ROMAS methodology overview

Emilia Garcia
mgarcia@dsic.upv.es

December 10, 2012

1 Introduction

This document gives an overview of this methodology by means of: (1) A description of the ROMAS methodology's objectives; (2) An introduction to the main concepts of the ROMAS architecture and metamodel; (3) An introduction to the ROMAS process lifecycle; (4) A brief description of the ROMAS background. Further information can be consulted in the bibliography [5, 6, 7, 4].

1.1 ROMAS objectives

ROMAS methodology tries to deal with some of the open issues on the analysis and design of normative open MAS. Specifically, ROMAS tries to contribute to the state of the art by offering a complete development process for analyzing and designing normative open MAS that includes a set of guidelines to identify, formalize and verify the normative context of the system, as well as, that allows the traceability of the normative context from the requirements to the design decisions and viceversa.

The general objectives of ROMAS are:

- Analyzing the system requirements from a global and individual point of view, i.e., analyzing the global requirements of the system and the individual requirements of every entity of the system.
- Analyzing and formalizing the social structure of the system and the relationships between its entities.
- Formalizing the relationships and interchanges between entities in a way that allows heterogeneous and autonomous entities to interact, even if these entities have been implemented by external providers using different technologies.
- Analyzing and formalizing the normative context of the system, i.e., the restrictions on the entities behavior derived from the system's requirements and the design decisions.
- Verifying the coherence of the designed normative context.
- Formalizing the normative context in a way that allows the traceability from the requirements to the design decisions and viceversa.

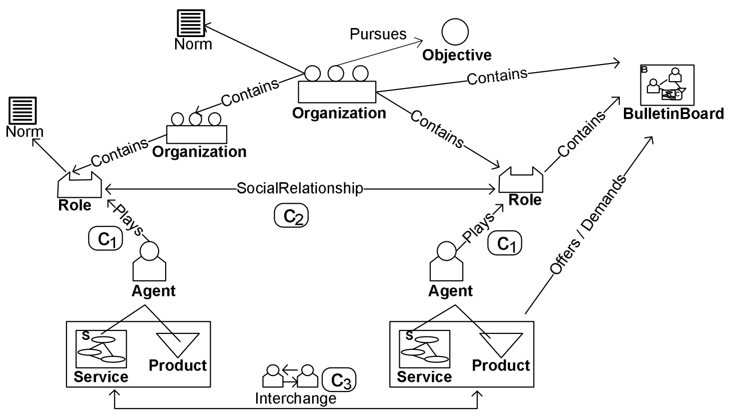

Figure 1: Overview of ROMAS architecture

1.2 ROMAS architecture and metamodel

In ROMAS, *agents*, *roles* and *organizations* are defined through a formal social structure based on a service-oriented open MAS architecture, whose main features are summarized in Figure 1. Here, organizations represent a set of individuals and institutions that need to coordinate resources and services across institutional boundaries. In this context, agents represent individual parties who take on roles in the system, within a given organization (e.g. a company), they can both offer and consume services as part of the roles they play. Beyond this, virtual organizations can also be built to coordinate resources and services across institutional boundaries. Importantly, each of these concepts must be strictly defined, alongside their interrelations. Organizations are conceived as an effective mechanism for imposing not only structural restrictions on their relationships, but also normative restrictions on their behavior. These restrictions are formalized in ROMAS by means of norms and contracts.

Norms in ROMAS are specified using the model described in [2], which defines norms that control agent behaviour, the formation of groups of agents, the global goals pursued by these groups and the relationships between entities and their environment. Specifically, it allows norms to be defined: (i) at different social levels (e.g. interaction and institutional levels); (ii) with different norm types (e.g. constitutive, regulative and procedural); (iii) in a structured manner; and (iv) dynamically, including later derogation. Figure 1 shows two types of norms: (i) those that are associated with each organization; and (ii) those that are associated with each role. Clearly, the former must be complied with by any organization member, while the latter must be complied with by all agents playing that role.

Finally, ROMAS also allows interactions to be formalized by means of *contracts*. These are necessary when working in an open regulated system, to be able to specify the expected behavior of others without compromising their specific implementation. ROMAS involves two types of contracts: *social contracts* and *contractual agreements*. Social contracts can be defined as a statement of intent that regulates behaviour among organizations and individuals. As shown in Figure 1, social contracts are used to formalize relationships: (i) between an agent playing a role and its host organization (as indicated by the contract labelled c_1); and (ii) between two agents providing and consuming services (as indicated by c_2). Social order, thus, emerges from the negotiation of contracts about the rights and duties of participants, rather than being given in advance. In contrast, contractual agreements represent the commitments between several entities in order to formalize an interchange of services or products (c_3).

The properties of each entity of the presented architecture and the allowed relationships between them are formalized in the **ROMAS metamodel**.

In order to facilitate the modeling tasks, this unified metamodel can be instantiated by means of four different views that analyze the model from different perspectives:

- The *organizational view* that allows specifying the system from a high-level of abstraction point of view. This view allows specifying the global purposes of the system, the relationships with its environment, the division of the functionality of the system in roles and the main structure of the system.

- The *internal view* that allows specifying each entity (organizations, agents and roles) of the system in high and low level of abstraction point of view. From a high-level of abstraction, this view allows specifying the believes and objectives of each entity, and how the entity participate in the system and interact with its environment. From a low-level of abstraction, this view allows specifying the internal functionality of each entity by means of the specification of which task and service implements. One instance of this view of the metamodel is created for each entity of the system.

- The *contractTemplate view* that allows specifying *contract templates* which are predefined restrictions that all final contract of a specific type must fulfill. Contracts are inherently defined at runtime, but contract templates are defined at design time and can be used at runtime as an initial point for the negotiation of contracts and to verify if the final contract is coherent with the legal context.

- The *activity view* that allows specifying interaction protocols, the sequence of activities in which a task or a service implementation is decomposed.

1.3 ROMAS process lifecycle

ROMAS tries to guide developers during the analysis and design phases in a intuitive and natural way. In that sense, ROMAS derives the whole design from the analysis of the requirements and their formalization by means of objectives. Following a goal-oriented approach, developers are focused from the early beginning in the purpose of the system.

ROMAS development process is composed of five phases, which help developers to analyze and design the system from the highest level of abstraction to the definition of individual entities and implementation details (Figure 2). Following a summary of the purposes and results of each phase is presented:

- *Phase 1. System specification:* The purpose of this phase is to analyze the system requirements from a global point of view, i.e., focusing on what

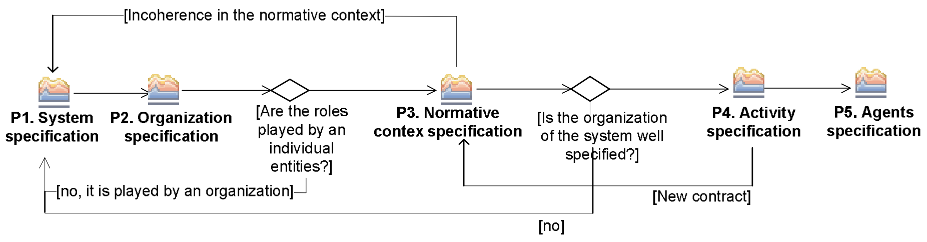
[Incoherence in the normative context]
P1. System specification
P2. Organization specification
[Are the roles played by an individual entities?]
P3. Normative contex specification
[Is the organization of the system well specified?]
P4. Activity specification
P5. Agents specification
[no, it is played by an organization]
[New contract]
[no]

Figure 2: The ROMAS process phases

it is important for the system and as whole instead of focusing on the individual interests of each entity. These requirements are translated in terms of objectives and restrictions. The global objectives of the system are studied and refined into operational objectives and the main use cases of the system are specified. Once all the requirements of the system have been analyzed, the last task of this phase is to evaluate the suitability of the ROMAS methodology for the development of the system regarding its specific requirements.

The results of this phase of the methodology are: (1) a textual description of the system requirements, (2) a textual description of the objectives of the system, (3) a objective decomposition diagram, (4) a set of the use cases diagrams, (5) an study of the suitability of the ROMAS methodology for this system.

- *Phase 2. Organization specification:* The purpose of this phase is to analyze and design the social structure of the system. First, the functionality of the system is associated to roles. Then, the relationships between this roles, the restrictions and the social environment of the system are analyzed in order to select the most suitable social architecture. This social architecture specifies in a high-level of abstraction which are the social relationships between the roles of the system (like authority or collaboration) and if the system is composed of several organizations.

  The results of this phase of the methodology are: (1) a textual description of the roles of the system, (2) one diagram for each role of the system specifying its properties. These diagrams are instances of the Internal view of the metamodel, (3) one diagram for representing the social environment and structure of the system. This diagram is an instance of the Organizational view of the metamodel.

- *Phase 3. Normative context specification:* The purpose of this phase is to formally specify the normative context of the system by means of norms and contracts. The requirements of the system, the normative documents associated to the system (like governmental legislation or institutional regulations) and the social structure of the system are analyzed in order to identify the norms and contracts that should be formalized. The processes of identification, formalization and validation of the normative context are supported by a set of guidelines.

  The results of this phase are: (1) modifications on the diagrams defined in the previous phase in order to add the norms and contracts identified,

(2) a set of diagrams for specifying the contract templates of all the identified social relationships. These diagrams are instances of the Contract template view of the metamodel.

- *Phase 4. Activity specification:* The purpose of this phase is to specify the tasks, services and protocols that have been identified in the previous phases of the development process. In that sense, this phase revises the role internal view diagrams, the organizational view diagram and the contract template view diagrams in order to identify which tasks, services and protocols should be detailed. For example, for each contract template a negotiation and an execution protocol should be specified.

  The results of this phase are a set of diagrams, one for each task, service and protocol, that are instances of the Activity view of the metamodel.

- *Phase 5. Agents specification:* The purpose of this phase is to analyze and design every individual entity of the system. This phase analyzes the requirements of each entity, its restrictions and which roles this entity should play in order to achieve its objectives. The last step of this phase is to validate the coherence between the design of every individual entity and the global design of the system.

  The results of this phase are a set of diagrams, one for each individual entity, that are instances of the Internal view of the metamodel and that specifies the features, properties and interactions of this entity with the rest of the system.

As the Figure 2 shows, this is not a linear process but an iterative one, in which the identification of a new element of functionality implies the revision of all the diagrams of the model and the work products produced, so it requires to go back to the appropriate phase. For example, during the second phase (Organization specification), part of the detected roles can be played by a group of agents that form another organization. In this case, it is necessary to go back to the first phase of the methodology to analyze the characteristics, global objectives and structure of this organization.

1.4 ROMAS background

Although the ROMAS background is quite extensive, there are two methodologies that influence the most to ROMAS: GORMAS [1] and OperA [3].

ROMAS uses the GORMAS metamodel as a starting point for the specification of its own metamodel. GORMAS is a service-oriented methodology that defines a set of activities for the analysis and design of organizational systems, including the design of the norms that restrict the behavior of the entities of the system. ROMAS metamodel inherits from GORMAS the concepts of agents, organizations, services and norms. ROMAS revises the GORMAS metamodel in order to refine these concepts. ROMAS also adds the concept of social and commercial contract. The process lifecycle of ROMAS and GORMAS are completely different. GORMAS bases the development process in the specification of the services that every entity must provide and use, while ROMAS bases it in the objectives of the system and the objectives of each entity of the system.

The graphical notation used in ROMAS to formalize the models is based on the notation used in GORMAS [1], ANEMONA [8] and INGENIAS [9]. ROMAS adds few graphical icons to represent some elements, like contract templates, that were not previously defined in these methodologies.

The concept of social contract used in ROMAS is similar to the concept of contract in the Opera methodology. However, ROMAS does not share the same concept of organizations and interactions. Organizations in OperA are defined as institutions where agents interact between them entering in previously determined scenes. Moreover, other differences are that OperA does not include the analysis and design of individual agents and it does not offer specific guidelines for identify the norms derived from the analysis of the requirements, legal documents or design decisions.

References

[1] E. Argente. *GORMAS: Guías para el desarrollo de Sistemas Multiagente abiertos basados en organizaciones.* PhD thesis, Departament de Sistemes Informàtics i Computació, Universitat Politècnica de València, 2008.

[2] N. Criado, E. Argente, and V. Botti. A normative model for open agent organizations. In *International Conference on Artificial Intelligence*, volume 1, pages 101–107, 2009.

[3] V. Dignum. *A model for organizational interaction:based on agents, founded in logic.* PhD thesis, Utrecht University, 2003.

[4] E. Garcia, A. Giret, and V. Botti. A Model-Driven CASE tool for Developing and Verifying Regulated Open MAS. *Science of Computer Programming*, page In Press, 2011.

[5] E. Garcia, A. Giret, and V. Botti. Regulated open multi-agent systems based on contracts. In *Information Systems Development*, pages 243–255, 2011.

[6] E. Garcia, A. Giret, and V. Botti. Developing Regulated Open Multi-agent Systems. In *International Conference on Agreement Technologies*, pages 12–26, 2012.

[7] E. Garcia, G. Tyson, S. Miles, M. Luck, A. Taweel, T. V. Staa, and B. Delaney. An Analysis of Agent-Oriented Engineering of e-Health Systems. In *13th International Workshop on Agent-Oriented Software Engineering (AOSE - AAMAS)*, pages 117–128, 2012.

[8] A. S. Giret. *ANEMONA: Una Metodologia Multiagente para Sistemas Holonicos de Fabricacion.* PhD thesis, Departamento de Sistemas Informaticos y Computacion, Universidad Politecnica de Valencia, 2005.

[9] J. Pavon, J. Gomez-Sanz, and R. Fuentes. The ingenias methodology and tools. In *Agent-Oriented Methodologies*, volume chapter IX, pages 236–276. Henderson-Sellers, 2005.