Meta-Modeling Communication and Interaction inside MASs with Ontologies

Valentina Cordì, Viviana Mascardi
DISI, Università di Genova, Via Dodecaneso 35, 16146, Genova, Italy.

Abstract. The need of a unifying meta-model for Multi-Agent Systems (MASs) in general, and for the interaction and communication aspects involved in MASs in particular, has rapidly grown in the last three years, as demonstrated by many recent papers and events.
The main idea behind meta-modeling is that different Agent-Oriented Software Engineering (AOSE) methodologies and tools may turn out to be useful for engineering different aspects of the MAS, but no single methodology is good for everything. Taking advantage of reusable "components" - or "fragments" - of existing AOSE methodologies, and integrating them according to a common meta-model, is becoming a widely accepted approach.
This paper describes the communication and interaction portion of a "MAS meta-model ontology", developed taking six different AOSE methodologies and related meta-models into account. The ontology, designed following well-established criteria and implemented using the tool Protégé, is aimed at helping the MAS designer in finding the right method fragment to do the right thing, by answering queries such as "What is a communicative act, and which methodologies use it?", "What is a message according to the methodology XYZ?", "Which AOSE methodologies take Agent Interaction Protocols into account?".
Keywords. Ontological Analysis of Interaction and Communication, Agent-Oriented Software Engineering, MAS Meta-model

Introduction

Over the last few years, several methodologies for the development of MASs (Multi-Agent Systems) have been proposed. As stated in [1], it is widely accepted that a unique specific methodology cannot be general enough to be useful for everyone, and that some level of personalization is needed. For this reason, the Methodology Technical Committee of the Foundation for Intelligent Physical Agents (FIPA)[1] states that *"In order to reuse contributions coming from existing methodologies we will adopt the method engineering as the referring paradigm. In this context the development methodology is constructed by the developer assembling pieces of the process (method fragments) from a method base. In this way he/she could obtain the best process for his/her specific need/problem. We will take method fragments (the composing elements of the development methodology) from several existing methodologies."*

However, adopting a method engineering approach in the AOSE context is not a plain task. In the object-oriented context, building the method fragments, assembling the methodology with them, and following the methodology, all rely on a common denominator, the

[1] `http://www.fipa.org/activities/methodology.html`

universally accepted concept of object and the related meta-model of the object-oriented system[2]. The situation concerning the agent-oriented approach is quite different since there is not a commonly accepted definition of agent. Since a meta-model is a means of unifying concepts, the lack of a unique definition of agents and related concepts, causes a lack of a unique MAS meta-model definition, which in turn leads each methodology to have its own concepts and system structure.

In order to overcome this problem, in [1] C. Bernon, M. Cossentino, M. P. Gleizes, P. Turci and F. Zambonelli propose to start with an accurate analysis of the existing MAS meta-models, on which the methodologies are based. If the results of this analysis were available to the research community and easily accessible, the MAS designers could take advantage of them for selecting the elements that compose the MAS meta-model they will build.

In this paper, we describe our research work that is aimed at making a clear and principled analysis of the existing MAS meta-models, and making the analysis result easily accessible by designers. Our approach is based on the definition of a "MAS meta-model ontology" designed following the the "Ontology Development 101" guide [7], and implemented using Protégé[3]. We have started the implementation of the ontology from those fragments related with interaction and communication, thus discarding, for the moment, the issues that concern organization, services provision, goals, skills, capabilities, etc. The ontology *is not aimed at being used by software agents running inside a MAS*: instead, *it will provide a support to human designers that must build a MAS*, by helping them in better understanding which AOSE method fragments are already available, which is their intended meaning, and which AOSE methodologies provide them.

The paper is organized in the following way: Section 1 overviews the six existing MAS meta-models that we have considered for defining our ontology, concentrating on communication and interaction issues; Section 2 describes our ontology and the process followed to define it; Section 3 concludes and sketches the future directions of our work.

1 Existing MAS Meta-models

Over the past three decades, software engineers have derived a progressively better understanding of the characteristics of complexity in software. It is now widely recognized that interaction is probably the most important single characteristic of complex software [5]. Since most real-world applications contain many dynamically interacting components, each with its own thread of control, and engaging in complex coordination protocols, designing and developing new approaches and tools for engineering communicating and interacting systems in a correct and efficient way is currently an hot research topic.

In order to cope with the complexity of engineering nowadays applications, in the last five years many researchers have tried to define a meta-model of the main application components, for allowing different engineering methodologies to refer to the same concepts and system structure.

In this section we sketch the main features of six meta-models, most of which are layered on top of one existing AOSE methodology (Figure 1), and we devote a particular attention to the way interaction and communication are dealt with. In the next section, we will describe the ontology that integrates these six meta-models, and allows the MAS designer to find the right meta-model, and thus the right methodology fragment, to do the right thing.

[2]The meta-model of the most widely used object-oriented notation and methodology, UML [10], defines the complete semantics for representing object models using UML. It is defined in a metacircular manner, using a subset of UML notation and semantics to specify itself.

[3]`http://protege.stanford.edu/`

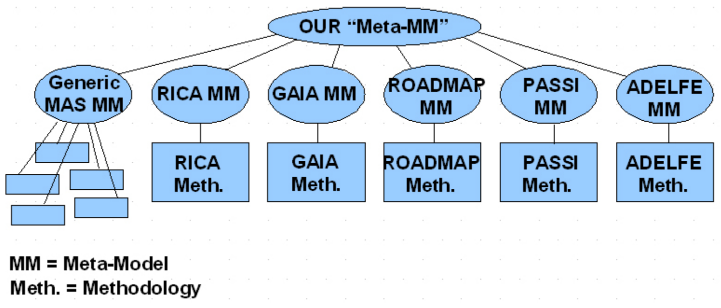


Figure 1: Existing methodologies and corresponding meta-models, and our "meta-meta-model"

A generic MAS Meta-model (2005): The generic meta-model introduced in [8] is the only one that does not build upon a single AOSE methodology, but borrows concepts from many existing ones. It has two layers - design-time and runtime -, and each layer may have two scopes - system related or agent related. Interaction is mainly modeled by the design-time system related classes, namely those classes concerned with roles, relationships between roles, relationship with message schemas; tasks, and their relationships with roles; agent definitions and relationships with roles. In the generic MAS meta-model, a role is defined as a specification of a behavioral pattern expected from some agents in a given system.

RICA (2004): The RICA meta-model [9] establishes a systematic link between the specifications of the organizational model and the Agent Communication Language (ACL) of a multiagent application: the specification of the organizational model comprises the definition of entities such as agents, roles and interactions, while the specification of the ACL deals with the definition of communicative actions and protocols. Besides generic roles, the RICA meta-model also includes the so-called social roles, a kind of role which represents the functionality of agents participating in social interactions. Communicative actions are conceived as special kinds of social action performed with the intention to be observed by agents playing a specific role.

Gaia (2003): Gaia [4] has been designed to explicitly model and represent the social aspects of open agent systems. The MAS meta-model is focused on the organizational structure of the system and all other concepts - agents, roles, services interactions - turn around the concept of organization. In the Gaia meta-model, an agent is an entity that plays one or more roles; a role is a specific behavior to be played by an agent, defined in terms of permission, responsibilities, and activities, and of its interactions with other roles.

ROADMAP (2003): The ROADMAP (Role Oriented Analysis and Design for Multi-Agent Programming) meta-model, defined in [2], describes runtime systems: all the constructs shown in the meta-model have concrete physical runtime manifestations. Interaction is strictly related with roles. The role hierarchy is represented as a tree where leaf nodes are atomic roles, whereas all other nodes in the tree are composite roles, that represent a localized organization of agents. The attributes of a composite role model social aspects of that organization, such as the organizational structure, social goals, social tasks and social laws; therefore a role can aggregate other roles, interact with other roles and modify other roles given the proper authorization.

PASSI (2002): System meta-models traditionally refer to two different domains: the problem domain (where the requirements are captured) and the solution domain (where the implemented system will be deployed). In conceiving the PASSI MAS meta-model [6], the authors introduce also the "agency domain", that represents the transition from problem-related concepts to the corresponding agent solution. The PASSI meta-model tries to adhere to the FIPA standards, by including "FIPA Platform" and "FIPA tasks" classes. Interaction is captured, like in other meta-models, by the presence of roles that either initiate or participate to a communication. A communication adheres to an AIP (Agent Interaction Protocol), and consists of a sequence of exchanged messages.

ADELFE (2002): ADELFE [3] is a methodology devoted to software engineering of adaptive multi-agent systems, namely systems able to change their behavior while running, in order to adjust it to the dynamic environment, either for achieving the task they are designed for, or for improving their function or performance. The MAS meta-model adopted by ADELFE is based on the notion of cooperative agent. The agent life cycle is the classical one, consisting of having perceptions, taking decisions and then doing actions. Interaction is mainly captured by the communication class, that refers to AIPs, and by cooperative rules.

2 An Ontology for the Meta-model of MAS Interaction and Communication

For the development of our "MAS Meta-model Ontology", we have followed

The "Ontology Development 101" guide, developed by Noy and McGuinness in 2001 [7]. That guide proposes an iterative approach to ontology development: it starts with a rough first pass at the ontology, then it revises and refines the evolving ontology and fills it in the details.

The methodology is divided into seven steps. In the following, we discuss how we have faced each of them.

1. Determine the domain and scope of the ontology by answering a set of basic questions and by identifying the ontology competency questions.

The methodology suggests starting the development by answering several basic questions:

1. What is the domain that the ontology will cover?
   Answer: The ontology is about MAS meta-model.

2. For what we are going to use the ontology?
   Answer: The ontology will be used to help the MAS designer in finding the right fragment of existing AOSE methodologies to do the right thing.

3. Who will use and maintain the ontology?
   Answer: The users of the ontology will be the MAS designers, while the "maintainers" will be the ontology developers (us).

4. For what types of questions the information in the ontology should provide answers?
   Answer: The ontology has to provide answers to an exhaustive list of Competency Questions. Some of the competency questions we identified, together with the answers that the ontology must be able to provide, are shown in the following, where they are expressed in natural language for the only convenience of the reader. Actually, our ontology does not include any natural language processing tool, and questions are input by means of the forms that Protégé offers. In particular, questions may be conjuctions or disjunctions of formulae "In class C there is a slot S that contains (or begins with, does not contain, etc.) a given object O (whose type depends on the type of the slot)". A question of this type is shown in Figure 7, where the "Match all" radio button means that the following

formulae are evaluated as conjunctions, and the formulae are “In class *Meta Description* there is a slot *original_name* that contains the string Resource” and “In class *Meta Description* there is a slot *appears_in_Methodologies* that contains the methodology PASSI”. Answers are provided by means of forms too. Usually, answers contain meta-descriptions of concepts, that are strings. Thus, the answer may actually contain sentences in natural language, but there sentences are simply fixed descriptions attached to the concepts by the ontology developers.

- “What is a message according to the PASSI methodology?”
  A: *A message is a part of comunication and it is expressed in an encoding language (e.g. ACL) that is totally transparent to agent. The message content could be expressed in several different content langauges (e.g. SL, KIF, RDF, etc.)* [1]
- “Which AOSE methodologies take Agent Interaction Protocols into account?”
  A: *The methodologies that take Agent Interaction Protocols into account are PASSI and ADELFE.*
- “What is a resource according to the PASSI methodology?”
  A: *A Resource is an entity that can be (also using sensors), shared, or produced by agents*
- “What is a communicative act, and which methodologies use it?”
  A: *A communicative act is the act of communication by a specific communicator over a specific time interval and originating from a specific location. Possible communicators include: people, organizations, and computational agents. The methodology that uses communicative acts is PASSI.*

2. Consider re-using existing ontologies.

Currently, existing MAS meta-models are almost all represented using UML Class Diagrams. The recent literature shows that there are strict relationships between UML Class Diagrams and Ontologies (see for example [13]). Thus, by basing our ontology on existing MAS meta-models, although represented in UML and not in a standard ontology language, we have actually re-used and integrated existing ontologies.

3. Enumerate the terms that the developer wants either to make statements about or to explain to a user.

A part of these terms derives directly from the MAS meta-models, while the other part derives from the need of describing and understanding the MAS meta-model. In particular the last part contains terms as: Meta Description, Bibliographical reference, Authors of the meta-model, Methodology, Description, etc.

The purpose of this part of ontology is to help the MAS designer in untangling the choice of which method fragment and which AOSE methodology are more suitable for modeling a given application component, and in understanding if and when similar fragments have similar meaning in different methodologies. This is achieved by associating a string to each fragment, that explains the meaning of that fragment in each of the methodologies where it appears, according to the literature where both the methodology and its meta-model are described[4]. We think that this would greatly help in comparing, and thus understanding and properly choosing, different fragments conceived by different authors, that have different

[4]Besides considering the literature, we have asked to some experts in meta-models, including L. Sterling and M. Cossentino, to check if they agree with our ontology, in order to make it as precise and useful as possible. We will integrate their feedback, as well as the feedback of other researchers that we plan to contact in the future, in the next ontology release.

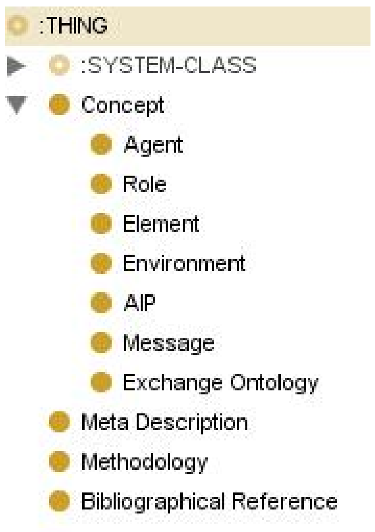


Figure 2: Class hierarchy

names but the same intended meaning. For example, the *Element* concept in ADELFE is equivalent to the *Resources* concept in GAIA, althought they have diffent names. Our ontology captures this situation by maintaining the information that the meta-concept *Element* (this name have been given by us), has original name *Element* in ADELFE and *Resources* in GAIA , but the meaning of *Element* and *Resources* is the same. Capturing this "semantic ambiguity of fragment names" in the early stages of the MAS design may avoid errors in the next stages of development.

The enrichment of our ontology with these explanations of the fragments' meanings is the part that mainly makes our work original, since it represents meta-information on the meta-model thus going toward the representation of a "meta-meta-model".

4. Define the classes and the class hierarchy by following one of the several existing approaches (top-down, bottom-up, middle-out).

We choose to use the top-down approach; the class hierarchy we have obtained is very simple and it is shown in Figure 2.

5. Define the properties of classes by means of slots, namely, the class attributes.

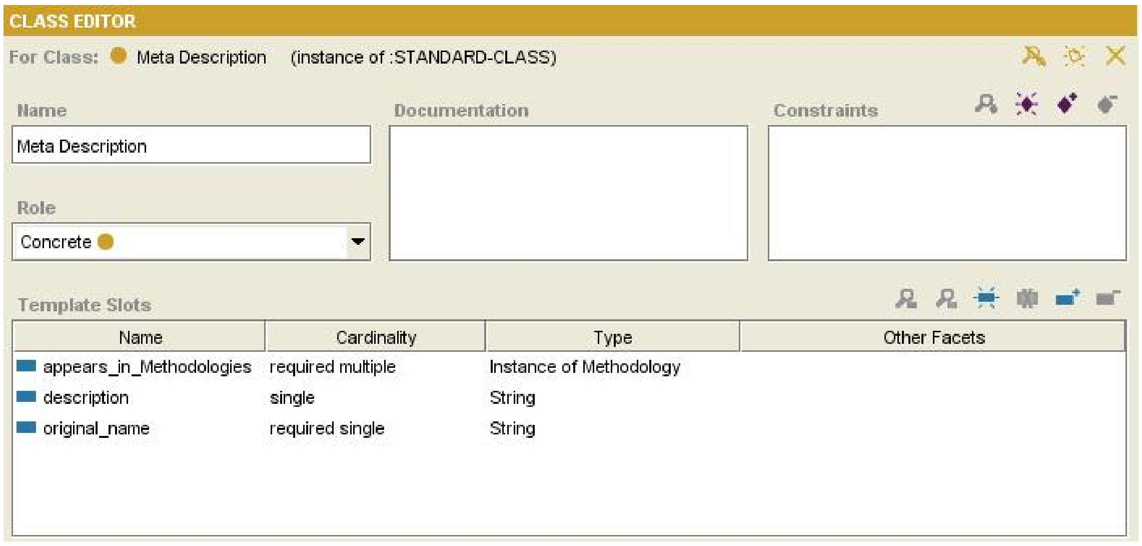
For defining the properties of the part of ontology related to the meta-models we have retrieved the information from the available literature about the meta-model cited in Section 1.

For the part of ontology that aims at describing and undestanding the MAS meta-models we report an example (see Figure 3) that depicts the properties of the class *Meta Description* (See `ftp://ftp.disi.unige.it/person/CordiV/Ontologia/OntoMetaModel.owl` for the details of the complete ontology).

6. Define the features ("facets") of the slots, such as the value type, the allowed values, the cardinality, etc.

Once completed the class hierarchy, we must define the facets of the slots. For example in Figure 3 we can see that the slot *appears_in_Methodologies* has cardinality *required multiple* and type *instance of Methodology*. This means that a Meta Description can appear in one or more methodologies. and represents a very simple form of constraint on the ontology structure.

7. Create instances of classes in the hierarchy. We have created all the instances of the ontology, using the information retrieved from the papers describing the methodologies and


CLASS EDITOR

For Class: ● Meta Description (instance of :STANDARD-CLASS)

Name: Meta Description

Role: Concrete ●

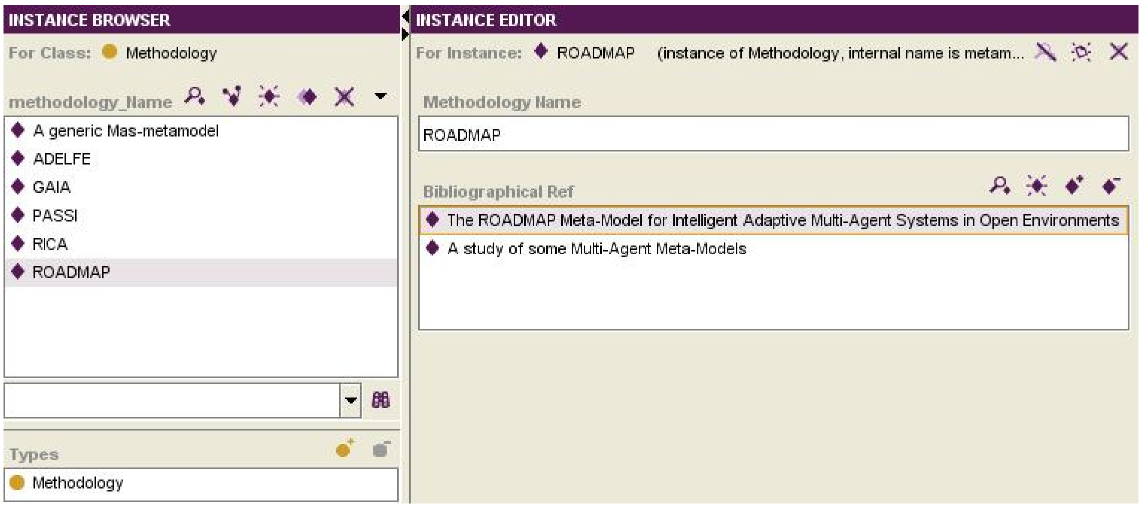
Documentation

Constraints

Template Slots

Name	Cardinality	Type	Other Facets
appears_in_Methodologies	required multiple	Instance of Methodology	
description	single	String	
original_name	required single	String	

Figure 3: Properties of class Meta Description


INSTANCE BROWSER

For Class: ● Methodology

methodology_Name

- ◆ A generic Mas-metamodel
- ◆ ADELFE
- ◆ GAIA
- ◆ PASSI
- ◆ RICA
- ◆ ROADMAP

Types

● Methodology

INSTANCE EDITOR

For Instance: ◆ ROADMAP (instance of Methodology, internal name is metam...

Methodology Name: ROADMAP

Bibliographical Ref

- ◆ The ROADMAP Meta-Model for Intelligent Adaptive Multi-Agent Systems in Open Environments
- ◆ A study of some Multi-Agent Meta-Models

Figure 4: Instances of Methodologies

meta-models prevoiusly cited in Section 1.

In Figure 4 we can see all the instances of these meta-models. The meta-models taken into account are the six represented in Figure 1. From the ontology, we can for example discover that ROADMAP has two bibliographical references from which we have retrieved all the information concerning the specified meta-model (left side of the windows).

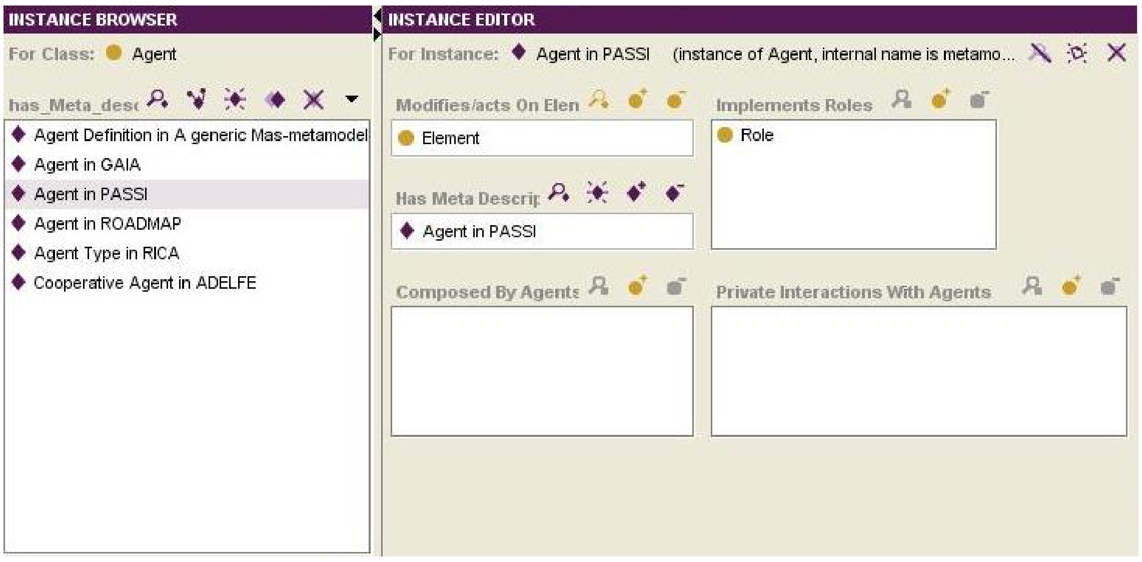
From Figure 5 we can understand that *Agent in PASSI modifies/acts on Element* (which is a component of the Environment) and *implements Role*. Moreover we can see that it has a *Meta Description* to explain the meaning and the use of Agents in PASSI.

A *Meta Description* (see Figure 3) has three properties (slots): *Description*, *Original Name* and *Appears in methodology*.

In Figure 6 an example of instance for the class *Meta Description* is reported. It summarises the following information: "An Agent in GAIA meta-model is an entity that plays one or more roles and it is identified by the keyword *Agent*".

Final Step: Querying the ontology. Now that the ontology development is terminated we can query the ontology and test if it answers correctly to the Competency Questions.

For example, we can try with the following Competency Questions: "What is a resource


INSTANCE BROWSER

For Class: Agent

has_Meta_desc

- Agent Definition in A generic Mas-metamodel
- Agent in GAIA
- Agent in PASSI
- Agent in ROADMAP
- Agent Type in RICA
- Cooperative Agent in ADELFE

INSTANCE EDITOR

For Instance: Agent in PASSI (instance of Agent, internal name is metamo...

Modifies/acts On Elen: Element

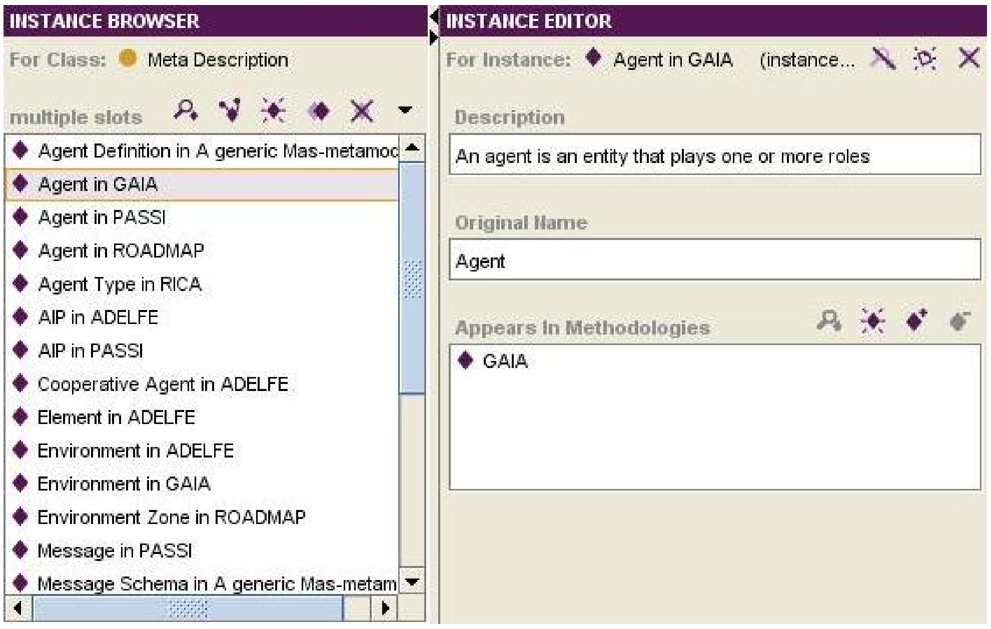
Implements Roles: Role

Has Meta Descrip: Agent in PASSI

Composed By Agents:

Private Interactions With Agents:

Figure 5: Instances of Agent


INSTANCE BROWSER

For Class: Meta Description

multiple slots

- Agent Definition in A generic Mas-metamod
- Agent in GAIA
- Agent in PASSI
- Agent in ROADMAP
- Agent Type in RICA
- AIP in ADELFE
- AIP in PASSI
- Cooperative Agent in ADELFE
- Element in ADELFE
- Environment in ADELFE
- Environment in GAIA
- Environment Zone in ROADMAP
- Message in PASSI
- Message Schema in A generic Mas-metam

INSTANCE EDITOR

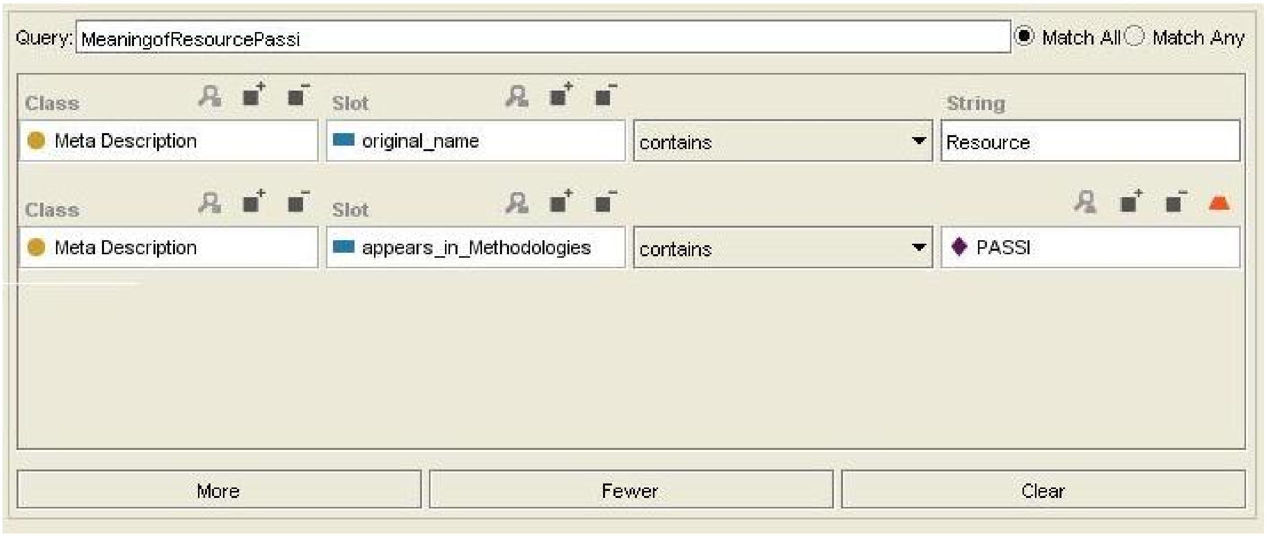
For Instance: Agent in GAIA (instance...

Description: An agent is an entity that plays one or more roles

Original Name: Agent

Appears In Methodologies: GAIA

Figure 6: Instances of Meta Description


Query: MeaningofResourcePassi [✓ Match All] Match Any

Class	Slot		String
Meta Description	original_name	contains	Resource
Meta Description	appears_in_Methodologies	contains	PASSI

Figure 7: Query for the Competency Questions

according to the PASSI methodology?", formulated as explained in Step 1.4. The Figure 7 shows the query in Protégé.

3 Conclusions and Future Work

In this paper we have described an ontology that models a MAS "meta-meta-model", restricted to interaction and communication aspects. We say that our ontology copes with a "second order meta-level", because it tries to unify in a coherent and consistent way not only the elements already included in other existing meta-models, but also information about the meta-models and the methodologies for which they have been defined. The purpose of our ontology is to help the MAS designer in untangling the choice of which method fragment and which AOSE methodology are more suitable for modeling a given application component, and in understanding if and when similar fragments have a similar meaning in different methodologies (which is a thorny question!). Our ontology can be downloaded from `ftp://ftp.disi.unige.it/person/CordiV/Ontologia/OntoMetaModel.owl` as an OWL file. We are working at equipping it with a user-friendly interface for better navigating and querying it, that we plan to develop using Jena[5].

As far as related work is concerned, after the analysis of the different MAS meta-models of ADELFE, Gaia and PASSI, described in Section 1, the authors of [1] developed a quite a huge model including all the relevant features of the three original meta-models. The resulting "unified meta-model" allows the designer to create societies without (like in ADELFE) or with predefined organizations. For this purpose, the generic agent has been enriched with all the properties an agent may have, be it cooperative or not. The generic agent has Gaia-like roles complemented by some PASSI features (tasks and a FIPA-compliant[6] communication structure), and may either belong to an organization, or follow cooperation rules.

Although taking three methodolgies (and related meta-models) into account, the "unified meta-model" provides the means neither for understanding from which meta-model each fragment derives, nor for querying the model. Our ontology adds three more meta-models to those considered by the "unified meta-model", namely the "generic MAS meta-model", RICA, and ROADMAP, and associates meta-data to them that can be analyzed and queryed,

[5]`http://jena.sourceforge.net`
[6]`http://www.fipa.org/`

as discussed in Section 2.

The work closest to ours is the Ingenias meta-model[7]. It consists of five MAS-related models, Tasks and Goals, Interaction, Agent, Environment, and Organization, plus the models for AUML interaction and activity diagrams, and for UML use case diagrams. All the models are available on the Web, and can be easily browsed through a graphical user interface developed using SVG[8]. Clear textual definitions are provided for all the components of the model, also explaining when they should be used and in which models they appear.

In Ingenias, the models that are mainly related with communication and interaction are those describing Interaction and AUML interaction diagrams [11]. Different types of interaction, including message passing and remote procedure calls are considered. The interaction unit is the speech act, defined as a string, and the way interaction is carried on may be described using AUML.

The interaction model of Ingenias is much more detailed than the ones of the six meta-models considered in Section 1, and for this reason we plan to integrate it in our ontology in the very near future. With respect to Ingenias, that refers to a unique AOSE methodology (the Ingenias methodology, [12]), our meta-model, although currently simpler, copes with more methodologies, providing thus a "meta-meta-model".

References

[1] C. Bernon, M. Cossentino, M. P. Gleizes, P. Turci, and F. Zambonelli. A Study of Some Multi-agent Meta-models. *In Proc. of AOSE'04*, pages 62–77. July 2004.

[2] T. Juan, and L. Sterling. A meta-model for intelligent adaptive multi-agent systems in open environments. *In Proceedings. of AAMAS'03*, pages 1024–1025. ACM Press, 2003.

[3] C. Bernon, M. P. Gleizes, S. Peyruqueou, and G. Picard. ADELFE, a Methodology for Adaptive Multi-Agent Systems Engineering. *In Proceedings. of ESAW'02*, 2002.

[4] F. Zambonelli, N. R. Jennings, and M. Wooldridge. Developing multiagent systems: the Gaia Methodology. *ACM Trans on Software Engineering and Methodology*, vol. 12, no. 3, pages 317–370, 2003 .

[5] M. Wooldridge, and P. Ciancarini. Agent-Oriented Software Engineering: The State of the Art. In P. Ciancarini and M. Wooldridge, editors, A*gent-Oriented Software Engineering. Springer Lecture Notes in AI Volume 1957*, January 2001.

[6] M. Cossentino. Different Perspectives in Designing Multi-Agent System. *In Proceedings of AgeS'02*, 2002.

[7] N. F. Noy, and D. L. McGuinness. Ontology Development 101: A Guide to Creating your First Ontology. Tech. rep., KSL-01-05, Stanford Knowledge Systems Laboratory, 2001.

[8] G. Beydoun, C. Gonzalez-Perez, G. Low, and B. Henderson-Sellers. Synthesis of a Generic MAS Meta-model. *In Proceedings of SELMAS'05*, 2005.

[9] J. M. Serrano, and S. Ossowski. On the Impact of Agent Communication Languages on the Implementation of Agent Systems. *In Proceedings of CIA'04*, Springer pages 96–102, 2004.

[10] Object Management Group. Unified Modeling Language: Superstructure version 2.0, 2005.

[11] M-P. Huget et al.. FIPA Modeling: Interaction Diagrams, First proposal, 2003-07-02. `http://www.auml.org/auml/documents/ID-03-07-02.pdf`, 2003.

[12] J. Pavón, and J. J. Gómez-Sanz. Agent Oriented Software Engineering with INGENIAS. *In Proceedings of CEEMAS'03*, Springer LNCS, vol. 2691, pages 394–403, 2003.

[13] S. Cranefield, and M. Purvis. UML as an ontology modelling language. *In Proceedings of IJCAI'99*, 1999.

[7]`http://grasia.fdi.ucm.es/ingenias/metamodel/`

[8]`http://www.adobe.com/svg/main.html`