AOSE Technical Forum Group

AL3-TF1 Report

30 June- 2 July 2004, Rome

1 Introduction

The AOSE TFG activity in Rome was divided in two different sessions, both of them scheduled for Friday, (2nd July): the first session has been held in the morning and dealt with methodologies and MAS (multi-agent system) meta-models. In the second session (in the afternoon), we discussed about methodologies evaluation, ideas for future activity of this TFG and future trends in AOSE in general have been discussed too.

More than forty people attended the two different sessions of the AOSE meeting and 17 talks have been presented.

In the following the discussed topics, presented talks and debates outcomes will be presented in their chronological order.

2 Methodologies

More than twenty different design methodologies can be found in literature for the design of MASs (Multi-Agent Systems). This can be seen as a proof that agent designers in accomplishing their different tasks, and solving specific problems in distinct production environments, often prefer to setup an own methodology specifically tailored for their needs instead of reusing existing ones. What seems to be widely accepted is that an unique specific methodology cannot be general enough to be useful to everyone without some level of personalization.

In this scenario we can identify two contrasting elements: first, in an interesting paper on object-oriented software (development) processes, Fuggetta (in “Software Process: a Roadmap”, 2000) states that the research in this field is stuck and most technologies developed by the software process community have not been adopted by the industrial world; second, the AgentLink community, in its roadmap for agent based computing, embodying the feelings of a large part of the agent research and industrial community, has identified the designation of a standard in design methodologies as an essential demand.

In order to accomplish this request, during this first meeting of the AOSE TFG, we focused our attention to the exploration of existing methodologies and the study of possible unification strategies.

2.1 Methodologies overview

Several methodologies (eight) have been presented during the meeting; they gave an idea of the relevant effort spent by the European agent community on this topic.

2.1.1 IDE-eli

The Integrated Development Environment for Electronic Institutions (IDE-eli) is a set of tools aimed at supporting the engineering of multi-agent systems as electronic institutions.

Software agents appear as the key enabler technology behind the electronic institutions vision. Thus, electronic institutions encapsulate the coordination mechanisms that mediate the interactions among software agents representing different parties. IDE-eli allows for engineering both electronic institutions and their participating software agents. Notably, IDE-eli moves away from machine-oriented views of programming toward organizational-inspired concepts that more closely reflect the way in which we may understand distributed applications. It supports a top-down engineering approach: firstly the organization, secondly the individuals. IDE-eli is composed of:

- **ISLANDER.** A graphical tool that supports the specification of the rules and protocols in an electronic institution.
- **AMELI.** Software platform to run electronic institutions. Once an electronic institution is specified with ISLANDER, it is ready to be run by AMELI without any other programming effort.
- **aBUILDER.** Agent development tool.
- **SIMDEI.** Simulation tool to animate and analyse specifications created with ISLANDER prior to the deployment stage.

Many more application areas may be tackled with the aid of the IDE-eli tools. In general terms, the electronic institutions approach is deemed as appropriate in complex domains where multiple partners are involved, and a high degree of coordination and collaboration is required. Thus, electronic institutions look promising to support workflow management, monitoring and management of shop-floor automation, or supply network management issues.

2.1.2 INGENIAS

The purpose of INGENIAS is the definition of a methodology for the development of MAS, by integrating results from research in the area of agent technology with a well-established software development process, which in this case is the Unified Software Development Process (USDP). This methodology is based on the definition of a set of meta-models that describe the elements that form a MAS from several viewpoints, and that allow to define a specification language for MAS. The viewpoints are five: agent (definition, control and management of agent mental state), interactions, organization, environment, and goals/tasks.

2.1.3 META-DIMA

MetaDIMA aims at bridging the gap between existing agent architectures with their development tools and agent-based methodologies. The idea is to start from existing architectures and tools and to elaborate meta-models in order to formulate the necessary knowledge about the development process.

MetaDIMA is inspired by the Model-Driven Architecture (MDA), proposed by OMG, which aims at separating application logic from the underlying technologies to improve reusability and development process.

The aim of this project is to illustrate the use of MDA to define a new multi- agent system development process that is based on a library of meta-models. These meta-models may be used by the designer to easily describe the multi-agent models which are automatically transformed in order to generate a runnable multi- agent system.

2.1.4 Agile PASSI

In Agile PASSI, the primary requirement is related to not distracting developers from their main goal (producing a system that solves some kind of algorithmic problem, for instance in robotics) with a long design process.

This could be achieved by using an agile process that supports a light (manual) design phase while it encourages the reuse of existing contributions in form of patterns; supporting design tools automatically produce a consistent documentation at different level of abstractions. Agile PASSI takes advantage of the work done with PASSI and reuses a couple of its most important features: (i) the identification of agents as a set of functionalities expressed in form of use cases, and (ii) the central role of ontology description in analyzing the agent solution.

2.1.5 RIO

The RIO meta-model is centred on two concepts: role and organization. A role is an abstraction of a behaviour or a status in an organisation, an organization is a set of roles where each role interacts with at least one another in the organization.

In RIO, formal specifications (obtained with ObjectZ and statecharts) are linked to implementation and the designer work is supported by tools that allow the simulation of the system and its verification.

2.1.6 Adelfe

ADELFE aims to cover all the phases of a classical software design: from requirements to deployment. A well-know process, the RUP (Rational Unified Process), has been tailored to take into account specificities coming from the design of Adaptive Multi-Agent Systems. Only the requirements, analysis and design phases require modifications in order to be adapted to AMAS, all the others appearing in the RUP remain the same.

Three main tools are integrated into the ADELFE toolkit: an interactive tool which describes the process and helps the designer to apply it, OpenTool, a graphical modelling tool, and a tool which analyses if using the AMAS technology is useful to implement the target system.

2.1.7 MASE

The Multi-agent Systems Engineering (MaSE) is a general purpose methodology for developing multi-agent systems that is founded on basic software engineering principles. MaSE divides the development process into two major phases: the analysis phase and the design phase. Each phase is composed of a set of stages.

The development process adopted in MASe is iterative. This means that any stage can be repeated many times until the final design is achieved and at any stage a designer may go back to a previous stage.

MaSE is supported by a CASE tool (called agent Tool) that has the capability of performing the analysis and design activities, validating the various models, generating automatic transformation of models, and generating skeleton code. A specific class library is available, called agentMOM, that can be used for code generation.

MaSE also supports the specification of mobility aspects and the definition of ontology.

2.2 Unification proposals

In the unification direction two different talks have been proposed. In the first talk, A. Garro presented his approach on the composition of a new methodology starting from fragments of other existing methodologies. In the second talk, J. Pena presented a fragment from the MacMAS methodology dealing with the analysis of the acquaintance organization in a MAS. These contributions will now be presented more in details.

2.2.1 A Methods Integration Approach

This work refers to the FIPA Methodology Technical Commitee activity and it consists in a quite open approach that allows the composition of elements coming from a repository of fragments of existing design processes that could be expressed in terms of a standard notation.

Specifically dealing with the methods integration problem in this contribution two different approaches have been considered to obtain methods integration: (i) guided by a (MAS) meta-model; (ii) guided by a development process.

In the first approach (guided by a MAS meta-model), while building his own methodology, the designer has preliminary to identify the elements that compose the meta-model of the MAS he is going to build; then, he has to choose the method fragments that are able to produce the identified meta-model elements. The second approach (guided by a development process) focuses on the instantiation of some software development process that completely covers the development of MAS. Given a specific problem and/or an application domain, the process will be instantiated by selecting, for each phase, suitable method fragments, chosen from agent-oriented methodologies proposed in the literature or ad-hoc defined.

2.2.2 An Acquaintance Organization Fragment

The methodology fragment proposed by J. Pena consists in a systematic, iterative and incremental approach for the development of the acquaintance organisation (interaction organisation) of a MAS structuring models in a set of abstraction layers. It is specially tailored to deal with complexity derived from interactions, thus, we focus all the modelling process on them. This fragment is a part of MacMAS that is a fractal methodology where a model is refined by its bottom layer models and abstracted by its top layer model

3 MAS meta-models

In the object-oriented context the construction of a new methodology and the execution of the design process, rely on a common denominator, the universally accepted concept of object and related meta-model of the object-oriented system. The situation, in the agent-oriented context 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 system elements like agent, role, behaviour, ontology, etc. that compose the actual system together with their relationships).

In this case, one of the first steps of a new methodology composition process, should therefore consists in an accurate analysis of the MAS meta-models in order to identify which elements will compose the system that will be built. Its availability will not only be helpful in defining the whole methodology but it will help at a practical level too: the same fact of clearly declaring the structure of the system will enable the development of design tools that are able to check for design correctness and find parts that are not completely/coherently defined. In this view, the final result of the design activity should, obviously, consists in a model of the system - an instantiation of the MAS meta-model – that solves the faced problem.

The situation up to date is that the lack of a unique MAS meta-model leads each methodology to have its own concepts and system structure. Starting from the proposed contributions (four talks) during this meeting our group tried to establish a strategy that could bring to the identification of a common meta-model that could be widely adopted.

The contributions were about already adopted meta-models and an unification proposal. They will be presented in the following.

3.1 MAS Meta-models of Existing Methodologies

3.1.1 INGENIAS

The INGENIAS agent meta-model defines an isolated agent. The agent concept underlying this meta-model is the one defined by Newell in "The Knowledge level" (1982). An agent is therefore seen as a program that exists at the *knowledge level*. It has a physical body with which it can act in the environment, a knowledge body which contains whatever the agent knows at some time, and a set of goals. Also, an agent behaves according to the principle of rationality (*if an agent has knowledge that one of its actions will lead to one of its goals, then the agent will select that action)*. Following this definition, an agent has goals and there should be some association type between agent tasks and goals.

The agent has a mental state which is used to decide what to do next; this mental state is managed by the *MentalStateManager* that is in charge of adding/removing knowledge as well as consistence maintenance. Agent's mental state consists of control entities and information entities. Control entities specify what is expected from the agent, whilst information entities describe the state of the world as seen by the agent.

On the other side, the INGENIAS agent plays roles in several workflows in the system. The association of an agent to a role means that the agent acquires all the properties and responsibilities assigned to the role (goals and interactions in which the role participates).

Social aspects in this approach are abstracted in the organization concept. An *organization,* in the INGENIAS approach, characterizes a group of agents that work together towards a common goal (*purpose*). The organization may consist of only one agent or several groups of cooperating agents, which form part of *organizational structures* that establish relationships among them.

3.1.2 MacMAS

In the MacMAS approach interactions are the central concept of each methodology fragment, and of the whole MAS meta-model.

This has been done to improve the ability to deal with systems with a high interaction degree. An interaction can relate an arbitrary number of roles (multi-Role Interactions) and can be refined by several finer–grain interactions.

This allows performing a layered description of the system which improves the possibility of dealing with complexity through an iterative and incremental process.

MacMAS is completed by a set of automatic algorithms to compose and decompose interactions that also helps in the transition between layers.

Interactions are related to systems goals and this enhances the traceability between requirements and final models.

3.1.3 Software Agents vs. Software Components

This work is based on the hypothesis that given a component, we can ascribe mental qualities, i.e., knowledge and goals, to it; this brings to the consideration that components described in terms of their mental qualities are agents.

In this approach, the goal consists in enhancing the possibility of reusing significant parts of a MAS, this can be done by adding some constraints on software infrastructures to better support reusability. Particularly infrastructures should be a means for: (i) transferring knowledge and goals (mainly goals), and not only for moving data, (ii) support agents finding each other transparently, i.e., without an explicit request (from the agent or from the programmer), (iii) allow agents finding problem solvers and not just task executors, (iv) enable choosing a problem solver on the basis of what it can do and not on how it can describe its capabilities.

In other words some "intelligence" should be moved to the infrastructure but still considering that the problem should not be moved there, i.e. the system designer should not program the infrastructure.

3.2 Unification proposals

One contribution in this direction has been proposed by C. Bernon. Her talk started from the consideration that some of the agent-oriented methodologies that exist nowadays are dealing with specific kinds of agents or multi-agent systems; this is, for instance, the case of ADELFE, Gaia and PASSI. 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 it limits its scope to FIPA-compliant systems. These differences are reflected in the meta-models elaborated by respecting authors to express the concepts used in their design activities.

In the talk, these meta-models have been compared in order to begin a unification work. One of the most interesting result is the discovery that all of the three (meta-)models share common concepts such as agent and interaction protocol, 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, while Gaia and PASSI have in common notions like role and service. Some concepts are only appearing in one of the three meta-models, for instance, responsibilities in Gaia, ontology in PASSI or representation (of others) in ADELFE.

Putting these different meta-models together, comparing their similarities and differences has enabled enriching them mutually as well as unifying the different used concepts and the work resulted in the unified MAS meta-model that has been presented at the end of the talk.

4 Methodologies evaluation (guidelines and techniques)

There were two presentations concerning the evaluation of methodologies. One more generic, from A. Sturm ("Evaluation Techniques for Agent-Oriented Methodologies") and the other referring to a concrete aspect, which is the way methodologies deal with complexity, specially for the modelling of interactions in MAS (J. Peña and R. Corchuelo, "Guidelines, Techniques and Modelling Artefacts at the Analysis Stage of AOSE Methodologies to Deal with Complexity").

4.1.1 Dealing with Complexity at the Analysis Stage

The first presentation in this section proposed a RFC (Request for Comments) about guidelines, techniques and modelling artefacts at the analysis stage of AOSE methodologies to deal with complexity. The complexity here is related to the interactions between agents and it is closely linked with the organizational aspects. The proposed approach is specially tailored to deal with complicated organizations and is based on three principles (initially identified by N. Jennings):

- Abstraction that defines simplified models of the system emphasising some details while avoiding others in order to focus the designer's attention,
- Decomposition that applies the "divide and conquer" principle to limit the designer's scope to a portion of the problem,

- Composition that identifies and manages inter-relationships between the different sub-systems. It is then possible to group together various basic components to treat them as higher-level units of analysis.

These principles have to be taken into account at the modelling process level which is not the case in most of the existing methodologies. In order to do that, a set of features that should be provided by AOSE methodologies at the analysis stage is proposed. These features are divided into:

- Techniques that are procedures to transform models,
- Modelling artefacts that are icons to graphically represent the system,
- Guidelines that indicate the best practices to deal with complexity by using the previous techniques and modelling artefacts.
- The RFC presents the three kinds of principles according to static and dynamic aspects:
- The acquaintance aspect models the relationships between agents in the system from the interaction point of view.
- The behaviour aspect models the order of apparition of these relationships over time.

To evaluate an analysis stage, this RFC can be found on-line via a form in which the different features are asked and can be weighted according to an agreement level (which ranges from 0 to 100) and the ability to conquer complexity (with the same range).

4.1.2 Evaluation of Agent Oriented Methodologies

The second presentation of this session was based on the fact that some comparisons between different methodologies (according to different criteria depending on the goal of the work) can be found in literature but no guideline or tool really exist to address advantages or drawbacks of methodologies or even to compare them.

Considering that a methodology is an entire set of guidelines and activities (lifecycle process, concepts and models…), the presented evaluation framework was based on four categories:

- Concepts and properties. Since no consensus exists within the agent community yet, the basic concepts related to agents and MASs were determined (autonomy, reactiveness, proactiveness and sociality) and then used as properties to evaluate whether an agent-oriented methodology is close to these concepts. Another evaluation determines if a methodology supports building blocks (e.g. agent, belief, desire, intention, message, role, task…), the associated notations and to what extent.
- Notations and modelling techniques. Some properties (e.g. accessibility, analyzability, modularity…) were considered to evaluate how a methodology deals with symbols used to represent elements within a system and how it deals with models depicting aspects of a system at different levels of abstraction.
- Process. The development process is evaluated according to the development context and the lifecycle coverage (from requirements to implementation and tests). The evaluation should also consider whether this methodology provides a detailed description of the various activities of the development lifecycle in order to ease its use.
- Pragmatics. The evaluation of a methodology must take into account how this methodology deals with practical aspects of deploying and using it in a real project or

organization. In this case, issues such as available resources, required expertise, language suitability, domain applicability and scalability have to be considered.

Finally, based on academic experiments, some results were given concerning the evaluation of four selected methodologies: Gaia, MaSE, Tropos and OPM/MAS.

Taking into consideration the goals of the AOSE TFG, it seems that evaluation is one of the first tasks we should address. Different techniques are being proposed: feature-based comparison, meta-modelling, metrics, ontological evaluation, and experimentation with case studies. How to organize a common activity in the AOSE TFG towards evaluation stays as an open issue in this meeting, but it is recognized as one of the issues to address in the next ones.

5 Open directions in AOSE

In the last few years, together with the increasing acceptance of agent-based computing as a novel software engineering paradigm, there has been a great deal of research related to the identification and definition of suitable models and techniques to support the development of complex software systems in terms of multi-agent systems. These researches, which can be roughly grouped under the term "agent-oriented software engineering", are endlessly proposing a variety of new metaphors, formal modelling approaches, development methodologies and modelling techniques, specifically suited to the agent-oriented paradigm.

However, the research in the area of agent-oriented software engineering is still in its early stages, and several challenges need to be faced before agent-oriented software engineering can fulfil its promises, becoming a widely accepted and a practically usable paradigm for the development of complex software systems. One possible way to identify and frame the key research challenges in the area of agent-oriented software engineering is to recognize that such challenges may be very different depending on the "scale of observation" adopted to model and build a software system.

At one extreme, the micro scale of observation is that where the system to be engineered has to rely on the controllable and predictable behaviour of (a typically limited number of) individual agents, as well as on their mutual interactions. There, the key engineering challenges are related to extending traditional software engineering approaches toward agent-oriented abstractions. Although the use of AUML and of a traditional "waterfall" process model – as promoted by current researches – may have been of some use in the first years of research, the current understanding is that brand new modelling and notational tools, as well as possibly brand new software process models may be needed, to exploit in full the potentials of the agents paradigm.

At the other extreme, the macro scale of observation is the one where a multi-agent system is conceived as a multitude of interacting agents, for which the overall behaviour of the system, rather than the mere behaviour of individuals, is the key of interest. In this case, a discipline of agent-oriented software engineering should focus on totally different problems, and should be able to develop novel "systemic" approaches to software engineering, possibly getting inspiration from areas such as complex systems sciences and systemic biology.

In between, the meso scale of observation is that where the need of predictability and control typical of the micro scale clashes with the emergence of phenomena typical of the macro scale. Therefore, any engineering approach at the meso scale requires accounting for problems that are typical of both the micro and the macro scale, and possibly for new problems specific to the meso scale. These include: identifying the boundaries of a systems – which may be challenging in the case of open multi-agent systems; electing trust as a primary design issue; identifying suitable infrastructures for multi-agent systems support.

6 Rome Discussions outcomes

In this first meeting of the AOSE group, large time has been spent in talks that enabled the presentation of several research activities and points of view of the members. This has been done to allow the identification of the common themes that could guide the continuation of this group activity. At the end of each talk, we also had some time for comments and/or short debates that enriched the value of the presentation and introduced new topics. In these discussions, a diffused agreement has been reached by the members about the need for the institution of a high-level AOSE conference. The interest that recent AOIS, AOSE, ESAW, SELMAS and similar workshops have obtained, and the growing community in the area, in fact, lead us to consider that the "upgrade" of one of these workshops to the conference level, could originate in a conference that is able to live independently and complementary to other major events.

Another interesting discussion dealt with the identification of a common activity that could be done offline in view of the next AL3 event; many of us accepted to work on the compilation of one or more MAS meta-models that could be used as reference points by the whole community. Although this work could present problems, all of us agree that achieving concrete results in this area, would be very useful for several reasons: (i) this partly solve the lack of standardization in this area highlighted by 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. In this activity we also decided to maintain a strong relation with the similar work that is in progress within the FIPA community.

At the end of the day we also discussed what should be included in the next meeting agenda. While some space should be obviously left to the discussion of the results of the offline activity presented above, other topics could range from modelling (and to some extent, implementation) that obviously remains among the main activities in AOSE and formal methods that are gaining relevance, especially for verification and validation activities (which in fact are not very much developed in the agent field still to date).