the agent may form graph representations of the related ontologies within the runtime environment of the agency. One such good example for the wrapper is the proper application of the JENA [8] framework in agent internal architecture and is discussed in [5].

The *Planner* of the Agency Layer includes necessary reusable plans with their related behavior libraries. The reusable agent plans are composed of tasks which are executed according to the agent's intentions. The planner is based on reactive planning paradigm e.g. HTN (Hierarchical Task Network) planning framework presented in [13]. In reactive planning a library of general pre-defined (may be defined at compile time) plans is provided to agent and the agent performs one or more of these plans in response to its perceptions of the environment [4].

Semantic Content Interpreter module uses the logical foundation of semantic web, ontology and knowledge interpretation. During its communications, the agent receives messages from other agents or semantic services. It needs to evaluate received message content to control its semantic validity and interpret the content according to its beliefs and intentions. Necessary content validity and interpretation takes place in this module.

The bottom layer of the architecture is responsible of abstracting the architecture's communication infrastructure implementation. For example, it may be an implementation of FIPA's Agent Communication and Agent Message Transport specifications [6] to handle agent messaging. Hence, the layer transfers any content (including semantic knowledge) by using FIPA ACL and transport infrastructure. Physical communication may take place via well-known HTTP-IIOP (Internet Inter-ORB Protocol). However, the content language within the message infrastructure is crucial.

3 Proposed Metamodel for Semantic Web Enabled MASs

In this section, we introduce an agent metamodel superstructure to define elements and their relationships of a Semantic Web Enabled MAS depending on the previously discussed conceptual architecture. Figure 2 depicts this core metamodel for Semantic Web enabled agent platforms.

Semantic Organization is a composition of *Semantic Web Agents* which is constituted according to organizational roles of those agents.

A *Semantic Web Agent* is an autonomous entity which is capable of interaction with both other agents and semantic web services within the environment. Interaction processes are executed according to predefined semantic communication protocols. Conceptually, a Semantic Web Agent can be considered as a specialization of the commonly-known software agent entity within internal agent structure and communication content perspectives.

It should be noted that a Semantic Organization should be implemented as only a composition of Semantic Web Agents. The organization does not involve other semantic entities such as semantic web services. However, it includes organizational roles and those roles are played by its agents. Taking into consideration of those organizational roles, a Semantic Web Agent may be a member

of different Semantic Organizations. That means, one agent may play more than one *Role* and one Role may be played by many Semantic Web Agents within the Semantic Organization context.

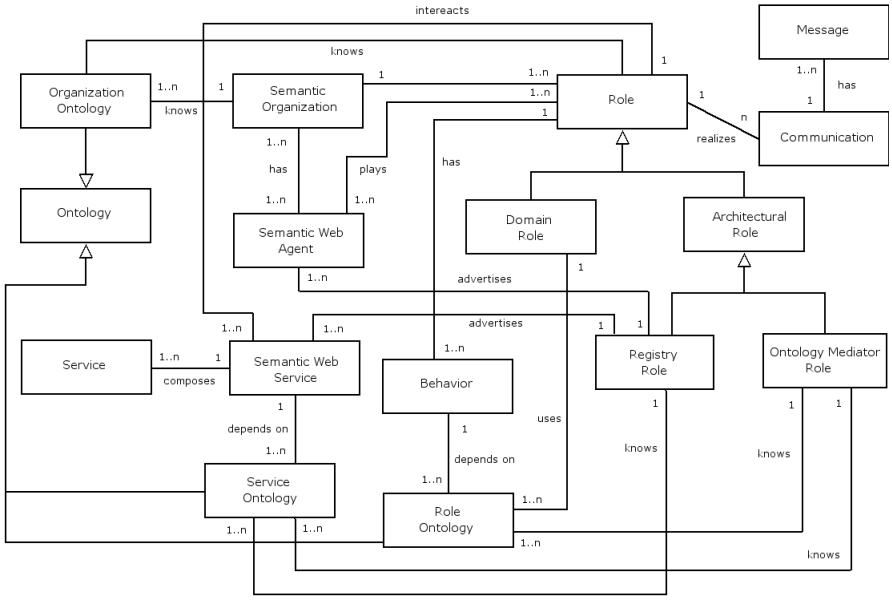

Fig. 2. The core metamodel for Semantic Web enabled MASs.

Roles provide both the building blocks for agent social systems and the requirements by which agents interact as it has been remarked in [10]. We believe that the same is true for roles played in Semantic Web enabled agent environments. However, this general model entity should be specialized in the metamodel according to task definitions of architectural and domain based roles: An *Architectural Role* defines a mandatory Semantic Web enabled MAS role that should be played at least one agent inside the platform regardless of the organization context whereas a *Domain Role* completely depends on the requirements and task definitions of a specific Semantic Organization created for a specific business domain.

Some of the organization agents must play architectural roles to provide services defined in Architectural Service Layer of the conceptual architecture for other agents. Hence two specialization of the Architectural Role are also defined in the metamodel: *Registry Role* and *Ontology Mediator Role*. Registry Roles are played by one or more Semantic Web Agents to provide yellow page services for other agents of the organization. Within the scope of the role, they may store capability advertisements of Semantic Web Agents or Semantic Web Services. So,

other agents may discover any needed service according to semantic capabilities of services by interacting with those facilitator agents which play registry roles.

Ontology Mediator Role in the metamodel defines basic ontology management functionality that should be supported by ontology mediator agents as it is discussed in the previous section. A Semantic Web Agent that plays an Ontology Mediator Role should be aware of whole ontology knowledgebase of the Semantic Organization.

One Role is composed of one or more *Behaviors*. Task definitions and related task execution processes of Semantic Web agents are modeled inside Behavior entities. Proper arrangement of those behaviors constitutes agent roles.

According to played roles, agents inevitably communicate with other agents to perform desired tasks. Each *Communication* entity defines a specific interaction between two agents of the platform which takes place in proper to predefined agent interaction protocol. One Communication is composed of one or more *Messages* whose content can be expressed in a RDF based semantic content language.

One of the important first-class entities of the metamodel is *Semantic Web Service*. A Semantic Web Service represents any service (except agent services) whose capabilities and interactions are semantically described within a Semantic Web enabled MAS. A Semantic Web Service composes one or more Service entities. Each service may be a web service or another service with predefined invocation protocol in real-life implementation. But they should have a semantic web interface to be used by autonomous agents of the platform. It should be noted that association between semantic web agents and services is provided over agent Role entities in the metamodel. Because agents interact with semantic web services, depending on their roles defined inside the organization.

Like any other Semantic Web environment, a Semantic Web enabled MAS is inconceivable without ontologies. Hence, the proposed metamodel includes an *Ontology* entity and its required specializations. An Ontology represents any information gathering and reasoning resource for MAS members. Collection of the ontologies creates knowledgebase of the MAS that provides domain context. Specializations of the Ontology called *Organization Ontology*, *Service Ontology* and *Role Ontology* are utilized by related metamodel entities. For example semantic interface and capability description of services are formed according to Service Ontology and this ontology is used by Semantic Web Agents in order to discover and invoke Semantic Web Services.

4 Conclusion and Future Work

A metamodel for Semantic Web enabled MASs is introduced in this paper. The proposed metamodel provides first class entities to model MASs which work in semantic web environment. We believe that the metamodel in here helps to bridge the gap of modeling agent and semantic web constructs in a single environment by defining entities of a semantic web enabled MAS at first time.

The metamodel discussed in this study can be used as a basis to develop methodologies for Semantic Web enabled MAS development within the context of Model Driven Architecture (MDA) [11]. However, the current metamodel could not be considered as a complete Platform Independent Model (PIM) for semantic web enabled MAS. We aim to define MDA compatible models which elaborate the entities of our metamodel to generate source code of semantic web enabled MASs. Currently, we are studying on defining platform independent semantic web service constructs and mapping them into platform specific semantic web service languages e.g. OWL-S and WSMO.

References

1. Bauer, B. and Odell, J.: UML 2.0 and Agents: How to Build Agent-based Systems with the New UML Standard. Journal of Engineering Applications of Artificial Intelligence, Volume 18, issue 2, March 2005, Pages 141-157.
2. Berners-Lee, T., Hendler, J. and Lassila, O.: The Semantic Web, Scientific American, 284(5), (2001), pp:34-43.
3. Bernon, C., Cossentino, M., Gleizes, M., Turci, P. and Zambonelli, F.: A Study of some Multi-Agent Meta-Models. Agent-Oriented Software Engineering (AOSE) 2004, Lecture Notes in Computer Science, Springer, Berlin.
4. Dickinson, I. and Wooldridge, M.: Agents are not (just) web services: considering BDI agents and web services. The Workshop on Service-Oriented Computing and Agent-Based Engineering (SOCABE'2005), July 2005, Utrecht, the Netherlands
5. Dikenelli, O., Erdur, R. C., Kardas, G., Gümüs, O., Seylan, I., Gurcan, O., Tiryaki, A. M. and Ekinci, E. E.: Developing Multi Agent Systems on Semantic Web Environment using SEAGENT Platform. Engineering Societies in the Agents World VI, Lecture Notes in Artificial Intelligence, Springer-Verlag, Vol. 3963, pp. 1 -13.
6. FIPA (Foundation for Intelligent Physical Agents): FIPA Specifications, available at: http://www.fipa.org
7. FIPA Modeling TC: Agent Class Superstructure Metamodel, available at: http://www.omg.org/docs/agent/04-12-02.pdf, 2004
8. JENA - A Semantic Web Framework for Java, available at: http://jena.sourceforge.net
9. Molesini, A., Denti, E. and Omicini, A.: MAS Meta-models on Test: UML vs. OPM in the SODA Case Study. 4th International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS'05), Budapest, Hungary.
10. Odell, J., Nodine, M. and Levy, R.: A Metamodel for Agents, Roles and Groups. Agent-Oriented Software Engineering (AOSE) V, LNCS, Springer, Berlin, 2005.
11. Object Management Group (OMG): MDA Guide Version 1.0.1. Document Number: omg/2003-06-01 (2003)
12. The OWL Services Coalition: Semantic Markup for Web Services (OWL-S), 2004, http://www.daml.org/services/owl-s/1.1/
13. Sycara, K., Williamson, M., and Decker, K.: Unified information and control workflow in hierarchical task networks. In: Working Notes of the AAAI-96 workshop "Theories of Action, Planning, and Control". (1996)
14. Sycara, K., Paolucci, M., Ankolekar, A., and Srinivasan, N.: Automated discovery, interaction and composition of Semantic Web Services. Journal of Web Semantics, Elsevier 1, (2003) pp. 27-46
15. Web Service Modeling Ontology, http://www.wsmo.org/