Modelling and Meta-modelling Issues in Agent Oriented Software Engineering: The AgentLink AOSE TFG Approach

Massimo Cossentino[1], Carole Bernon[2], Juan Pavòn[3]

[1] *Istituto di Calcolo e Reti ad Alte Prestazioni – Italian National Research Council, cossentino@pa.icar.cnr.it*

[2] *IRIT – University Paul Sabatier, France, bernon@irit.fr*

3 Universidad Complutense Madrid, jpavon@sip.ucm.es

1. Introduction

AgentLink III is a coordination action for agent-based computing funded by the European Community; it aims at providing support for the network of European researchers and developers with a common interest in agent technology through the organization and support for a large number of events aimed at industry outreach, standardization issues, as well as providing support for academic dissemination and collaborations.

Among the events organized by AgentLink a relevant role is played by Technical Fora. Currently two of them have been held, the first in Rome (July 2004) and the second in Ljubljana (March 2005). Each Technical Forum includes the activity of several different groups (Technical Forum Groups, TFGs hence afterwards) and in this paper we report the work done within the Agent-Oriented Software Engineering (AOSE) TFG meeting in Ljubljana.

As there are many issues and perspectives in the scope of AOSE, the AOSE TFG purpose is the creation of a path towards integration and interoperability of methodological approaches for MAS development. This involves the definition of a common framework for MAS specification, which includes the identification of a minimum set of concepts and methods that can be agreed in the different approaches. The tool for defining this framework is meta-modelling. The principle of meta-modelling has been already used in other fields of software engineering, for instance, in the specification of UML by OMG, to describe the elements of the language, their constraints and relationships. This approach is also valid to specify the concepts that are used by agent-oriented methodologies.

With this assumption, one of the main issues in AOSE TFG has been to elaborate and discuss MAS meta-models for different methodologies. Work on the compilation of one or more MAS meta-models can be used as reference points by the whole community. Although this work presents several challenges (see section 3.3.1), all of us agree that achieving concrete results in this area, would be very useful for several reasons: (i) this partly solves the lack of standardization in this area, as it has been remarked in the AgentLink Roadmap, (ii) this could encourage the development of more flexible and versatile design tools and (iii) this is one of the essential steps for reaching a concrete maturity in the study of the whole agent design process.

During the second meeting, the proposed aims were more especially focused on refining the contributions about MAS meta-models that some supporters presented. This activity

led to the identification (and formalization) of a meta-model that members hope will meet a sufficient consensus to be adopted as a common reference point by the European research community in this area. Other aspects of agent-oriented software engineering have been faced as well with specific contributions about agent modelling languages and agent mobility design.

In order to report in details the activity of the AOSE TFG in Ljubljana, this paper is organized as follows: section 2 provides a complete overlook on the talks and discussion held during the meeting, section 3 introduces the main motivations that are behind the choice of identifying an unified MAS meta-model, section 4 describes the main sources that have been considered in this unification work, whose result is presented in section 5; finally some conclusions are drawn in section 6.

2. Modelling Languages and MAS Meta-models

Discussion sessions in the second meeting were justified by the challenges derived from the first AOSE TFG meeting in Rome and more especially addressed in the last talk given by F. Zambonelli [33] about open directions in AOSE. The lack or the possible improvement of modelling tools concerning agents or multi-agent systems, and the lack of agreement on concepts used in the AOSE area are the main factors explaining the talks given in Ljubljana.

2.1 Modelling Languages

As discussed in Rome, one of the challenges in the AOSE area was to give new tools to express and control the behaviour of agents and/or the behaviour of the MAS they are evolving within. Working in such a way would facilitate the spreading of agent-based concepts and applications in the industrial world.

The first talk came from this industrial world with the presentation of AML (Agent Modelling Language) by I. Trencansky [30]. AML is a semi-visual modelling language, versatile and easy to expand, based on the UML 2.0 specifications thus allowing to reuse well-defined concepts and to enable the support of existing CASE tools [12]. AML is designed to support business modelling, requirements specification, analysis, and design of software systems that use MAS concepts and principles. AML has a layered structure built on top of UML to provide an abstract syntax, a semantics and a notation. Expressing the typical features of multi-agent systems is done by extending UML with several new modelling concepts (such as agents, resources, environment, role, social aspects or ontologies) while ensuring that the resulting language preserves UML specificities. A non-preserving extension is also provided to add some meta-properties to UML and complete the definition of the AML meta-model. Above this meta-model and notation, two UML profiles for AML are given. They enable implementing AML in CASE tools based on UML 1.* or UML 2.0; in concrete, AML is supported by IBM Rational Rose and LS/TS from Living Systems. Using these AML profiles, a designer is free to customise AML through the definition of extensions to this language. The V.0.9 release of AML is available since December 2004, and has been submitted to OMG for the RFP on Modelling Agent-Based Systems.

In specific cases, mobile agents are useful, therefore, the provision of tools to model their deployment, migration and interactions, for example, becomes then an important issue. Mario Kusek [20], talked about how to model agent mobility with UML sequence diagrams. This work is justified because this mobility is not represented in the current UML sequence diagrams and because modelling agent creation, mobility paths and

current location of an agent has not yet been fully addressed by FIPA, UML, AUML or AML. In AUML a deployment diagram can capture the reason why an agent moves and the location where it moves, and an activity diagram may express when the agent has to move. In UML these expressions are made possible by extending the activity or sequence diagrams and/or defining a stereotype «host». Finally, AML enables a designer to model the structural and behavioural aspects of entity mobility through, as seen above, extension of UML relationships and definition of a stereotype «move». However, all these modelling languages do not give an overall view about agent roaming and execution path. Four solutions, extending UML sequence diagrams, were described to try to tackle this issue. First, with stereotyped mobility diagrams, an agent is located on a node by sending a stereotype message to it and then moves to another node by sending it a message stereotyped with another value. Second, in swimlaned mobility diagrams, a swimlane represents a node and an agent moves in the same way than in the first approach except that it terminates at the source node and is created again on the destination node. Third, in state representation mobility diagrams, the mobility is represented by changing the state of the moving agent. Finally, in frame fragment mobility diagrams, each frame fragment of a sequence diagram represents execution on a node. The advantages and drawbacks of these approaches were then compared, with respect to the number of nodes involved, the space needed for the graphics, the expression of the mobility, using a case study in which agents roam the Internet to find better prices for products.

While in this section we dealt with agent modelling related issues, in the next one we present MAS meta-models related to the different methodological approaches born in the AgentLink AOSE community.

2.2 MAS Meta-models

Several studies have been carried on recently about the idea of assembling a new design process for agents by taking methods from existing agent-oriented methodologies. These can be seen as an adaptation of the *method engineering* approach [8][24]. Other researchers and software developing companies are working on the production of agent-specific design and coding tools or the extension of existing ones. We can say that from requirements identification down to the final deployment of the executable code there are important research activities that while looking at the agent systems indeed lack of a well consolidated and sometimes even defined meta-model of the multi-agent society.

With the term meta-model we mean a model of the concepts that can be used to design and describe actual systems. The models describing a system are instances of the meta-model (i.e., entities of the system model are instances of the meta-model entities). In this way, it is possible to build several models (views) of a system; for instance, one model could represent the organization view of the system at an abstract level (as an instance of the concepts in the meta-model that describe organizational issues) while another could be more implementation oriented and show how the system is deployed in a target platform (as an instance of a platform specific meta-model).

In the object-oriented context the construction of a design process, the definition of its components (analysis, design, testing activities) and the execution of the design rely on a common denominator, the universally accepted concept of object and related meta-model of the object-oriented system.

It is not so in the agent-oriented context where the lack of such a shared meta-model brings to significant differences in the way different researchers deal with similar (at

least in the name) concepts. We are not here saying that we desire to achieve an unification of all the agent-oriented design approaches since their number is *per se* a richness, but instead we mean that a unique well defined meta-model including a set of definitions of its concepts will enable a deeper understanding of the differences of all of these approaches and their integration.

There are several direct applications of meta-models. First, as already argued, meta-models can guide the development process as they can be seen as templates of what a system model has to be. In the case of MAS, a meta-model specifies what types of entities the developer has to look for: agents, goals, interactions, tasks, resources, etc. It also establishes constraints on the relationships and use of those elements. Meta-models can also be seen as the specification of a modelling language, and therefore they are fundamental in developing tools that can support the development process, as it has been proposed in MESSAGE and implemented by INGENIAS [17].

In the scope of the AOSE TFG, similarly to the work that is going on within the FIPA Methodology TC[1], meta-models are also used to get a better understanding of the agent concepts as applied in different methodologies and to look for some agreement in the main elements that can be used to specify a MAS.

During the Rome meeting several MAS meta-models were presented and a first proposal of unification, based on three AO methodologies – ADELFE, Gaia and PASSI, was discussed. Nevertheless, more work had to be done during the second meeting in order to continue the effort of unification and to make a further step towards a reference point for the whole European agent community.

Several MAS meta-models were presented in Ljubljana to enrich the previous discussion, they included the MAS meta-models of ADELFE presented by C. Bernon [4], INGENIAS presented by Jorge J. Gómez-Sanz [15], PASSI presented by L. Sabatucci [13], RICA presented by J. Manuel Serrano [29] and Tropos presented by D. Bertolini [5]. Because working on a unifying MAS meta-model was one of the main aims of the AOSE Technical Forum Group (TFG), all of these talks will be more detailed in a dedicated section (the next one).

Finally, Zahia Guessoum [18], talked about an MDA-based approach for MAS meta-models and how to fill the gap between existing MAS tools (such as the multi-agent platforms DIMA, Jade, MadKit or Zeus) and agent-oriented methodologies or meta-models using the OMG's MDA (Model Driven Architecture) approach. This approach consists in separating the application logic (described in a PIM – Platform Independent Model) from the underlying technology (described in a PSM – Platform Specific Model). She gave an overview of Meta-DIMA, a MDA-based MAS development process, that could be: define the PIMs and PSMs by analysing the multi-agent applications, define a library of meta-models by identifying the concepts used and design the transformation rules to implement a meta-model from its description. A first step has been done trying to define a PSM for the multi-agent tool DIMA and PIMs from PASSI and Aalaadin/PASSI meta-models. Some rules were given to enable transformations from these PIMs towards the DIMA-based PSM.

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

3. Review of MAS Meta-models

In this section we describe the details of the MAS meta-models discussed and presented during the Ljubljana meeting. They became the basis for the construction of the proposed unified meta-model that is presented in the next section and is the major outcome of this event.

3.1 The ADELFE MAS Meta-model

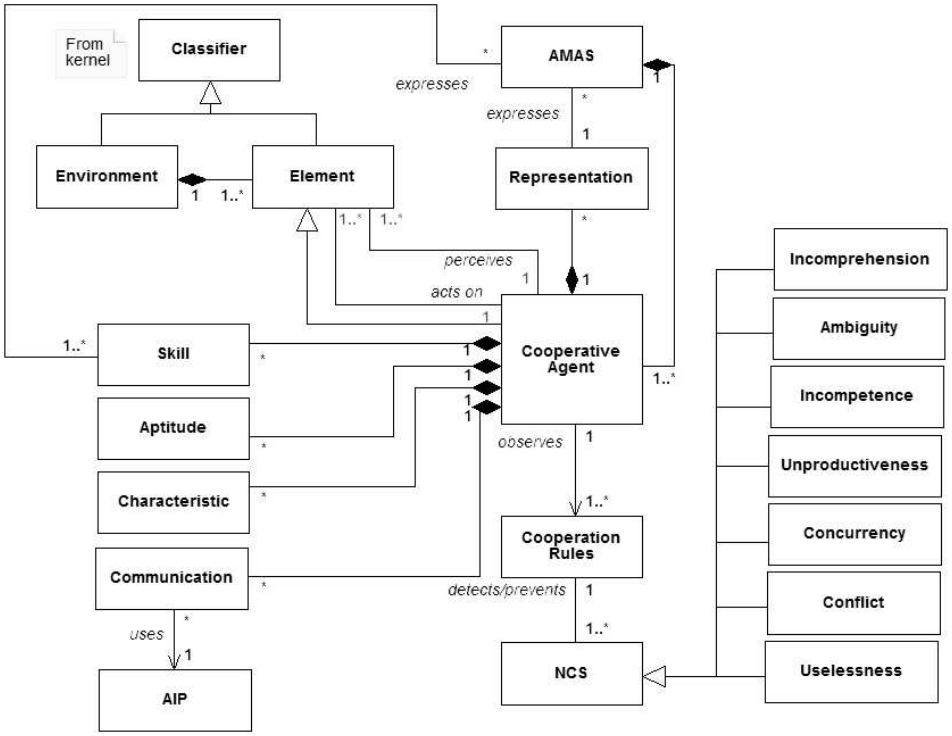

Figure 1. The MAS Meta-model adopted by ADELFE.

ADELFE [1][2] is a methodology devoted to the design of adaptive multi-agent systems (AMAS). According to this approach, building a system which realises the right desired global function (which is functionally adequate) is achieved by designing agents with a cooperation-driven social attitude.

An agent belonging to an AMAS ignores the global function of the system, only pursues a local goal and tries to always keep cooperative relations with other agents. Such an agent is called a "Cooperative Agent" because its social attitude is based on cooperation but its lifecycle is still a classical one. It consists in having perceptions, taking decisions and then doing actions (perceive-decide-act). Local cooperation rules enable the agent to detect and solve Non Cooperative Situations (NCS) that play a fundamental part in the ADELFE design. These NCS are cooperation failures (e.g., cooperation protocol not obeyed, or unpredictable situations) and they are inconsistent with the cooperative social attitude of a cooperative agent.

The MAS meta-model adopted for ADELFE [3] is therefore explained by the features such cooperative agents possess. It tries to constrain the agent behaviour with a cooperative attitude by using local cooperation rules that an agent observes and which enable it to detect and solve NCS of different kinds (such as incomprehension and uselessness). The concept of environment is essential for MAS, agents are evolving in an environment which can be a physical one, made up of elements of different kinds, or

a social one, made up of other agents. An element composing an environment is a specialisation of the UML Classifier, more especially an agent can be also viewed as such a specialisation. A cooperative agent has interactions with its environment, it can receive information through perceptions and act on the environment during its action phase. Interactions may use communications which can be done in a direct manner (by exchanging messages, using AIPs to express the communication pattern, for instance) or an indirect one (environment-mediated). An agent has a representation of the world in terms of beliefs about other agents, the configuration of the physical environment around it and the agent itself. The agent uses these representations to determine its behaviour. A representation can be shared by different agents. If an agent has representations that may evolve, they can be expressed using an adaptive multi-agent system.

On the left part of Figure 1, we find Skill, Aptitude and Characteristic; skills represent the specific knowledge that enable each agent to realise its own partial function in the world. If these skills have to evolve, they may also be implemented as an AMAS. Characteristics are intrinsic or physical properties of the agent while aptitudes relate the agent's capability of reasoning both about knowledge and beliefs it owns.

3.2 Gaia

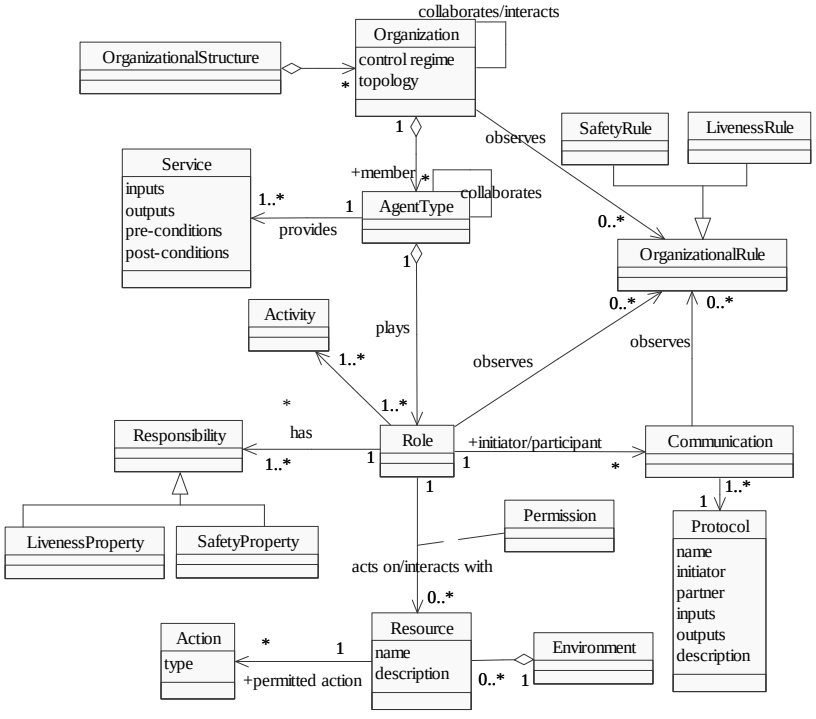

Figure 2. The MAS Meta-model adopted by Gaia.

The Gaia methodology evolved from an initial version [31], mainly focused on the design of handle small-scale, closed agent-based systems to the new one [32] based on the key consideration that an organization is more than simply a collection of roles and agents. Therefore the main difference is that it has been designed in order to explicitly model and represent the social aspects of open agent systems, with particular attention to the social goals, social tasks or organizational rules.

Having a deeper look at the MAS meta-model for the second, extended version of Gaia (Figure 2) [3] we notice that the basic building blocks of the former version of Gaia – namely agents, roles, activities, services, and protocols – are still present but now they are located in the context of a specific environment and of a specific organization.

The Gaia agent is an entity that can play one or more roles; a role is a specific behaviour to be played by an agent, defined in terms of permission, responsibilities, and activities, and of its interactions with other roles. In playing a role, an agent actualizes its behaviour in terms of services that can be activated accordingly to some specific pre- and post-conditions.

The environment abstraction is a key element in Gaia MAS meta-model; it explicitly specifies all the entities and resources a multi-agent system may interact with, restricting the interactions by means of the permitted actions.

As already said, the explicit representation of the agent organization is the main improvement in the Gaia extension presented in [32], this is mainly achieved by introducing the organizational rules and the organization structure.

Organizational rules, specify some constraints that the organisation has to observe; these constraints may be global, affecting the behaviour of the society as a whole, or concerning only specific roles or protocols while organisation structure establishes the overall architecture of the system, that is the position of each role in the organisation and its relationship with other roles. Organizational rules and organizational structures are strictly related, in that organizational rules may help designers in the identification of the organizational structures that more naturally suit these rules.

3.3 INGENIAS

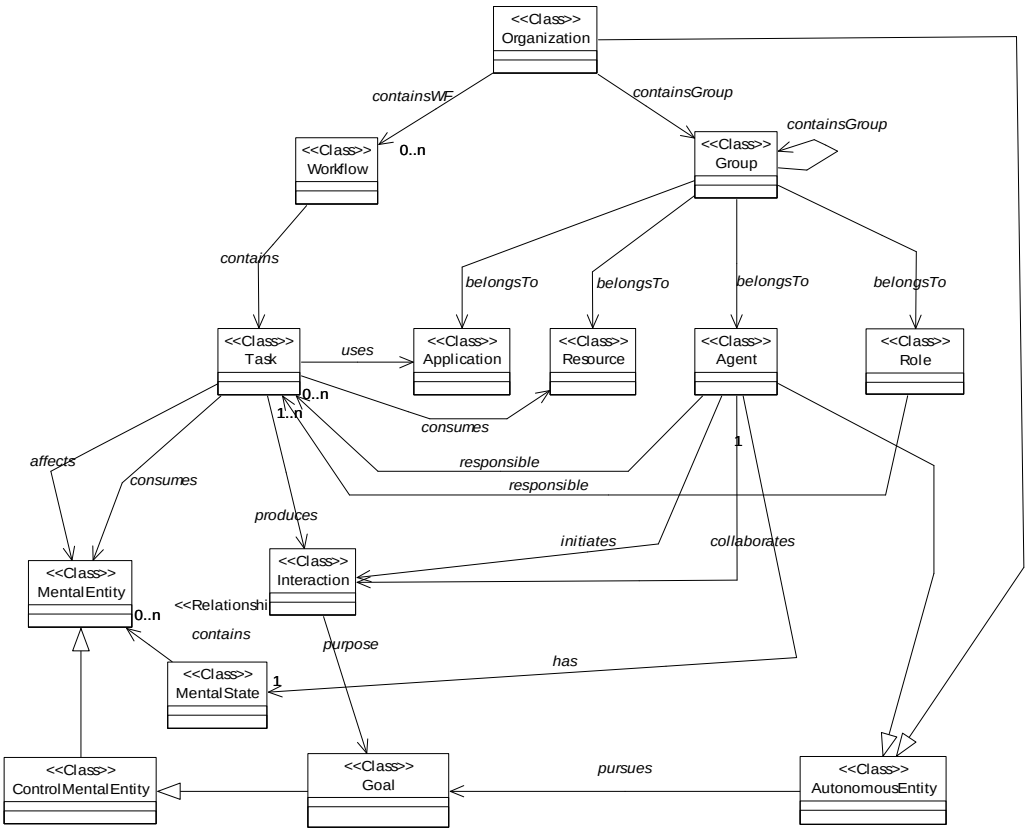

Figure 3. Summary of INGENIAS MAS meta-model

INGENIAS [23] is both a methodology and a set of tools for development of multi-agent systems (MAS). As a methodology, it tries to integrate results from other proposals and considers the MAS from five complementary viewpoints: organization, agent, tasks/goals, interactions, and environment. It is supported by a set of tools for modelling (graphical editor), documentation and code generation (for different target platforms). INGENIAS adopts the definition of meta-models for MAS from MESSAGE [10], but it refines and extends them, and builds support tools for all stages of the development cycle, from requirements elicitation to implementation and testing.

INGENIAS considers the specification of a MAS from five viewpoints:

1. Organization viewpoint. Describes how system components (agents, roles, resources, and applications) are grouped together, which tasks are executed in common, which goals they share, and what constraints exist in the interaction among agents. These constraints are expressed in form of subordination and client-server relationships.

   An Organization is an Autonomous Entity, which pursues a Goal, and can be structured in Groups (structural entities), and contains Workflows (dynamics of the organization processes). A Group may consist of Roles, Agents, Resources, Applications. Workflows define precedence relationships among Tasks, the Resources assigned to Tasks and their responsible (in terms of Roles or Agents).

2. Agent viewpoint. Describes single agents, their tasks, goals, initial mental state, and played roles. Moreover, agent models are used to describe intermediate

states of agents. These states are presented using goals, facts, tasks, or any other system entity that helps in its state description. This way, an agent model could represent in what state should be an agent that starts an interaction.

An Agent is also an Autonomous Entity, which plays some Roles and pursues Goals. It has a Mental State, which consists of Mental Entities, such as Goals, Facts, Beliefs. There is a Mental State Manager that provides the mechanisms for creating, deleting, modifying mental state entities, and a Mental State Processor that determines how the Mental State evolves and what actions an agent should try.

3. Interaction viewpoint. Describes how interaction among agents takes place. Each interaction declaration includes involved actors, goals pursued by the interaction, and a description of the protocol that follows the interaction.

   In INGENIAS, an Interaction is initiated by an Agent, with some Goal (intention). Several Agents can participate in an Interaction. Several formalisms can be used to describe an interaction, such as UML collaboration diagrams, AUML and GRASIA diagrams.

4. Tasks and Goals viewpoint. Describes relationships among goals and tasks, goal structures, and task structures. It is used also to express which are the inputs and outputs of the tasks and what are their effects on the environment or an agent's mental state.

5. Environment viewpoint. Defines agent's perception in terms of existing elements of the system. It also identifies system resources and who is responsible of their management.

INGENIAS viewpoints can be complemented with extensions of known notations, such as UML use case diagrams or UML collaboration diagrams. These extensions consist of relating INGENIAS elements with UML entities, for instance use cases with interactions.

Developers should be aware that there are elements that may appear in different viewpoints. This repetition of entities across different viewpoints induces dependencies among them. For instance, the same task entity can appear in an agent view, a task/goal view, an organization view, and an interaction view. Therefore, to completely define a task, creating different diagrams for different views is required. If the developer fails to create all of these diagrams, the system specification may be incomplete. On the other hand, if the developer creates all required diagrams, but in an organization view a task is assigned to a role and in an agent view it is assigned to another, different role, it could be interpreted that the specification is not consistent. These constraints are checked by the INGENIAS Development Kit (IDK, available at http://ingenias.sourceforge.net), whose base is the INGENIAS meta-model specification.

3.4 PASSI

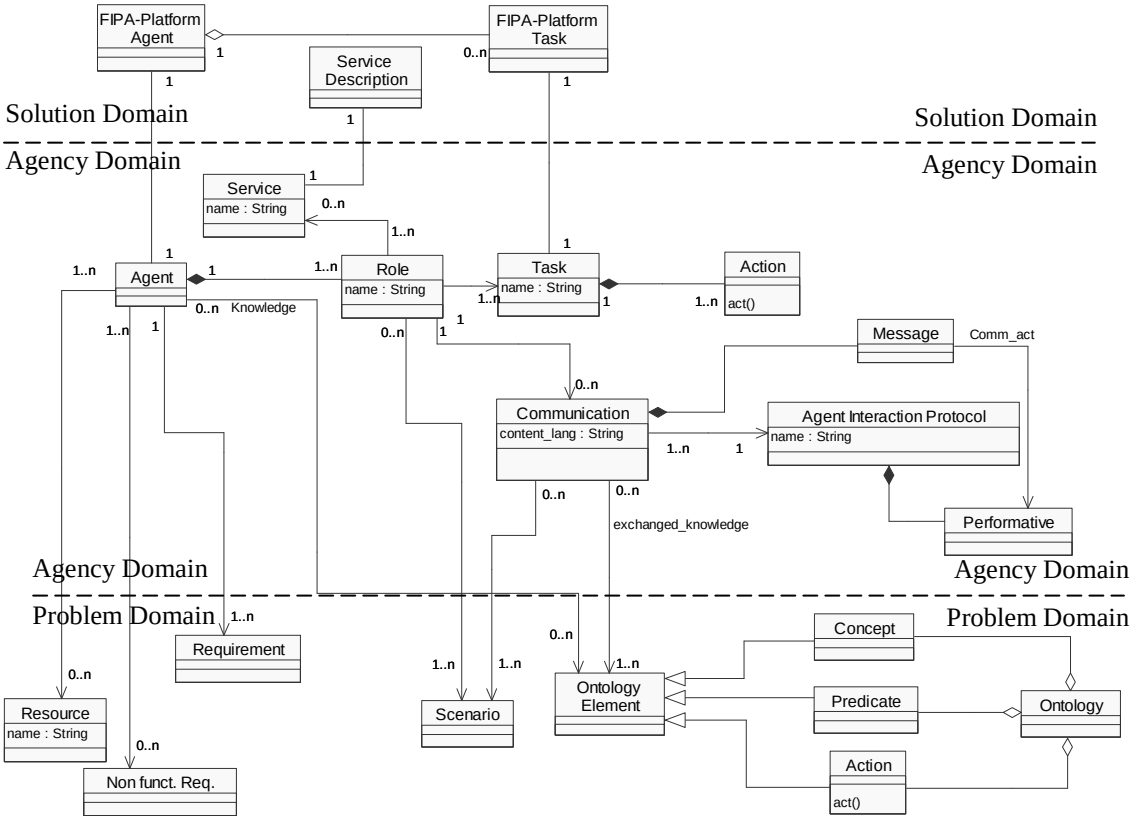

Figure 4. The MAS meta-model adopted by PASSI

PASSI (Process for Agent Societies Specification and Implementation) [14][9] is an iterative-incremental process for designing multi-agent systems starting from functional requirements that adopts largely diffused standards like UML (as the modelling language, although extended to fit the needs of agents design) and FIPA (as the agent platform). PASSI covers all the phases from requirements analysis to coding and testing with a specific attention for the automation of as many activities as possible with the support of PTK (PASSI ToolKit) a specifically conceived design tool.

The PASSI MAS meta-model [3] is organized in three different domains: the Problem Domain (where requirements are captured), the Agency Domain that represents the transition from problem-related concepts to the corresponding agent solution (that is a logical abstraction) and the Solution Domain (where the implemented system will be deployed).

The Problem Domain (Figure 4) deals with the user's problem in terms of scenarios, requirements, ontology and resources; scenarios describe a sequence of interactions among actors and the system. Requirements are represented with conventional use case diagrams.

The system operating environment is depicted in terms of concepts (categories of the domain), actions (performed in the domain and effecting the status of concepts) and predicates (asserting something about a portion of the domain elements), the environment also includes resources that can be accessed by agents.

The Agency Domain includes the agent that is the real centre of this part of the model; each PASSI agent is responsible for accomplishing some functionalities descending from the requirements of the Problem Domain. Each agent during its life can plays some roles; these are portions of the agent social behaviour characterized by some specificity such as a goal, or providing a functionality/service and in so doing it can also access some resources. The Service component represents the service provided by a role in terms of a set of functionalities (including pre- and post-conditions as well as many other details mostly coming from the OWL-S specifications), and can be required by other agents to reach their goals. Agents could use portions of behaviour (called tasks) or communications to actuate the role's aims.

In PASSI, the term task is used with the significance of atomic part of the overall agent behaviour and, therefore, an agent can accomplishing its duties by differently composing the set of its own tasks.

A PASSI communication is composed of one or more messages expressed in message content language and following an agent interaction protocol (AIP) composed of performatives (the predefined semantic of the message content [26]). This structure is the consequence of the choice of adopting FIPA specifications for the systems to be designed with PASSI.

The Implementation Domain describes the structure of the code solution in the chosen FIPA-compliant implementation platforms and it includes three elements: (i) the FIPA-Platform Agent (the implementation class for the agent entity represented in the Agency Domain); (ii) the FIPA-Platform Task (the implementation structure available for the agent's Task); (iii) the ServiceDescription component that is the implementation-level description (for instance an OWL-S file) of each service already specified in the Agent Domain. This description is also useful to enable the system openness and the reusability of its components (agents).

3.5 RICA

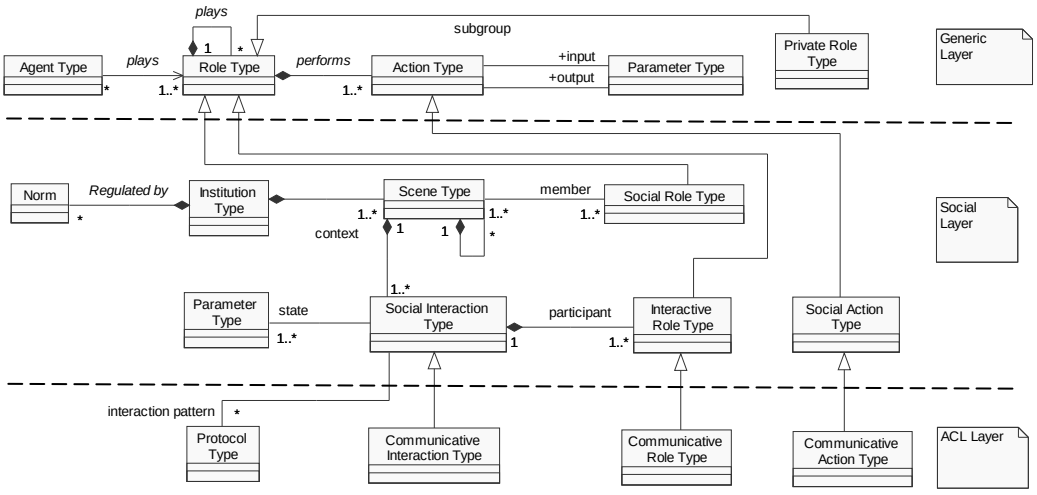

Figure 5. The MAS meta-model adopted by RICA

The RICA (Role/Interaction/Communicative Action) approach [27] integrates relevant aspects of Agent Communication Languages (ACL) and Organisational Models and it is itself based on the concepts of Communicative Roles and Interactions. RICA describes

a conceptual framework that guides the designer from the specifications of the organisational model of the agent society to the definition of the Agent Communication Languages of the multi-agent system.

The organisational model is specified in terms of entities such as agents, roles and interactions, while the specification of the ACL is based on the definition of communicative actions and protocols.

RICA is essentially made of two major components: a meta-model (defining the language used to specify the organisational and communicative entities), and a specification of the structure and behaviour of these entities. The RICA MAS meta-model [19] [27] is organized in three different layers: the first one deals with the generic elements of the system (agent, role and action types); the second one includes social concepts like norms, and institutions; the last one is devoted to agents' interactions via communications.

More in details, in Figure 5, we can see that the RICA agent can play several roles; some roles are called 'private' thus meaning that they do not require interactions with other agents but only with the environment. In playing its roles the agent performs some actions characterized by inputs and outputs parameters.

In the social layer, generic role types are specialized in social and interactive role types. Social role types participate in scenes (represented by Scene Type in the model) that can be regarded as meeting points used to study interactions. Scene Types are used to build Institution Types that are regulated by Norms. Interactive Role Types regards the agent's social interactions (represented by the Social Interaction Type element) where each agent acts as a participant.

In the third layer, we can find the specialization of some elements of the previous layer according to a communicative direction; this produces such elements as: Communicative Interaction Type, Communicative Role Type, Communicative Action Type and Protocol Type.

3.6 Tropos

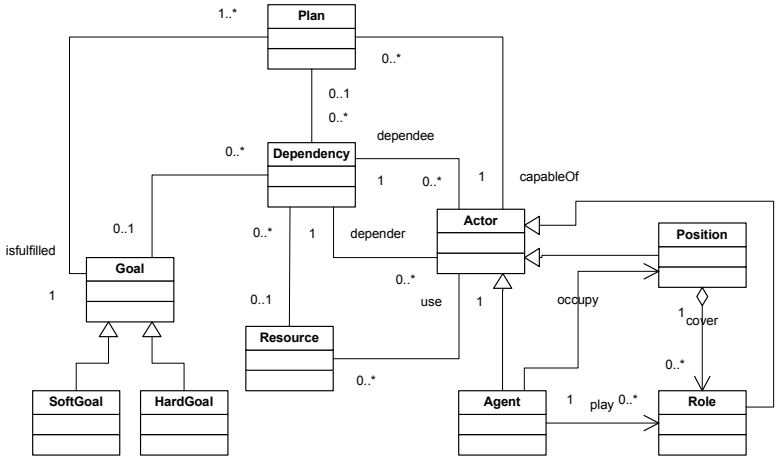

Figure 6. The MAS meta-model adopted by Tropos

Tropos [11][7] is an incremental design methodologies that aims at modelling both the multi-agent system and its environment . The process covers the early phases of the analysis where it is characterised by the adoption of a goal-based approach.

From the beginning (Early and Late Requirements analysis), the designer deals with domain stakeholders (modelled as social actors) and the goals they want to reach, resources they can share and plans they may perform. Actors are related by mutual and intentional dependencies that are consequences of their goals. In the next phase (Architectural Design), actors are mapped to a set of software agents while in the final Detailed Design, agents capabilities and interactions are defined in details.

Differently from the other MAS meta-models, the Tropos one introduces the concept of actor seen as a generalisation of the agent [6] (Figure 6). This is the direct consequence of the early requirements phase, where actors are initially identified and then translated to possible agents during the architectural design. Tropos role is an abstract characterisation of the behaviour of a social actor and a set of roles compose a position. The concept of position occurs in the architectural design phase, where agents are used to occupy previously identified positions.

More in details, we can say that in Tropos, an actor [6] models an entity with strategic goals and intentionality. An actor can represent a physical or a software agent as well as a role or a position. Goals (representing actors' strategic interests) can be of two different kinds: hard-goals and soft-goals, the latter have no specific criteria for deciding whether they are satisfied or not and are often used to model non-functional requirements. An actor can achieve his goals by adopting a Plan and/or using environment Resources. Actors can have a Dependency one on the other in order to satisfy their own goal or access a resource and their goals can be decomposed in terms of sub-goals that can be related with AND/OR relationships.

There are other two important elements of the Tropos meta-model: the Means/end Analysis (reporting that a means can satisfy a goal by using plans, resources or other goals) and the Contribution (showing that a goal, plan or resource can contribute to the achievement of some objective).

3.7 A First Proposal of a Unifying MAS Meta-model

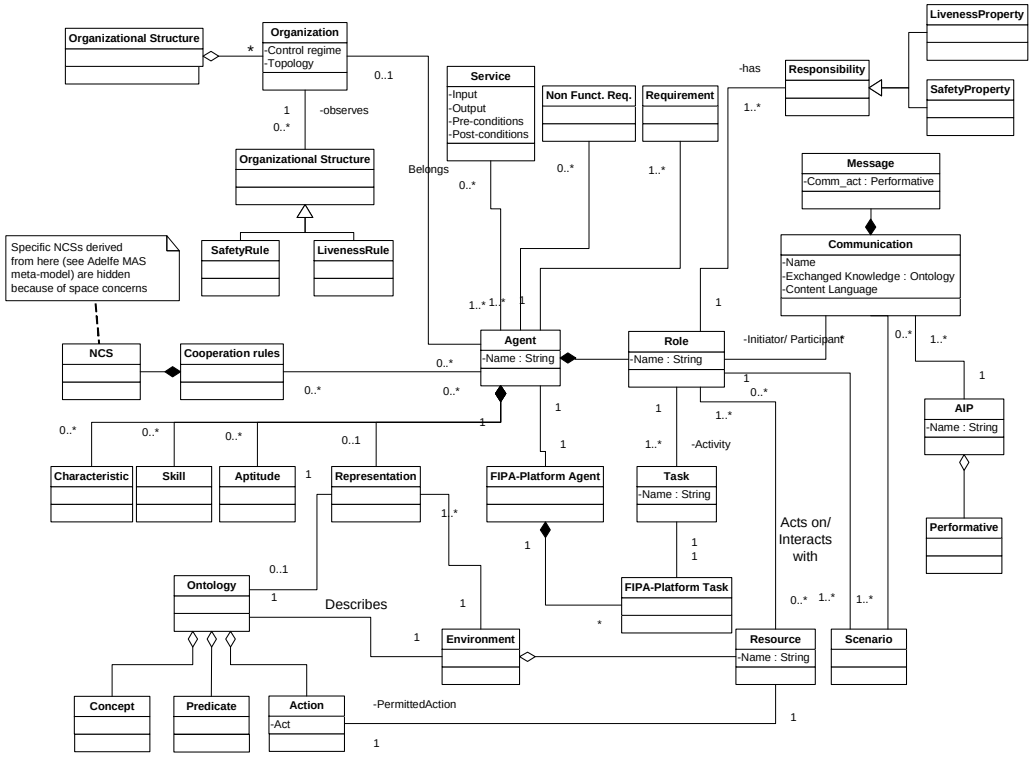

Figure 7. A first proposal of unifying MAS meta-model

Starting from the analysis of ADELFE, Gaia and PASSI MAS meta-models, in [3] there is a first proposal of a unifying MAS meta-model, which is presented in Figure 7. This metamodel is guided by the aim of creating societies without (ADELFE) or with predefined organizations, in accordance with the growing interest for open systems in which an organization cannot always be given during the design phase. To achieve this result the generic agent is enriched with all the properties an agent may have, being cooperative or not. Furthermore, this generic agent is composed of Gaia-like roles complemented by some PASSI features (tasks and a FIPA-compliant communication structure). This generic agent has two choices: belonging to an organization or following cooperation rules. Agents are implemented (at code level) in the PASSI way. The proposed meta-model is also characterized by the possibility of identifying in it the three domains (problem, agency, solution) discussed in the PASSI approach.

From the experience of merging these three models we learnt that their composition adds some significant improvements to the new structure since they complement each other in several aspects, for example the ADELFE representation that the agent has of its environment, the Gaia environment and the PASSI ontology, naturally relates by representing the fact that an agent has a representation (possibly affected by errors or uncertainty) of the environment expressed in terms of an ontological model of it.

3.8 A Comparison of MAS Meta-Models

The MAS meta-models presented in the previous subsections will now be compared with the precise aim of identifying their commonalities as a first step towards a common agreed part of an unified MAS Meta-Model (MMM), and studying their distinctive differences (that can positively enrich the collection of common elements by introducing ideas coming from just one or a few approaches).

Meta-Model	Common components	Peculiarities
ADELFE	Agent (*cooperative*), (*almost*) FIPA communications structure	Environment, Cooperation rules, Non cooperative situations (NCS), Skill, aptitude, characteristic
Gaia	Agent (type), Role, (*almost*) FIPA communications structure	Organization, Service, Resource, Environment, Organizational rules
INGENIAS	Agent, Role, Task, Interaction	Goal, Mental State, Organization, Group
PASSI	Agent, Role, Task, FIPA communications structure	Ontology, Resources, Requirements (functional and non-functional), Implementation Platform agent and task, Service, ServiceDescription
RICA	Agent (*type*), Role (*type*), Protocol (FIPA communications structure)	Norm, Institution type, Social/Interactive/ Communicative Role Type, Action Type, Social/Communicative Action Type
Tropos	Agent, Role, Plan *(similar to a task [6])*	Goals, Resources, Dependency
Unifying MMM	Agent, Role, Task, FIPA communications structure	Service, Environment, Characteristic, Skill, Aptitude, Resource, Ontology

Table 1. A comparison of the discussed MAS meta-models from the structural point of view

With regards to this comparison it is worth to note that in their MMM unification proposal (there the term unifying was used to justify the intention of finding a compromise that could summarize the three original MMMs) the authors of [3] proposed a comparison of the different MMMs based on some specific aspects classified according to four different criteria:

- Agent structure: this criterion deals with how each of the meta-models represents its core elements (commonly agents, roles, tasks).
- Agent interactions: how agents of different meta-models are supposed to interact using communications or the environment.
- Agent society and organizational structure: different MMMs differently approach the modelling of agents aggregations and cooperation structures.

- Agent implementation: this deals with the way the code-level structure of the agent system is specified.

In this work we are now dealing with a slight different problem: we are aiming at identifying a unified MAS meta-model, that could be a common reference point for the AgentLink AOSE community; this means that our attention is initially directed towards the identification of commonalities among the different works we examined; the appreciation of solutions that are not very diffused among the different MMMs will be part of a second step where specific contributions will be considered to enrich the first-release unified proposal. Because of this different aim, we now prefer to adopt a different comparison method that transversally cuts the previously listed criteria. In Table 1 we reported a list of common and different components of the discussed MMMs.

Elements that can easily be identified as common points of the different MAS meta-models are:

- Agent: it is present in all the methodologies. Gaia and RICA refer to it as *Agent Type*.
- Role: all methodologies but ADELFE includes the Role component in their MMMs.
- Task: it is present in INGENIAS, PASSI, and Tropos (with a slightly different meaning: Plan [6])
- FIPA communications structure: with this we mean a collection of elements representing agents communications as based on messages and following some specific Agent Interaction Protocol (AIP). Several methodologies have a good support for this kind of interaction mean: ADELFE, Gaia, INGENIAS, PASSI, and RICA.

From the analysis of the most characterising non-common elements of the studied MMMs we can see:

- Environment: it is present in ADELFE, INGENIAS, Gaia and the Unifying MMM. In ADELFE it is seen as a possible communication way for agents, in INGENIAS as application interfaces and resources.
- Organization (and social structures): in Gaia agents collaborate within organizations that are governed by Organizational Rules (a similar structure can be found in RICA). Organizations in INGENIAS can be structured in Groups and its dynamics is described in terms of Workflows.
- Cooperation related elements: ADELFE has a deep structure about cooperations, it includes: Cooperation Rules and an hierarchy of Non Cooperative Situations (NCS); RICA has Social Roles and Actions; in INGENIAS organizations contain Workflows describing the collaborative work.
- Mental attitudes and states of agent: ADELFE agent is modelled in terms of Skills, Aptitudes and Characteristics; INGENIAS agent is intentional, and has a Mental State (Goals, Facts, Beliefs), a Mental State Manager and a Mental State Processor; PASSI agent knowledge is based on an extensive model of domain ontology; Tropos methodology is based on the concept of goal as a problem decomposition entity and each agent acts to reach the goals it has been assigned to.

- Services and Resources: Gaia and PASSI define a similar concept of Service (PASSI includes a ServiceDescription too as an implementation level transposition of the agent Service). Resources are modelled in Gaia, INGENIAS, PASSI and Tropos (in this latter with more details).

The proposed analysis will directly influence the adoption of each single element of the unified MAS meta-model described in the next section and the different meanings that each methodology gives to concepts that are similar in their name become a challenge for the definition of the glossary of terms that will complements this unified MAS meta-model.

4. Towards a Unified MAS Meta-model: the AgentLink AOSE TFG Proposal

In this section we will describe the process that brought us to identify the AgentLink proposal of MMM. The work included the identification of a core subset of MAS meta-model components and the clear definition of the meaning of these components by establishing a sort of glossary. Then, defining the relationships between those components we completed the MAS meta-model.

The different talks given during the different meetings of this TFG, as well the work done in the FIPA Methodology and Modelling TCs, constitute the background of this identification work:

- Existing methodologies for which authors are involved in this TFG are: ADELFE, Gaia, INGENIAS, PASSI, Tropos and RICA, and presented above,
- The first reflection on modelling structures that the FIPA Modelling TC proposed [Odell et al. 04],
- And an attempt for a unifying meta-model done in [3].

The second step would be to agree on the components that would be identified and included in a common MAS meta-model:

- In a first live phase, what a MAS meta-model should include will be defined with the definition of concepts like agent, role, task (or plan), communication (message, protocol, performative, etc.),
- In a second time, after having launched a discussion during the meeting, concepts such as environment, goal, organisation (society, group, institution, etc.), resource, service, would be discussed off-line,
- Other components such as ontology, dependency, action, mental states, organisation rules, norms, skills, aptitudes, characteristics (or similar BDI concepts) could be studied later on.

4.1 Identifying the Basic Components

4.1.1 Defining “Agent”

The starting point for the definition of “agent” was the definition given by the FIPA Methodology TC[2]. The aim of this discussion was to enumerate the minimal properties

[2] http://www.pa.icar.cnr.it/~cossentino/FIPAmeth/glossary.htm

an entity must have to be considered as an agent. The essential properties of an agent are its capabilities to act, its autonomy, its communication with others (because the notion of collective is important in many approaches) by interacting, its perception of its environment. Other properties were given, such as proactivity, reactivity, ability to move, or ability to reproduce itself, what could lead to defining different kinds of agents, for instance cognitive or reactive ones. But other agents exist and they are neither cognitive nor reactive or may be considered as a mix of these two types. A double inheritance from "agent" in the MAS meta-model could solve the problem but since it was not mandatory to be exhaustive and define all kinds of agents, participants finally agreed on this minimal definition for an agent as an entity:

- which is capable of acting in its environment,
- which is autonomous: this means that an agent has control over its own behaviour based on internal or external stimuli,
- which can communicate with other agents,
- and which is capable of perceiving its environment.

Some features such as ability to move or reproduce were not considered as mandatory and concepts like skills or capabilities, for example, are included in the fact that an agent is able to act. Furthermore since this definition of a "generic" agent is sufficient to define a reactive agent (to be reactive is a specialisation of being capable of acting in an environment), having a definition for this latter concept and differentiate the cognitive nature of an agent from the reactive one was not considered as useful at this time. A cognitive agent will be considered as an agent:

- which is proactive: this means, its behaviour is driven by a set of tendencies, in the form of individual objective, or a satisfaction/survival function which it tries to optimise, or desire, or emotion,
- and which uses a representation of its environment.

4.1.2 Defining "Role"

Starting from the definition used in PASSI, a role has been defined as "an abstraction of a portion of a social behaviour of an agent".

4.1.3 Defining "Task"

The discussion for defining the concept of "task" was based on the FIPA TC Modelling definition and the one PASSI gives. For FIPA TC Modelling, a task is a part of the agent behaviour and a behaviour is the observable effects of an operation or an event, including the results, and it specifies the computation that generates the effect of the behavioural features. For PASSI, a task specifies the computation that generates the effect of the behavioural features.

To be general enough, the AOSE TFG participants agreed that a task "specifies a (set of) activity(ies) that generates some effects".

A role uses observable and not observable tasks, being characterised by the first ones since they are visible; an agent could also do something that is not a part of playing a role.

4.1.4 Defining “Environment”

The definition of the Environment concept involves several aspects (it is in fact the central topic of another AgentLink III TFG, the Environment TFG) and raises a great number of questions:

1. PASSI enlarges the software engineering dichotomy about problem and solution domains by introducing the intermediate agency domain. In the same way could it be possible to speak about a problem environment (in which the system exists) an agency environment where agents live and a solution environment where object-oriented implementations of agents are deployed?
2. Should the environment be characterized in a way similar to the one Russell and Norvig did in their book [Russell and Norvig 95], in which an environment can be accessible or not, dynamic or not, deterministic or not, continuous or not?
3. With regards to the solution domain, could we speak about a “system outer environment”, a “system inner environment” and a “controllable or not environment”?

Unfortunately the scope of this TFG was too narrow to have a discussion about what is really an environment and we decided to limit our work to the definition of an environment as “something that an agent can interact with and/or perceive”. This definition goes in the same direction of the one given by the FIPA TC Methodology in which the environment represents all that is external to the agent, and that includes the social environment and the physical environment.

4.1.5 Defining “Organisation”

In Gaia, an organisation aggregates agents and in INGENIAS, an organisation is an autonomous entity, with a purpose, the organizational structure is defined with Groups.

Organisations can be given by roles (for example, in a soccer match where there is a goalkeeper) but they can also emerge from interactions between agents (for example, in adaptive MAS for which ADELFE is specialised). The discussion concerned the nature of the link between “agent” and “organisation” because both an agent and an organisation can be seen as aggregations of roles. The definition for “organisation” was not completed during the meeting, it is just said as being composed of roles.

4.2 The Current Unified MAS Meta-model Proposal

With the above reported definition of the concepts , the relationships among them could be drawn and in this way it has been possible to build a first draft of a unified MAS meta-model on which AOSE TFG participants agreed and which is shown in Figure 8.

Obviously, the central concept is a generic agent from which a cognitive agent can be derived. An agent is situated in an environment for which it is able to build representations if it possesses cognitive capabilities. An agent can also be part of the environment of other agents, that explains the bidirectional link between these two concepts. An agent communicates with others and to communicate with an intended purpose can use FIPA-compliant conversations. A role is related to an agent (agents play roles); some agents can use tasks without playing any role and therefore it becomes necessary to relate “agent” to “task” with a 0..n cardinality. Finally, following the previous discussion on the definition of “organisation”, an organisation may be composed of agents but also of roles depending on the concerned methodology. This is

expressed on the MAS meta-model by following navigation links from “agent” to “organisation” via “role”.

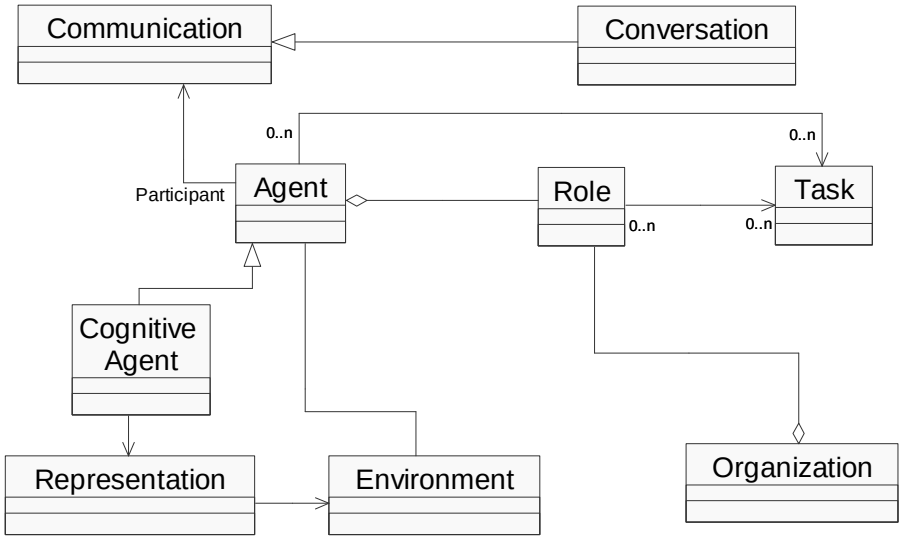

Figure 8. The AgentLink AOSE TFG MAS meta-model proposal

5. Conclusions

As far as agent-oriented methodologies have matured in the last years, there is a need for consolidating concepts and methods for agent-oriented development. In the AgentLink AOSE TFG the approach towards this goal has been to study MAS meta-models for different methodologies and find an agreement on basic concepts. This is a work in progress and the proposed unified MAS meta-model and concepts definition will be extended. In the end, this will serve as a foundation for future agent-oriented modelling languages and development tools.

6. References

[1] Bergenti F., Gleizes M-P., and Zambonelli F., Methodologies and Software Engineering for Agent Systems, Editions Kluwer to appear in 2004.

[2] Bernon C., Camps V., Gleizes M-P., and Picard G., Tools for Self-Organizing Applications Engineering, In Di Marzo Serugendo G., Karageorgos A., Rana O.F., Zambonelli F., eds. First International Workshop on Engineering Self-Organising Applications (ESOA), Melbourne, Australia, July 2003, LNCS 2977, Springer Verlag publ., 2004.

[3] Bernon C., Cossentino M., Gleizes M-P., Turci P., and Zambonelli F., A Study of some Multi-agent Meta-models. In Agent-Oriented Software Engineering V: 5th International Workshop, AOSE 2004. Revised Selected Papers. Editors: James Odell, Paolo Giorgini, Jörg P. Müller. LNCS Series, pp62-77, Vol. 3382 / 2005. Jan. 2005.

[4] Bernon C., Gleizes M. P. Adelfe MAS meta-model. Agentlink III – AOSE TFG2, Ljubljana (Slovenia). 28 Feb-2 Mar 2005. Online at: http://www.pa.icar.cnr.it/~cossentino/ al3tf2/docs/adelfe_ppt.pdf

[5] Bertolini D., Perini A., Susi A. and Mouratidis H. TROPOS meta-model. Agentlink III – AOSE TFG2, Ljubljana (Slovenia). 28 Feb-2 Mar 2005. http://www.pa.icar.cnr.it/~cossentino/al3tf2/docs/tropos_bertolini.ppt

[6] Bertolini D., Perini A., Susi A., and Mouratidis H., The Tropos Visual Language. A MOF 1.4 Compliant Meta-model. Agentlink III AOSE TFG 2. Ljubljana (Slovenia). 28 Feb-2 Mar 2005. Available online at: http://www.pa.icar.cnr.it/cossentino/al3tfg2/docs/tropos.pdf

[7] Bresciani P., Giorgini P.,Giunchiglia F., Mylopoulos J., and Perini A., Tropos: An Agent-Oriented Software Development Methodology. Journal of Autonomous Agent and Multi-Agent Systems, 8(3):203 – 236, May 2004.

[8] Brinkkemper, S., Lyytinen, K., and Welke, R., Method Engineering: Principles of Method Construction and Tool Support. Proceedings of the IFIP TC8, WG8.1/8.2 Working Conference on Method Engineering on Method Engineering : Principles of Method Construction and Tool Support. S. Brinkkemper, K. Lyytinen and R. J. Welke Editors, Chapman & Hall, Ltd., Atlanta (Georgia, USA) 1996.

[9] Burrafato P., and Cossentino M., Designing a multi-agent solution for a bookstore with the PASSI methodology. In Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS-2002) - 27-28 May 2002, Toronto (Ontario, Canada) at CAiSE'02.

[10] Caire, G., Coulier, W., Garijo, F. Gomez, J., Pavon, J., Leal, F., Chainho, P., Kearney, P., Stark, J., Evans, R., and Massonet, P., Agent Oriented Analysis using MESSAGE/UML. In Wooldridge, M., Weiss, G., and Ciancarini. P. (Eds.): The Second International Workshop on Agent-Oriented Software Engineering (AOSE 2001). Lecture Notes in Computer Science, Vol. 2222. Springer-Verlag, Berlin Heidelberg New York (2002) 119-135.

[11] Castro J., Kolp M., and Mylopoulos J., A Requirements-Driven Development Methodology. In Proceedings of the 13th International Conference on Advanced Information Systems Engineering (CAiSE'01), Interlanken, Switzerland, June 2001.

[12] Cervenka R., Trencansky I., Calisti M., and Greenwood D., AML: Agent Modeling Language Toward Industry-Grade Agent-Based Modeling, In Agent-Oriented Software Engineering V: 5th International Workshop, AOSE 2004. Revised Selected Papers. Editors: James Odell, Paolo Giorgini, Jörg P. Müller. LNCS Series, pp31-46, Vol. 3382 / 2005. Jan. 2005.

[13] Chella A., Cossentino M., Sabatucci L., Seidita V. The PASSI and Agile PASSI MAS meta-models. Agentlink III – AOSE TFG2, Ljubljana (Slovenia). 28 Feb-2 Mar 2005. http://www.pa.icar.cnr.it/~cossentino/al3tf2/docs/passi_ppt.zip

[14] Cossentino M., From Requirements to Code with the PASSI Methodology. In Agent-Oriented Methodologies, B. Henderson-Sellers and P. Giorgini (Editors). Idea Group Inc., Hershey, PA, USA. 2005 (in printing)

[15] Gómez-Sanz Jorge J., Pavòn J. INGENIAS MAS meta-model. Agentlink III – AOSE TFG2, Ljubljana (Slovenia). 28 Feb-2 Mar 2005. http://www.pa.icar.cnr.it/ ~cossentino/al3tf2/docs/ingenias_slides.zip

[16] Gómez-Sanz, J. and Pavón, J., *Meta-modelling in Agent Oriented Software Engineering.* Proc. 8th Ibero-American Conference on AI, Iberamia 2002, Seville, Spain, November 12-15, 2002.. Springer-Verlag, LNCS 2527, pp. 606-615.

[17] Gómez-Sanz, J., and Pavón, J., Meta-modelling in Agent Oriented Software Engineering. Advances in Artificial Intelligence - IBERAMIA 2002. LNCS 2527. Springer-Verlag (2002): 606-615.

[18] Guessom Z. MAS Meta-Models and MDA. Agentlink III – AOSE TFG2, Ljubljana (Slovenia). 28 Feb-2 Mar 2005. http://www.pa.icar.cnr.it/~cossentino/al3tf2/docs/zahia_slovenia.pdf

[19] J. Manuel Serrano, S. Ossowski, S. Saugar. The RICA metamodel: an organizational stance on agent communication languages. Agentlink III AOSE TFG 2. Ljubljana (Slovenia). 28 Feb-2 Mar 2005. Available online at: http://www.pa.icar.cnr.it/cossentino/al3tfg2/docs/rica_serrano.pdf

[20] Kusek M. and Jezic G., Modeling Agent Mobility with UML Sequence Diagram. Agentlink III – AOSE TFG2, Ljubljana (Slovenia). 28 Feb-2 Mar 2005. Online at: http://www.pa.icar.cnr.it/~cossentino/al3tf2/docs/kusek_ppt.ppt

[21] Luck M., McBurney P., Preist C. Agent Technology: Enabling Next Generation Computing. A Roadmap for Agent Based Computing. Online at http://www.agentlink.org/roadmap.

[22] Odell J., Norine M., and Levy R., A Metamodel for Agents, Groups and Roles. In Agent-Oriented Software Engineering V: 5th International Workshop, AOSE 2004. Revised Selected Papers. Editors: James Odell, Paolo Giorgini, Jörg P. Müller. LNCS Series, pp78-92, Vol. 3382 / 2005. Jan. 2005.

[23] Pavón, J., and Gómez-Sanz, J.J., Agent Oriented Software Engineering with INGENIAS. In Multi-Agent Systems and Applications III, 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003. Lecture Notes in Computer Science 2691, Springer Verlag (2003) 394-403.

[24] Ralyte, J., and Rolland, C., An approach for Method Reengineering. Proceedings of the 20th International Conference on Conceptual Modeling, ER2001, Yokohama, Japan, November 27-30 2001

[25] Russel, S. and Norvig P., Artificial Intelligence: A Modern Approach. Prentice Hall Series, 1995.

[26] Searle J.R., Speech Acts. Cambridge University Press, 1969.

[27] Serrano J. M., and Ossowski S., On the Impact of Agent Communication Languages on the Implementation of Agent Systems, Eighth International Workshop CIA 2004 on Cooperative Information Agents, September 27 - 29, 2004, Erfurt, Germany.

[28] Serrano J.M., Ossowski S. On the Impact of Agent Communication Languages on the Implementation of Agent Systems, Eighth International Workshop CIA 2004 on Cooperative Information Agents, September 27 - 29, 2004, Erfurt, Germany

[29] Serrano J.M., Ossowski S., Saugar S. The RICA meta-model. Agentlink III – AOSE TFG2, Ljubljana (Slovenia). 28 Feb-2 Mar 2005. http://www.pa.icar.cnr.it/~cossentino/al3tf2/docs/serrano_ppt.pdf

[30] Trencansky, I., Agent Modeling Language (AML). Agentlink III – AOSE TFG2, Ljubljana (Slovenia). 28 Feb-2 Mar 2005. Online at: http://www.pa.icar.cnr.it/~cossentino/al3tf2/docs/aml_trencansky.pdf

[31] Wooldridge M., Jennings N.R., and Kinny D., A Methodology for Agent-Oriented Analysis and Design, In Proceedings of the 3rd International Conference on Autonomous Agents (Agents 99), pp 69-76, Seattle, WA, May 1999.

[32] Zambonelli F. Jennings N., and Wooldridge M., Developing Multiagent Systems: the Gaia Methodology, ACM Transactions on Software Engineering and Methodology, 12(3):417-470, July 2003.

[33] Zambonelli F., Open Directions in AOSE. Agentlink III – AOSE TFG2, Ljubljana (Slovenia). 28 Feb-2 Mar 2005. Online at: http://www.pa.icar.cnr.it/~cossentino/al3tf1/programme.html