A Study of Some Multi-Agent Meta-Models

Carole Bernon, Massimo Cossentino, Marie-Pierre Gleizes, Paola Turci, Franco Zambonelli

A Study of Some Multi-agent Meta-models

Carole Bernon[1], Massimo Cossentino[2], Marie-Pierre Gleizes[1], Paola Turci[3], and Franco Zambonelli[4]

[1] IRIT - University Paul Sabatier - Toulouse, Cedex 4 (France)
{bernon, gleizes}@irit.fr
[2] Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR) - Consiglio Nazionale delle Ricerche (CNR)- Palermo, Italy
cossentino@pa.icar.cnr.it
[3] Dipartimento di Ingegneria dell'Informazione - Universitá degli Studi di Parma - Parma, Italy
turci@ce.unipr.it
[4] Dipartimento di Scienze e Metodi dell'Ingegneria - Universitá di Modena e Reggio Emilia - Reggio Emilia, Italy
franco.zambonelli@unimo.it

Abstract. Several agent-oriented methodologies have been proposed over the last few years. Unlike the object-oriented domain and unfortunately for designers, most of the time, each methodology has its own purposes and few standardization works have been done yet, limiting the impact of agent design on the industrial world. By studying three existing methodologies - ADELFE, Gaia and PASSI - and the concepts related to them, this paper tries to find a means to unify their meta-models. Comparing a certain number of features at the agent or system level (such as the agent structure, its society or organization, its interactions capacities or how agents may be implemented) has enabled us to draw up a first version of a unified meta-model proposed as a first step toward interoperability between agent-oriented methodologies.

1 Introduction

Over the years several methodologies and approaches have been proposed for the development of multi-agent systems. Nevertheless many users have still trouble finding a method and notation that would satisfy their needs completely. What seems to be widely accepted is that a unique specific methodology cannot be general enough to be useful to everyone without some level of personalization. As a matter of fact the need for systematic principles to develop situation-specific methods, perceived almost from the beginning by the object-oriented community, has led to the emergence of the proved successful in developing object-oriented information systems [1]. Its importance in the object-oriented context should be evaluated considering not only the direct influence (not so many companies and individuals work in this specific way) but mainly the indirect consequence. The most important and diffused development processes (e.g., the Rational Unified

Process [2]) are in fact not rigid, instead they are a kind of framework within which the single designer can choose his/her own path.

We believe that the agent-oriented community should follow a similar path, trying to adapt the method engineering for using it in agent-oriented design. It is in this ambit that the FIPA Methodology TC[1] is situated. Its aim, and our aim as members of the committee, is to propose quite an open approach that allows the composition of a very large repository of human experiences (design process is first of all a human process) that could be expressed in terms of a standard notation.

Right from the beginning however it was clear that adopting the method engineering approach in the AOSE context is not a plain task. In the object-oriented context the construction of method fragments, the assembling of the methodology with them 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. The situation concerning the agent-oriented approach is quite different since there is not a commonly accepted definition of the concept of agent and related meta-model of the multi-agent system - a structural representation of the elements (agent, role, behavior, ontology, etc.) that will compose the actual system with their composing relationships. Since a meta-model is a means of unifying concepts, the lack of a unique MAS meta-model consequently leads to each methodology having its own concepts and system structure.

Analyzing the process of designing a system (object or agent-oriented) we have come to the conclusion that it consists in instantiating the system meta-model that the designers have in their mind in order to fulfill the specific problem requirements. In the agent world this means that the meta-model is the critical element when applying the method engineering paradigm, because of the variety of the methodology MAS meta-models. Indeed the first step of the composition process should consist in a selection of the elements that compose the meta-model of the MAS the designers will build. The MAS meta-model so derived will be useful in the method fragment selection phase at least in order to avoid the selection of methods referring to different elements. But without a unique MAS meta-model, the various concepts and system structures characterizing the different methodologies could make very laborious or even impossible to carry out the method fragment composition.

Bearing in mind the above described composition process centered on the MAS meta-model, the main scope of this work is two-fold: (i) to analyze the MAS meta-models of three existing design methodologies - ADELFE, Gaia and PASSI - in order to support what has been asserted above; (ii) to design a unifying MAS meta-model, obtained by merging the most interesting aspects of each meta-model, with the aim of making a significant step toward the definition of a unique omni-comprehensive MAS meta-model.

We would like to emphasize that despite the fact that the choice of the three methodologies was a logic consequence of the people involved in writing the

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

paper, we think that all in all the heterogeneousness of the three methodologies allows us to draw interesting remarks.

2 ADELFE Meta-model

ADELFE[2] is a methodology devoted to software engineering of adaptive multi-agent systems [3], [4]. Adaptive software is used in situations in which the environment is unpredictable or the system is open. To solve these problems ADELFE guarantees that the software is developed according to the AMAS (Adaptive Multi-Agent System) theory [5].

According to this theory, building a system which is functionally adequate (which realizes the right desired global function) is achieved by designing agents with a cooperation-driven social attitude. Agents composing an AMAS ignore the global function of the system, only pursue a local goal and try to always keep cooperative relations with one another. They are called "cooperative agents".

The MAS meta-model adopted for ADELFE (cf. Figure 1) is fundamentally explained by this specialization of ADELFE and by the features a cooperative agent possesses. Its life cycle is a classical one; it consists in having perceptions, taking decisions and then doing actions (perceive-decide-act).

Besides local cooperation rules are enabling it to detect and solve Non Cooperative Situations (NCS). These NCS are cooperation failures (e.g., cooperation protocol not obeyed, unpredictable situations) that are, from its point of view, inconsistent with its cooperative social attitude. Different kinds of such failures can be detected according to the context of the concerned application, such as Incomprehension (an agent does not understand a perceived signal), Ambiguity (it has several contradictory interpretations for a perceived signal), Incompetence (it cannot satisfy the request of another one), Unproductiveness (it receives an already known piece of information or some information that leads to no reasoning for it), Concurrency (several agents want to access an exclusive resource), Conflict (several agents want to realize the same activity) or Uselessness (an agent may make an action that is not beneficial, according to its beliefs, to other agents). When detecting a NCS, an agent does all it is able to do to solve it to stay cooperative for others. For example, faced up with an incomprehension situation, it does not ignore the message but will transmit it to agents that seem (from its point of view) relevant to deal with it.

An agent possesses world representations that are beliefs concerning other agents, the physical environment or the agent itself. These representations are used by the agent to determine its behavior. If an agent has representations that may evolve (e.g., a semantic network), these representations can be expressed using a multi-agent system. A representation can be shared by different agents.

[2] ADELFE is a French acronym meaning "toolkit to develop software with emergent functionalities". It was a French RNTL-funded project (2000-2003) which partners were: ARTAL Technologies and TNI-Valiosys from industry, and IRIT and L3I from academia. See http://www.irit.fr/ADELFE

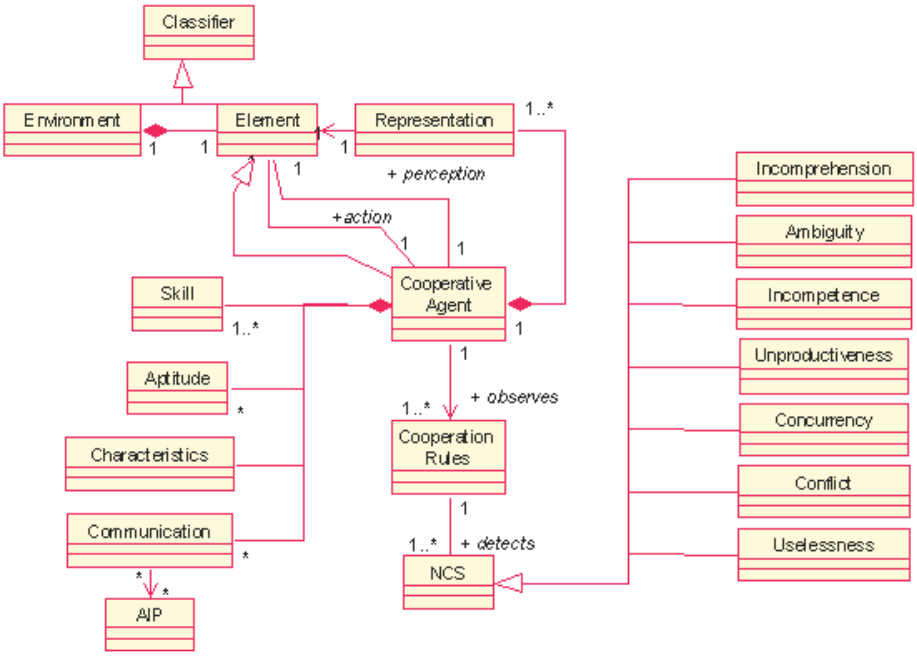

Fig. 1. The Multi-Agent System Meta-Model Adopted in ADELFE

An agent is able to communicate with other agents or its environment. This communication can be done in a direct manner (by exchanging messages) or an indirect one (through the environment). Tools that enable an agent to communicate are interaction languages. When an agent uses a direct communication through messages exchanges, AIPs may also be used to express the communication pattern between agents.

An agent can interact with its environment (physical or social) by means of perceptions and actions. For an agent, an action is a way to act on its environment during its action phase and a perception enables it to receive information from this environment.

Aptitudes show the ability of an agent to reason both about knowledge and beliefs it owns. For instance, an aptitude of a software agent can be expressed by an inference engine on a base of rules or any other processing on perceptions and world representations. Aptitudes can also be expressed using data, e.g. an integer value which represents the exploration depth of a planning tree.

An agent owns some skills that are specific knowledge that enable it to realize its own partial function. For instance, a skill may be a simple datum which is useful to act on the world (e.g., an integer distance which represents the minimal distance a robot has to respect to avoid obstacles) or may be more complex when expressing a reasoning that the agent makes during its decision phase (e.g., a reasoning to avoid obstacles). If they are complex and able to evolve, skills may also be implemented by MAS.

An agent may possess some characteristics which are its intrinsic or physical properties. It may be, for instance, the size of an agent or the number of legs of a robot-like or ant-like agent. A characteristic may also be something the agent can perform to modify or update one of its properties; for example, if the agent is an ant, enabling it to modify its number of legs.

3 Gaia Meta-model

The first version of the Gaia methodology was designed to handle small-scale, closed agent-based systems [6]. Consequently, it modeled agents, roles, interactions, but missed in modeling explicitly the social aspects of a multi-agent system. The official extension of Gaia extends Gaia based on the key consideration that an organization is more than simply a collection of roles and agents [7]. 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. This is quite evident from the MAS meta-model (see Figure 2): the methodology is focused on the organizational structure of the system and all other concepts - agents, roles, services interactions - turn around the concept of organization and are modeled in order to better specify the relationship between the different entities in the context of a specific organization.

Having a deeper look at the MAS meta-model for the extended version of Gaia we notice that the basic building blocks of the former version of Gaia - namely agents, roles, activities, services, and protocols - are still present. In particular: 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; an agent plays a role by actualizing the behavior in terms of services to be activated and de-activated in dependence of specific pre- and post-conditions.

The extended version of Gaia starts from the above basic concepts and enriches them by putting them in the context of a specific environment and of a specific organization.

We emphasize Gaia does not deal with the requirements capture phase, and considers the requirements statement simply as an input for the methodology. However, the environment in which a multi-agent system is immersed is elected to a primary analysis and design abstraction in order to promote a clear understanding of the overall system. The environment abstraction explicitly specifies all the entities and resources a multi-agent system may interact with, restricting the interactions by means of the permitted actions. Thus, to some extent, the explicit representation of the environmental resources that can be manipulated by agents can be considered as a reference to the problem domain.

The explicit representation of an agent organization and the central role of organizational concepts come into play with the abstractions of organizational rules and organizational structures.

Organizational rules have the scope of specifying some constraints that the organization has to observe. They may be global, affecting the behavior of the

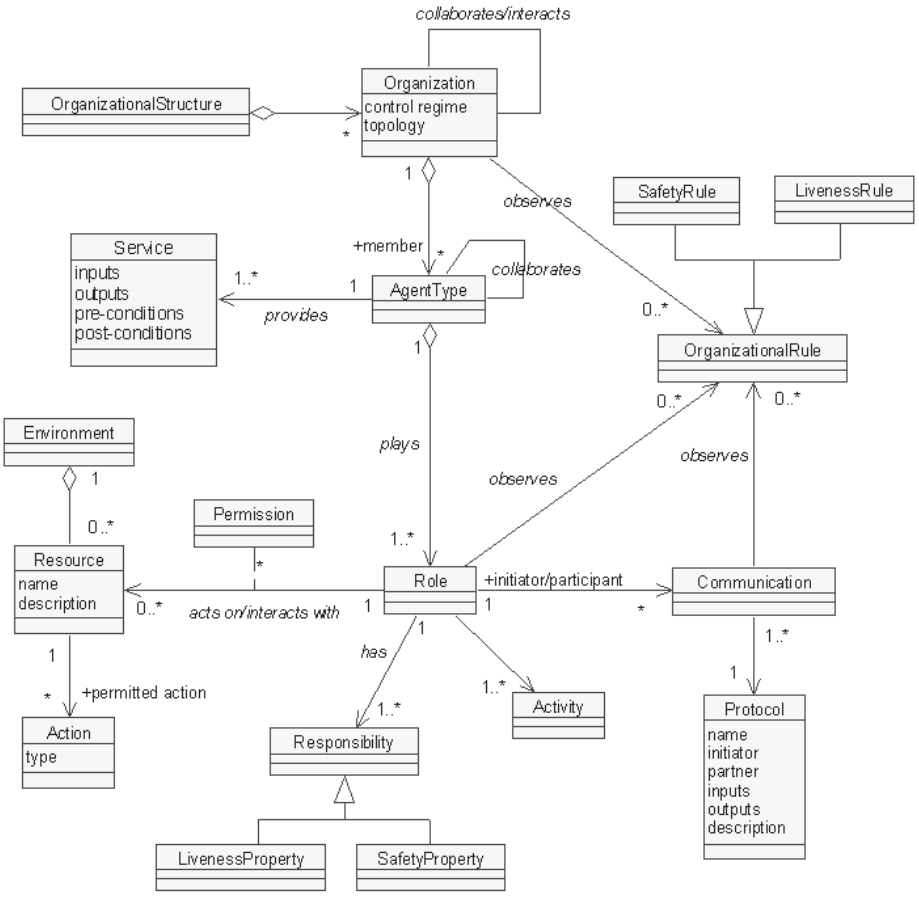

Fig. 2. The Multi-Agent System Meta-Model Adopted in Gaia

society as a whole, or concerning only specific roles or protocols. Organization structure on the other hand aims at making the overall architecture of the system, that is the position of each role in the organization and its relationship with other roles, explicit.

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. Therefore, in the extended version of Gaia, the organizational structure is not implicitly defined via the role model, instead the identification of the roles is explicitly derived from an analysis of the chosen organizational structure. As a consequence the role model and the related interaction model will be completely defined in the design phase when an accurate identification of the organizational structure will take place.

4 PASSI Meta-model

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 [8] MAS meta-model (see Figure 3) we found that this duality does not properly reflect the needs of an agent approach and therefore in our meta-model we introduce the agency domain. It represents the transition from problem-related concepts to the corresponding agent solution (that is not at an implementation level but it is still a logical abstraction). In this (agent) domain we will design all the agent-related elements like roles, communications, and the same agents, in order to define the solution to the requirements drawn in the problem domain. Since we decided to implement our solution with a FIPA-based infrastructure, we do not have any agent-oriented language that can be used to code the system but we map our choices to an object-oriented implementation level. In PASSI we do not think this is a limit because this presents several advantages, in fact the agent paradigm is used where it is more profitable: providing an abstraction level that could enable a simpler solution where classical software engineering concepts like decoupling, information hiding and responsibility division among components are naturally pursued. Moreover, final code-level implementation is devoted to affordable object-oriented languages that can be managed by several already skilled programmers and can be easily tested referring to a broad existing experience and a huge literature.

In the PASSI MAS meta-model (Figure 3), the Problem Domain 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. There is a strong point behind these choices: a lot of highly skilled designers are already present in different companies and can be more easily converted to the use of an agent-oriented approach if they are already confident with some of the key concepts used within it. Analysis related issues (like requirements and scenarios) being situated in the highest abstraction phase are strategic in enabling this skill reuse and allow a smooth entering in the new paradigm.

Ontological description of the domain is composed 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). This represents the domain in a way that is substantially richer than the classic structural representations produced in the object-oriented analysis phase. As an instance, we can consider ontologies devoted to reasoning on strategies or problem solving methods whose essence is very difficultly captured in object-oriented structures [9].

Resources are the last element of the problem domain. They can be accessed/shared/manipulated by agents. A resource could be a repository of data (like a relational database), an image/video or also a good to be sold/bought. We prefer to expressly model them since goals of most systems are related to using and capitalizing available resources.

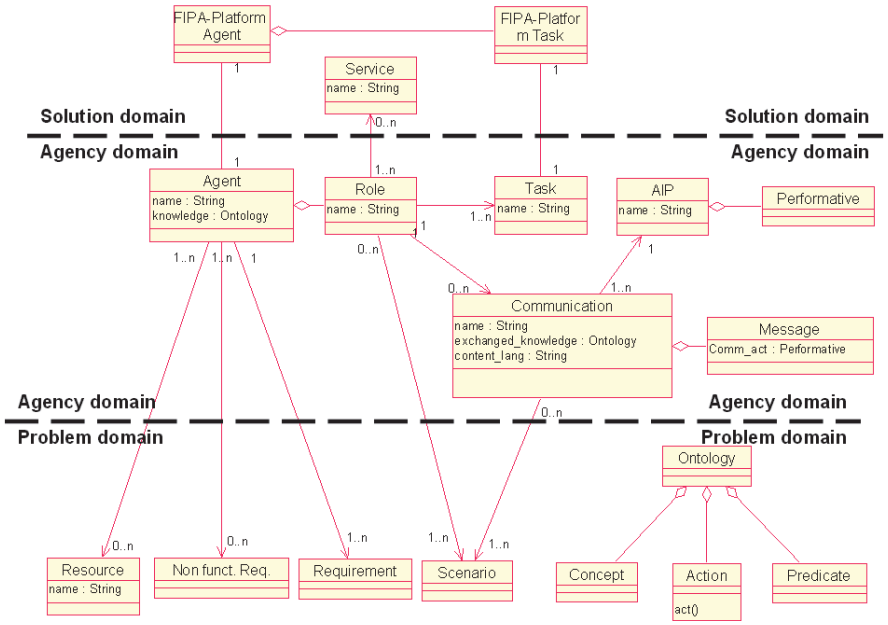

Fig. 3. The Multi-Agent System Meta-Model Adopted in PASSI

The Agency Domain contains the elements of the agent-based solution. None of these elements is directly implemented; they are converted to the correspondent object-oriented entity that constitutes the real code-level implementation. The concept of agent is the real center of this part of the model; each agent in PASSI is responsible for realizing some functionalities descending from one or more requirements. The direct link between a requirement and the responsible agent is one of the strategic decisions taken when conceiving PASSI. Sometimes an agent has also access to available resources. This could happen because it accesses the corresponding information (for example stored in a DB) or it can perceive it using its sensors (like in the case of embodied robotic agents sensing the environment). Each agent during its life plays some roles; that are portions of the agent social behavior characterized by some specificity such as a goal, or providing a functionality/service.

From this definition easily descends that roles could use communications in order to realize their relationships or portions of behavior (called tasks) to actuate the role proclivity. In PASSI, the term task is used with the significance of atomic part of the overall agent behavior and, therefore, an agent can accomplishing its duties by differently composing the set of its own tasks. Tasks cannot be shared among agents, but their possibilities could be offered by the agent to the society as services (often a service is obtained composing more than one task); obviously according to agent autonomy, each single agent has the possibility of accepting or refusing to provide a service if this does not match its personal attitudes and will.

A communication is composed of one or more messages expressed in an encoding language (e.g. ACL [10]) that is totally transparent to agents. The message content could be expressed in several different content languages (SL, KIF, RDF, ...); we chose to adopt RDF [11][12] and the PASSI supporting tool (PTK) offers a concrete aid in generating the RDF code from the designed ontology. Each communication explicitly refers to a piece of ontology (in the sense that information exchanged are concepts, predicates or actions defined in the ontology) and its flow of messages is ruled by an interaction protocol (AIP) that defines which communicative acts (the predefined semantic of the message content [13]) may be used in a conversation and in what order the related messages have to be sent to give the proper meaning to the communication.

The Implementation Domain describes the structure of the code solution in the chosen FIPA-compliant implementation platforms (like FIPA-OS or JADE) and it is essentially composed of three elements: (i) the FIPA-Platform Agent that represents the implementation class for the agent entity represented in the Agency domain; (ii) the FIPA-Platform Task that is the implementation structure available for the agent's Task and, finally, (iii) the Service element that describes a set of functionalities offered by the agent under a specific name that is registered in the platform service directory and therefore can be required by other agents to reach their goals. This description is also useful to ensure the system openness and the reusability of its components.

5 Comparison and Discussion

The three meta-models presented in the previous sections are very different and are a well representative example of the debate in the agent community about these strategic issues. In order to catch the essence of each of them we should consider the specific approach followed by the respective authors and the system structure pursued by them.

The ADELFE meta-model (Figure 1) clearly represents the aim of solving the problem with an adaptive MAS and therefore a great effort is done in order to study, through "cooperation rules", all the situations that could enable or inhibit the cooperation among agents. The cognitive and behavioral representations of the agent are performed in terms of its aptitudes, skills, characteristics, and representations (social or physical); agents interact via direct communications or the environment.

The Gaia meta-model (Figure 2) is mostly devoted to represent a MAS system as a social organization. For this reason, roles more than agents are the central subject of the model, as the basic building block of agents. While a Gaia role is characterized by an activity structure and by internal responsibilities, an organization is characterized by a structure - i.e., a set of roles interacting with each other according to specific protocols - and by "organizational responsibilities" or "organizational rules" - i.e., the constraints that the actual evolution of an organization mush adhere to. Little or no attention is paid to cognitive and representational issues.

The PASSI meta-model (Figure 3) aims at conciliating classical software engineering concepts like problem and solution domain with the potentiality of the agent-based approach while pursuing the goal of a traceability of the solution from requirements to the related code implementation. Authors clearly points to a FIPA-based implementation of their systems and therefore communications and implementation issues are typical of those specifications and most common related platforms (FIPA-OS, JADE). The convergence between agents and traditional issues of software engineering is obtained by introducing a new abstraction layer (agency domain) that complements the well-known problem-solution domain dichotomy.

Generally speaking, it is interesting to note that none of the discussed approaches explicitly refers to one of specific 'classical' agent architectures (like BDI or purely reactive agents) but these are seen as some kind of low level architectures that can be adopted during the MAS implementation. Only PASSI partially limits this range by referring to FIPA-compliant systems but this does not seem to be a real constraint since such systems have been used to implement all of the cited architectures.

In the following we will compare these meta-models by looking at some of their specific aspects; specifically we will consider:

- Agent structure: this means how each of the meta-models represents the agent and its most common elements (namely roles).
- Agent interactions: agents of different meta-models are supposed to interact using communications or the environment. Communications are sometimes specified by attributes like interaction protocols, content language and so on.
- Agent society and organizational structure: the goal of some of these meta-models is to model a specific society or an organizational infrastructure constrained by rules that enforce agents to some collective or individual behavior.
- Agent implementation: the code-level structure of the agent system.

Each of the cited categories will now be diffusely discussed and this study will be used to compose a new unifying meta-model that will try later to take the best of the different approaches.

5.1 Agent Structure

Looking at agent structure and specifically at agent and role definitions in the different meta-models, we can find that the ADELFE meta-model is quite different from the others because it tries to constrain the agent behavior with a cooperative attitude. In fact the ADELFE meta-model is not centered on the role notion because designers have to focus on the ability an agent possesses to detect and solve cooperation failures by observing cooperation rules. If a designer gives roles to agents, by describing a task or protocols, he/she will establish a fixed organization for these agents. However, a fixed organization in an AMAS is not welcomed because this organization must evolve to enable the system adaptation (cf. section 5.3).

The PASSI agent is the composition of some roles but each role is defined as the manifestation of the agent activity in some scenarios, it is associated with one or more communications and provides some services composing the capabilities offered by the agent's tasks (elementary agent behaviors). This structure can be regarded as the expected consequence of PASSI authors commitment in following the agent specifications provided by FIPA.

The Gaia agent is defined as a composition of roles. The specification of roles requires identifying the activities for which the role is responsible, including those activities that may require interactions with other agents, as well as the internal responsibilities of an agent. Once the abstract concept of role is translated into an actual agent, activities and responsibilities are translated into a set of services and a set of pragmatic activation and de-activation rules.

Goal and plan are other elements that should be considered in discussing the agent structure. None of the considered methodologies decidedly deal with them that are, conversely, central in other approaches (for instance goals are at the base of requirements analysis in the Tropos [14] methodology). In ADELFE, the notion of goal is only used to determine skills, but is not defined in a formal context. In the same way, plans are not modeled because usually, in complex and open applications, designers do not know plans. A plan will be built at runtime by the global system. However, if designers do know a plan, they can manage it by defining appropriate aptitudes. In Gaia, the concept of "goal" is implicit in roles, because a role in an organization (and thus the agent in charge of playing such a role) is by definition identified to achieve some specific application sub-goals. Plans play no explicit role in Gaia, although one can somehow consider that the activities of a role may include some sort of planning activities. In PASSI, goals are considered as non functional requirements and they are attached to agents according to their duties. As an example we can consider response or computational time constraints for agents operating in real-time contexts like robotics. They are usually described in the requirements analysis documentation in form of text. As regards agents' plans, they are not seen as a structural element of the PASSI meta-model, and they are usually modeled in a near algorithmic form (activity diagrams used as flow charts) during the Task Specification phase.

5.2 Agent Interaction Capabilities

In almost all the agent-based approaches, agents can interact with other agents or with the physical environment. About that, ADELFE, Gaia and PASSI are quite similar because in all of them agents are supposed to interact with others using communications ruled by some kind of interaction protocol (AIP) that could also ensure some level of interoperability among agents designed with different methodologies if they are all FIPA-compliant.

The most complete approach comes from ADELFE in which an agent can interact with other agents through direct communications but also in an indirect manner using the environment. An agent can perceive its environment and operate on it with its actions. Furthermore, ontologies have not to be modeled in ADELFE because if agents have to adapt themselves to their environment

they are also able to adapt to the other agents. This adaptation can lead agents to learn to understand each other. For instance, if an agent does not understand a request made by another one, the former has to detect a NCS and solve it. May be it will be able to learn what the other wanted to say or it will find another manner to help it (e.g., by relaxing the request to another judged relevant agent).

In PASSI, agent perceptions (obtained by sensing the environment of by communicating with other agents) are not directly represented but they are shown in form of the knowledge that the agent acquires from them. Communications are designed as the composition of several messages according to the interaction rules defined by an AIP (Agent Interaction Protocol). Each message is purposeful since it expresses the precise intention specified by its communicative act (speech act theory [13]). In PASSI, communicating is a privilege of a role and therefore it significantly concurs in defining the PASSI concept of role as a communicational role.

In Gaia, communications are related to both AIP and mediated interactions via the environment. With regard to AIP, Gaia does not enter in details about ontologies and specific types of ACL messaging schemes: while Gaia developers' consider these as necessary concepts, they consider them as not very influential in the analysis and design processes. With regard to communications mediated by the environment, these are considered as a sort of side effect - due to the fact that different agents may influence and perceive overlapping portions of an environment. However, such an issue has never been analyzed in depth in Gaia.

5.3 Agent Society and Organizational Structure

Societies modeled in ADELFE are open. The society exists only by the representation an agent possesses about other agents and these representations may change at runtime. As a consequence the organization between agents is not predefined and fixed when the system starts and even less at the design stage. This organization emerges from the evolving interactions between agents and makes the system adapt. ADELFE agents have to obey cooperation rules at the (local) micro-level, to possibly change their relationships with others in order to ensure that the collective behavior is coherent at the macro-level. A large part of the ADELFE MAS meta-model is then devoted to model all the factors of that social attitude but not the society that the agents could form.

Gaia agent is particularly devoted to the creation of societal organizations, and recognizes organizations as a primary abstractions to be exploited in MAS analysis and design. For these reasons, Gaia considers a MAS organization more than a collection of agents somewhat interacting. Rather, Gaia considers an organization as an entity having a well-defined structure (the organizational "architecture") characterizing the position of each agent (better, of the agents playing specific roles) in it, as well as a set of "organizational rules". Organizational rules make explicit the fact that an organization as a whole cannot be simply assumed to work well because of the well-defined behavior of its individual components. Rather, supra-role and supra-agent specifications are required,

expressing constraints on the inter-related activities of agents. Shifting to a societal metaphor, one can consider organizational rules as the social laws that have to drive all interactions in the organization and the evolution of the organization itself.

The PASSI model represents society aspects by defining services that can be provided/accessed by agents (specifically by some of their roles) and their participation in scenarios where they are supposed to interact via the already discussed communications. An agent is also supposed to have the availability of some resources that are explicitly modeled in order to identify its relevance for the remaining part of the society.

5.4 Agent Implementation

Even if the graphical modelling tool used within the ADELFE methodology (OpenTool) generates code skeletons, the problem of the system implementation is not treated yet and no platform is imposed.

Gaia totally abstracts from implementation tools. The key point is that - in the Gaia developers' intentions - the Gaia design specifications should be abstract enough that they could be used as guidelines to implement agents independently of the specific technology adopted.

In PASSI, a direct map exists among the most important elements of the model and their implementation; this is largely supported by a dedicated design tool (PTK, PASSI ToolKit) and the pattern reuse approach that is widely applied in the PASSI methodology. Each agent is coded using the base agent class of the selected implementation (FIPA-compliant) platform and it contains the tasks that are used by roles. A role has not a direct code level implementation since it is seen as an agent society domain element with only a virtual (not tangible) presence in the code. The service is described in a form that is suitable to be introduced in the deployment platform service directory in order to enable agents' collaborations.

6 Toward a Unifying Meta-model

After having analyzed the different MAS meta-models of ADELFE, Gaia and PASSI, we think that each of them has some very interesting features, but these are mainly located in different contexts (as discussed in section 5). This consideration brought us to design a new MAS meta-model that, including the most interesting aspects of each of the studied ones, could result in some kind of improvement to the state-of-the-art in this topic.

This model is presented in Figure 4 and we can see that it is quite a huge model. The fundamental choice that justifies it, is that we aim to create societies without (ADELFE) or with predefined organisations, 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 we enriched the generic agent with all the properties an agent may have, being cooperative or not. Fur-

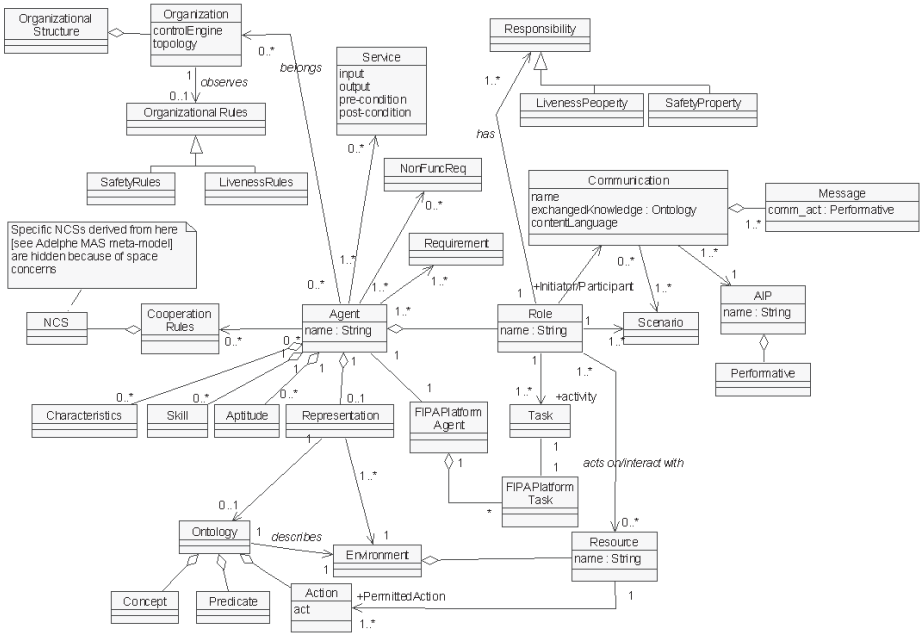

Fig. 4. A Unifying MAS Meta-Model - The new MAS meta-model is composed by merging the most significant contributions of ADELFE, Gaia and PASSI

thermore, 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 organisation or following cooperation rules (due to some lack of space in the figure above, inherited NCS such as incomprehension, uselessness have not been explicitly mentioned, see Figure 1). Agent 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 our 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.

After identifying this extensive MAS meta-model the following natural step would be to define a methodology for designing systems according to it. Although we will move in this direction, we fear that probably such a great model could need a design methodology that is composed of too many activities to be really profitable. It is presumable that while several different methodologies could cover different parts of this model (e.g., some will produce cooperative agents while

some others non cooperative ones), the presented model could be regarded as a unifying framework for the systems produced with different approaches thus enabling their interaction and providing a substantial step in the direction of a unique omni-comprehensive MAS meta-model.

7 Conclusion

A great number of agent-oriented methodologies exist nowadays; some are dealing with specific kinds of agents or multi-agent systems, like, for instance, the three ones that are depicted in this paper. ADELFE is devoted to cooperative agents and adaptive MAS, while Gaia aims more at creating social organisations and PASSI, the more general one, considers the whole life-cycle from the problem domain to the agent-based solution and the final level code implementation but limits the scope to FIPA-compliant systems. These differences are reflected by the meta-models elaborated by respecting authors to express the concepts used in the design activities and the resulting systems related to these three methodologies.

In this paper, these meta-models have been compared in order to begin a unification work that would be beneficial to the agent-oriented engineering domain. It has then appeared that all of the three models share common concepts such as the agent and interaction protocols ones while other elements are present only in some of them: this is the case of ADELFE and Gaia that share the communication and environment notions, and Gaia and PASSI that have notions like roles and services in common. Some concepts are only appearing in one of the three meta-models, for instance, responsibilities in Gaia, ontology in PASSI or representations (of others) in ADELFE. Putting these different meta-models together has enabled enriching them mutually as well as unifying the different used concepts. This preliminary unification has led methodologies authors to revise their respective meta-models to make choices and concessions to present the merged meta-model in Figure 4. Furthermore, we are sure that this unification would be useful to build tools in the OMG's MDA [15] spirit in order to automatically transform a meta-model into a model depending on a target platform.

This unification problem leads us to some interesting questioning that could represent (our) future works:

- Is it possible to identify a meta-model from which all the meta-models used in the multi-agent community could be derived? For instance, this latter could be defined from an extension of this unification work as well as FIPA Modelling TC standardisation activities.
- What description level has to be reached in the meta-model? For instance, skills and aptitudes in ADELFE are certainly used to implement the role notion of Gaia or PASSI.
- How may a designer choose meta-model elements he is interested in? What kind of tools can we provide him to ease his choices?

References

1. Saeki, M.: Software specification & design methods and method engineering. International Journal of Software Engineering and Knowledge Engineering (1994)
2. Kruchten, P.: The Rational Unified Process: An Introduction. Addison-Wesley (2000)
3. Bergenti, F., Gleizes, M.P., Zambonelli, F.: Methodologies and Software Engineering for Agent Systems. Kluwer (2004)
4. Bernon, C., Camps, V., Gleizes, M.P., Picard, G.: Tools for self-organizing applications engineering. In: First International Workshop on Engineering Self-Organising Applications (ESOA), Melbourne, Australia (2003)
5. Capera, D., Georg, J.P., Gleizes, M.P., Glize, P.: The amas theory for complex problem solving based on self-organizing cooperative agents. In: Proc. of the 1st International Workshop on Theory And Practice of Open Computational Systems (TAPOCS03@WETICE 2003), Linz, Austria (2003)
6. Wooldridge, M., Jennings, N.R., Kinny, D.: The gaia methodology for agent-oriented analysis and design. Journal of Autonomous Agents and Multi-Agent Systems **3** (2000) 285–315
7. Zambonelli, F., Jennings, N., Wooldridge, M.: Developing multiagent systems: the gaia methodology. ACM Transactions on Software Engineering and Methodology **12** (2003) 417–470
8. Cossentino, M., Sabatucci, L.: Agent System Implementation. In: Agent-Based Manufacturing and Control Systems: New Agile Manufacturing Solutions for Achieving Peak Performance. CRC Press (2004)
9. Chandrasekaran, B., Josephson, J.R., Benjamins, V.R.: What are ontologies, and why do we need them? IEEE Intelligent Systems (1999)
10. FIPA: Acl message structure specification. (Available online at http://www.fipa.org/specs/fipa00061/SC00061G.html)
11. FIPA: Rdf content language specification. (Available online at http://www.fipa.org/specs/ fipa00011/XC00011B.html)
12. W3C: Resource description framework (rdf) model and syntax specification. w3c recommendation. Available online at http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/. (1999)
13. Searle, J.R.: Speech Acts. Cambridge University Press (1969)
14. Castro, J., Kolp, M., Mylopoulos, J.: Towards requirements-driven information systems engineering: The tropos project. Information Systems (2002)
15. Kleppe, A., Warmer, J., Bast, W.: MDA Explained: The Model Driven Architecture : Practice and Promise. Addison-Wesley (2003)