ROMAS metamodel manual

Emilia Garcia
mgarcia@dsic.upv.es

December 19, 2012

1 Introduction

This document details the ROMAS metamodel elements, relationships and structure. 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.

A complete description of these views of the metamodel is presented in Section 1.1. Section 1.2 shows the graphical notation used to draw the ROMAS models.

Table 1 describes the entities that the ROMAS metamodel uses for modeling normative open MAS.

Concept	Definition	Metamodel views
Objective	An objective is a specific goal that agents or roles have to fulfill. It can be refined into other objectives.	Organizational Internal view
Organizational Unit (OU)	A set of agents that carry out some specific and differentiated activities or tasks by following a predefined pattern of cooperation and communication. An OU is formed by different entities along its life cycle which can be both single agents or other organizational units, viewed as a single entity.	Organizational Internal Contract template
Role	An entity representing part of the functionality of the system. Any entity that plays a role within an organization acquires a set of rights and duties.	Organizational Internal Contract template Activity
Agent	An entity capable of perceiving and acting into an environment, communicating with other agents, providing and requesting services/resources and playing several roles.	Organizational Internal Contract template Activity
Norm	A restriction on the behavior of one or more entities.	Organizational Internal Contract template Activity
Contract template	A set of predefined features and restrictions that all final contract of a specific type must fulfill. A contract represent a set of rights and duties that are accepted by the parties.	Organizational Internal Contract template Activity
Bulletin Board	A service publication point that offers the chance of registering and searching for services by their profile.	Organizational Internal Contract template Activity
Product	An application or a resource.	Organizational Internal Contract template Activity
Service Profile	The description of a service that the agent might offer to other entities	Organizational Internal Activity
Service Implementation	A service specific functionality which describes a concrete implementation of a service profile	Internal Activity
Task	An entity that represents a basic functionality, that consumes resources and produces changes in the agent's Mental State.	Organizational Internal Contract template Activity
Stakeholder	A group that the organization is oriented to and interacts with the OUs.	Organizational
Believe	A claim that an agent (or a role taken by an agent) thinks that it is true or will happen.	Internal
Fact	A claim that is true at the system's domain.	Internal
Event	The result of an action that changes the state of the system when it occurs.	Internal
Interaction	An entity defining an interaction between agents.	Activity
Interaction Unit	A performative employed during the interaction.	Activity
Translation Condition	An artifact that allows defining the sequence of tasks depending on a condition.	Activity
Executer	A participant in an interaction. It can be an Organization, an Agent or a Role.	Organizational Internal Contract template Activity

Table 1: Definition of ROMAS metamodel elements

1.1 ROMAS metamodel views

1.1.1 ORGANIZATIONAL VIEW

In this view the global goals of the organizations and the functionality that organizations provide and require from their environment are defined (Figure 1). The static components of the organization, i.e. all elements that are independent of the final executing entities are defined too. More specifically, it defines:

- The entities of the system (*Executer*): *AAgents* and *Roles*. The classes *Executer* and *AAgents* are abstractions used to specified the metamodel, but neither of them are used by designers to model systems.
- An *AAgent* is an abstract entity that represents an atomic entity (*Agent*) or a group of members of the organization (*Organizational Unit*), seen as a unique entity from outside.
- The *Organizational Units* (OUs) of the system, that can also include other units in a recursive way, as well as single agents. The *Contains relationships* includes conditions for enabling a dynamical registration/deregistration of the elements of an OU through its lifetime.
- The global *Objectives* of the main organization. The objectives defined in this view are non-functional requirements (softgoals) that are defined to describe the global behavior of the organization.
- The *Roles* defined inside the OUs. In the *contains* relationship, a minimum and maximum quantity of entities that can acquire this role can be specified. For each role, the *Accessibility* attribute indicates whether a role can be adopted by an entity on demand (external) or it is always predefined by design (internal). The *Visibility* attribute indicates whether entities can obtain information from this role on demand, from outside the organizational unit (public role) or from inside, once they are already members of this organizational unit (i.e. private role). A hierarchy of roles can also be defined with the *InheritanceOf* relationship.
- The organization social relationships (*RelSocialRelationship*). The type of a social relationship between two entities is related with their position in the structure of the organization (i.e. information, monitoring, supervision), but other types are also possible. Some social relationships can have a *ContractTemplate* associated which formalize some predefined commitments and rights that must be accepted or negotiated during the execution time. Each *Contract Template* is defined using the Contract Template view.
- The *Stakeholders* that interact with the organization by means of the publication of offers and demands of *Products* and *Services* in the *BulletinBoard*.
- The *Bulletin Board* can be considered as an information artifact for Open MAS. This artifact allows the designer to define the interaction with external entities and facilitates trading processes. When an agent wants to trade, the agent can consult or publish their offer into the BulletinBoard.

Each offer or demand can be associated with a ContractTemplate. It means that this offer or demand has some predefined restrictions which are specified in this ContractTemplate view.

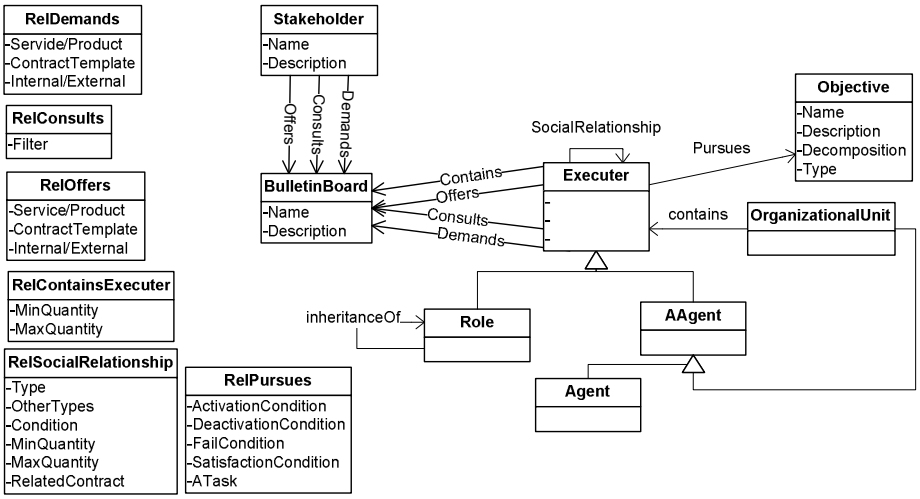

Figure 1: Organizational view (the class *RelXXX* represents the attributes of the relationship *XXX*)

1.1.2 INTERNAL VIEW

This view allows defining the internal functionality, capabilities, believes and objectives of each entity (organizations, agents and roles) by means of different instances of this model (Figure 2). More specifically, it defines the following features of each entity:

- The *Objectives* represent the operational goals, i.e., the specific goals that agents or roles have to fulfill. They can also be refined into more specific objectives. They might be related with a Task or Interaction needed for satisfying this objective.
- The *Mental States* of the agent, using believes, events and facts.
- The *products* (resources/applications) available by an OU.
- The *tasks* that the agent is responsible for, i.e. the set of tasks that the agent is capable of carrying out. Task An entity that represents a basic functionality, that consumes resources and produces changes in the agent's Mental State.
- The *Implements Service Profile*
- Internal entities can publish offers and demands in a *BulletinBoard*, as external stakeholder can do by means of the organizational view. This publications can also have an associated Contract Template to describe some predefined specifications.

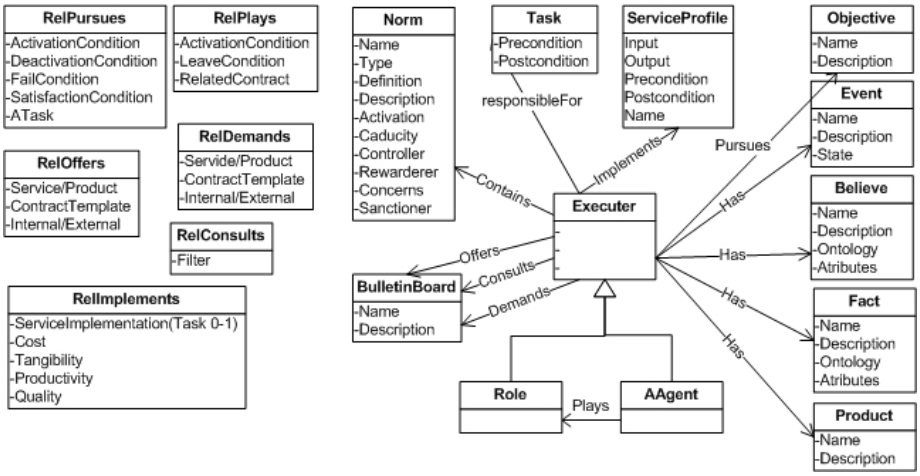

Figure 2: Internal view (the class *RelXXX* represents the attributes of the relationship *XXX*)

- The *roles* that an agent or an organizational unit may play inside other organizational units (*Plays* relationship). *ActivationCondition* and *Leave-Condition* attributes of this relationship indicate in which situation an OU acquires or leaves a role.
- The *roles* played by each agent. *ActivationCondition* and *LeaveCondition* attributes of this *play* relationship indicate in which situation an agent can acquire or leave a role.
- The *Norms* specify restrictions on the behavior of the system entities. The relationship *Contains Norm* allows defining the rules of an organization and which norms are applied to each agent or role. *norms* that control the global behavior of the members of the OU.

1.1.3 CONTRACT TEMPLATE VIEW

This view allows defining *Contract Templates*. Contracts are inherently defined at runtime. Despite this, designers represent some predefined restrictions that all final contract of a specific type should follow by means of a *contract template*. Contract templates 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 syntax of a contract template is defined in Figure 3. More specifically, it defines:

- The relationship *Signants* indicates who is allowed to sign this type of contracts. It could be a specific agent, an agent who plays a specific role or an organization. A *ThirdPart* could be anyone who participates in the negotiation protocol or who is affected by the final execution of the Contract.
- The relationship *Protocol* indicates which protocols are recommended to negotiate this type of contract.

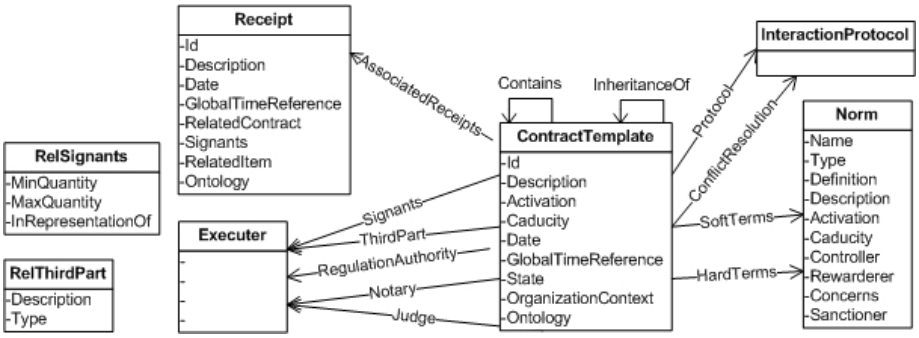

Figure 3: Contract Template view (the class *RelXXX* represents the attributes of the relationship *XXX*)

- After the negotiation, the *Notary* is responsible for verifying the correctness and coherence of the final contract definition. He should check if any term of a contract violate any norm of the regulated environment.
- Each type of contract can define which *Receipts* will be generated during the execution time. Receipts are proves of facts, for example, a receipt can be generated when an agent successfully provides a service.
- In case of conflict, the *Judge* has to evaluate the Complaints and the generated Receipts following the *ConflictResolution* protocol. If he decides that there has been a violation of a norm, the *RegulationAuthority*, who is the main authority in the context of a contract, can punish or reward the agent behaviors.
- The relationship *Hard clause* indicates that any instance of this type of contract has to include this norm. *Soft clause* are recommendations, so during the negotiation stage *Signants* will decide whether this norm will be included or not in the final contract.

1.1.4 ACTIVITY VIEW

This view allows defining the sequence of actions in which a task, a service or a protocol can be decomposed (Figure 4). Each *state* represents an action or a set of actions that must be executed. An *action* is a first order formula that indicates which task or service is executed or which message is interchanged between the agents that participate in this state. The relationship *next* indicate the sequence of states. These sequence can be affected by a *translation condition* that indicates under which circumstances the a state is going to be the next step of the process.

1.2 ROMAS notation

ROMAS models are graphically represented following the notation detailed in Figure 5. This notation is based on the notation used in the GORMAS [**?**] and which was initially proposed by the INGENIAS methodology [**?**]. In order to

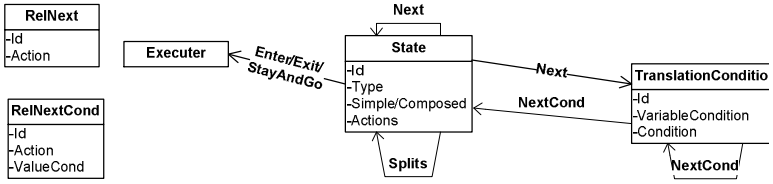

Figure 4: Activity View (the class *RelXXX* represents the attributes of the relationship *XXX*)

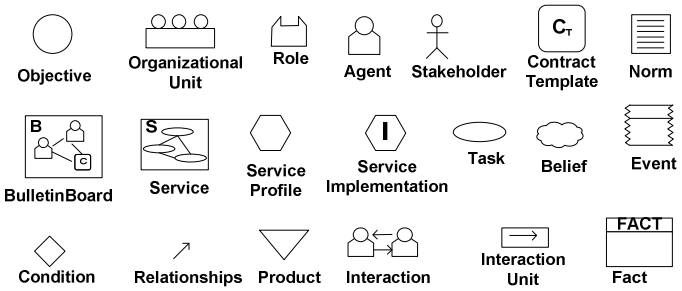

Figure 5: Entities from the ROMAS graphical notation

represent the entities of the ROMAS metamodel that do not exist in these other methodologies like the abstraction of contract template, new graphical icons has been created.