Developing Regulated Open Multi-agent Systems*

Emilia Garcia, Adriana Giret, and Vicente Botti

Universitat Politecnica de Valencia, Camino de Vera S/N, Valencia, Spain
{mgarcia,agiret,vbotti}@dsic.upv.es

Abstract. The demand of computer systems that integrate heterogeneous and autonomous entities and institutions is increasing. These entities and institutions usually coexist in a complex social and legal framework that can evolve to address the different and often conflicting objectives of the many stakeholders involved. In this paper, we present ROMAS, an agent-oriented methodology that guides developers on the analysis and design of systems of this kind. Contracts and norms are used to formalize the normative context and the interactions. This methodology has been described using the template proposed by the FIPA Design Process Documentation and Fragmentation Working Group.

Keywords: Agent methodologies, Normative systems, Contracts

1 Introduction

As collaborative working increases in many domains, there is more and more demand for large-scale, flexible and adaptive software systems to support the interactions of people and institutions distributed in heterogeneous environments. In many cases, the interacting entities are bound by rights, duties and restrictions, which influence their behavior. Commonly, the defining characteristics of systems of this kind are that they are *social*, *open* and *regulated*. First, they are *social* in the sense that autonomous and heterogeneous entities interact between them to achieve global and individual objectives. Besides, the entities of the system can be structured as institutions or groups of agents with similar characteristics, functionality or that interact with the rest of the system as a single entity. Second, they are *open* in the sense that, dynamically at runtime, external parties can interact and become part of the system. Third, they are *regulated* in the sense that every entity or institution in the system can have associated a set of norms that must fulfill. Besides, the expected behavior of each entity should be clearly specified by means of specifying its rights and duties inside the system.

During the last years, agent-oriented methodologies have dealt with the development of systems of this kind [4, 6, 21]. However, as is presented in Section 2, current agent methodologies do not completely cover the analysis and design of

* AT2012, 15-16 October 2012, Dubrovnik, Croatia.

social, open and regulated systems. In this paper we present ROMAS, a methodology for developing systems of this kind that tries to deal with some open issues in this topic. ROMAS phases have been appropriately defined to catch all the requirements of the design of systems from the global system's purposes to the specification of the behavior of each individual entity. ROMAS methodology analyzes and formally describes the normative contexts of a system and its entities. In this paper, we consider the *normative context* of a system to be the set of norms that regulates the behavior of each entity and the set of contracts that formalize the relationships between these entities. The *normative context* of each entity is specified by the set of norms that directly affects the behavior of this entity.

The remain of the paper is organized as follows: Section 2 presents a gap analysis in the support of agent-oriented methodologies to the development of systems of this kind; Section 3 presents the ROMAS methodology following the template proposed by the FIPA Design Process Documentation and Fragmentation Working Group [10]; Section 4 conclude the paper by analyzing the contributions of the ROMAS methodology to the state of art and future works.

2 Related work

In this section, to what extend Multi-agent systems (MAS) methodologies support the analysis and design of social, open and regulated systems is analyzed.

The social structure and coordination are usually represented in agent approaches by means of roles and structured organizations. Recently, some MAS methodologies have adopted an organizational approach [5, 7], and most well-known agent methodologies offer guidelines for identifying and specifying the most suitable organizational structure for a particular system in consideration of its particular requirements [1, 15, 4].

In order to deal with open systems where information, resources and functionality can be interchanged between internal or external heterogeneous entities, some MAS methodologies have integrated organizational abstractions and structures with services [14, 9, 6]. This integration is beneficial in the sense that services offer a well-defined infrastructure and high interoperability, whereas agent technology aims to provide intelligent and social capabilities (trust, reputation, engagement, etc) for applications.

In order to adapt these kinds of systems to legal and restricted environments, agents' social relationships, organizational behavior, interactions and service interchanges must be regulated. The most common mechanism to formalize the rights and duties of each entity of the system is by means of norms and contracts. The specification of a formal syntax for defining norms is well covered by most agent methodologies [3, 21, 7]. Over last years, the integration of electronic contracting into organizational MAS are becoming more important to design system architectures for agent behavior regulation [17]. Contracts are flexible and expressive as they allow businesses to operate with expectations of the behavior of others based on high-level behavioral commitments, and provide flexibility in

how the autonomous agents fulfil their own obligations [22]. Some approaches like [8] use contracts to formalize not only the contractual commitments but also the social relationships between agents and between agents and organizations. However, current methodologies largely omit any guidance on how the normative context of a system should be analyzed and formalized, often assuming this to be trivial. When specifically considering normative context, this assumption is often not reasonable: norms can come from many sources and translating from those different sources to a semi-formal specification is non-trivial. In the literature, only few works provide guidelines for actually identifying the normative context of the system [20, 19, 2]. But, each of these guidelines is focused on the identification of a specific types of norms and they are not integrated in a complete development process for social and open systems.

The verification of the designed models is an important issue in any kind of system. In normative systems many conflicts can arise from the interaction between the different normative contexts of each entity and institution. Regarding the verification of the models and the consistency and coherence of norms and contracts inside an organization, there is some work in the literature but it is still an open issue. Most work here is focused on offline verification of norms by means of model checking [23]. For example, the OMase and Opera methodologies provide case tools to model and verify some properties of these models using model checking (Agent Tool III [13] and Operetta [16] respectively). However, these methodologies do not integrate the verification of the normative context within their phases of the methodology.

After the analysis of the state-of-the-art, we can conclude that none of the studied approaches offers a complete methodology that guides developers from the analysis requirements to the implementation stage taking into account the notions of agent, organization, service, norm and contract. The design of social and open systems is well covered by the literature. There are guidelines that standardize and guide the analysis and design of the organizational structure of the system and the services that each entity offers and requires from the system. However, although some methodologies allow the formalization of norms none of them integrate within their development process guidelines for identifying and formalizing the norms regarding the system requirements. In complex system the process of identifying the normative context of a system is not simple and without any guideline important restrictions of the systems can be omitted. This omission could produce unstable, unreliable and incomplete system designs. Moreover, there is also a lack of methodologies that integrate the verification of the coherence of the normative context within their development process.

3 ROMAS methodology

ROMAS methodology is focused on the analysis and design processes for developing organizational multiagent systems where agents interact by means of services, and where social and contractual relationships are formalized using norms and contracts. In ROMAS, *agents*, *roles* and *organizations* are defined

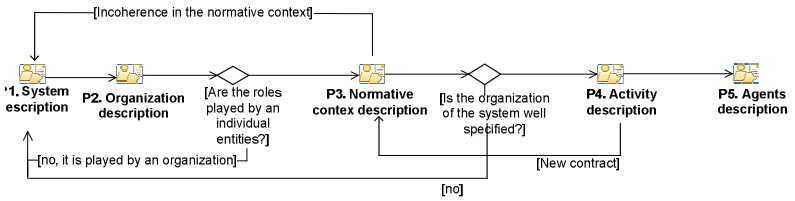


Fig. 1. ROMAS design process

through a formal social structure based on a service-oriented open MAS architecture. 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. *Norms* defined as permissions, obligations and prohibitions restrict the behavior of the entities of the system. *Contracts* are used to formalize the relationships between entities. In our approach, we differentiate between two types of contracts: *contractual agreements* and *social contracts*. Contractual agreements represent commitments between several entities in order to formalize an interchange of services or products. In contrast, social contracts represent commitments between the entities that comprise the structure of the organization. In essence, social contracts can be considered as a subtype of contractual agreements, but the main difference is the purpose of creating the contract. Social relationship contracts are created to formalize the structure and social coordination of the organization, while contractual agreements are created to fulfil the individual interests of the related entities.

The rest of the section presents ROMAS methodology following the template proposed by the FIPA Design Process Documentation and Fragmentation Working Group [10]. The proposed documentation is composed of an introduction that contains an overview of the process and a description of the metamodel used on it (Section 3.1), a description of each phase (Section 3.2) and a case study (Section 3.3).

3.1 Introduction

ROMAS methodology 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. Figure 1 shows the ROMAS design process. This is not a linear process but an iterative one, in which the identification of a new element of functionality may imply the revision of previous defined work products, so it requires to go back to the appropriate phase. For example, during the second phase when a role that can be played by an organization as a whole is detected, it is necessary to go back to the first phase of the methodology to analyze the characteristics, global objectives and structure of this organization.

ROMAS methodology is supported by a formal metamodel, which is described in [12]. This metamodel can be instantiated by means of four different views that analyze the model from different perspectives:
- Organizational External view: This view allows specifying the global goals of the organizations, the functionality that organizations provide and require from their environment and their social structure.
- Internal view: This view allows specifying the internal functionality, capabilities, believes and objectives of each entity (organizations, agents and roles) by means of different instances of this model.
- ContractTemplate view: This view 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.
- Activity view: This view allows specifying interaction protocols and the sequence of activities in which a task is decomposed.

ROMAS methodology is supported by a model-driven case tool based on Eclipse modeling technology [11]. ROMAS models use the same graphical notation than Ingenias [18] and Gormas [1] methodologies.

3.2 Phases of the process

This section describes each phase of the ROMAS process. The description of each phase is composed by a brief introduction, a table that explains the tasks to be executed and a list of the guidelines and work products used and produced during this phase. Due to space limitations the full description of each guideline is not presented here.

PHASE 1: System description As is presented in Table 1, during this phase the analysis of the system requirements, global goals of the system and the identification of use cases are carried out. Besides, the global goals of the organization are refined into more specific goals, which represent both functional and non-functional requirements that should be achieved. Finally, the suitability of the ROMAS methodology for the specific system to develop is analyzed. The result of these activities are formalized by means of the following guidelines and work products:
- System description document This document analyzes the main requirements of the system and its relationship with the environment. It is a structured document that describes the system by means of its main objectives, the external stakeholders with which the system must interact and which services the system must provide to these stakeholders.
- Objectives description document This structured text document analyzes the global objectives of the system and decomposes them into operational objectives. Every global objective specified in the *system description document* is

described using this document. Global objectives are refined into more specific ones that should also be described using this document. The document will be completed when all the global objectives are decomposed into operational objectives, i.e. they are associated to tasks, protocols or restrictions that must be fulfilled in order to achieve these objectives. It is recommended to create one table for each global objective. The first column of each table will contain the properties name, the second the description of the global objective and the following columns the descriptions of the objectives in which this global objective has been decomposed.
- Objective decomposition diagram This is a diagram to graphically represent the decomposition of the objectives. It provides a general overview of the purpose of the system that can be easily understand by domain experts.
- Use case diagram These diagrams are UML graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. The actions identified in the analysis of the operational objectives are related forming activity diagrams in order to clarify the sequence of actions that will be performed in the system.
- ROMAS suitability guideline After analyzing the requirements of the system, it is recommended to use this guideline in order to evaluate the suitability of the ROMAS methodology for the development of the analyzed system. This guideline consists in a set of questions that help the system analyst to decide regarding the requirements of the system when the system should be implemented as distributed or centralized and with or without social features, openness attributes or regulations. ROMAS is appropriate for the development of distributed system, with autonomous entities, with a social structure, with the need of interoperability standards, regulation and trustworthiness between entities and organizations. ROMAS is not suitable for the development of centralized systems or non multiagent systems. Although non normative systems could be analyzed using ROMAS, it is not recommended.

PHASE 2: Organization description As is presented in Table 2, during this phase the analysis of the structure of the organization is carried out. In the previous phase of the methodology, the operational objectives are associated to specific actions or restrictions. In this phase, these actions and restrictions are analyzed in order to identify the roles of the system. A *role* represents part of the functionality of the system and the relationships between roles specify the structure of the system. The result of these activities are formalized by means of the following guidelines and work products:
- Role identification guideline It is an step-by-step guideline that helps the system analyst to detect the main roles of the system and their associated functionality regarding the analysis of the objectives of the system and its use case diagrams.
- Role description document and internal view diagram Each role should be described by means of the structured text *role description document*. This document is associated with a guideline that allows the analysis of the roles

	Task	Description
1.1	Identify system requirements	Following the guideline *system description document*, the requirements of the system are analyzed, including global objectives of the system, stakeholders that interact with the system, products and services are offered and demands to/from stakeholders, external events that the system handles and normative documents such governmental laws attached to the system.
1.2	Identify operational objectives	Following the guideline *objective description document*, the global objectives of the system are analyzed and split into operational objectives, i.e., into more low level objectives that can be achieved by means of the execution of a task or a protocol.
1.3	Identify use cases	Using the information obtained in the previous task, the use cases of the system regarding the tasks and protocols associated to the operational objectives identified are defined.
1.4	Evaluate ROMAS suitability	Following the guideline *ROMAS suitability guideline*, the suitability of the ROMAS methodology for the development of the system to be developed regarding its specific features is evaluated.

Table 1. Phase 1: Activity tasks

and also the analysis of the relationships between them. After this analysis, this information is graphically represented by means of an *internal view diagram* for each role.

- Organizational view diagram One organizational view diagram is created to graphically represent the structure of the system. Besides, this diagram also describes the overview of the system by means of its global *objectives* and how the system interact with its environment of the system (which *services* offers and demands to/from the *stakeholders* and which *events* the system is able to handle). The necessary information to fulfill these diagram is obtained from the *system description document*. Due to the fact that in the literature there are several well-defined guidelines to identify the organizational structure of a system, ROMAS does not offer any new guideline. Instead the use of the guideline defined by the GORMAS methodology in [1] is recommended.

PHASE 3: Normative context description As is presented in Table 3, during this phase the normative context of the system is analyzed by means of identifying and formalizing the norms and the social contracts that regulate the

	Task	Description
2.1	Identify roles	Following the guideline *Role identification guideline* the roles of the system are identified and associated to different parts of the system functionality.
2.2	Describe roles	Following the guideline *Role description document* each identified role is analyzed. The details about each role are graphically represented by means of instances of the *internal view diagram*.
2.3	Identify organizational structure	Identify how the members of the organization interact between them, i.e., which social structure has the organization and graphically represent that using an *organizational view diagram*

Table 2. Phase 2: Activity tasks

	Task	Description
3.1	Identify restrictions from requirements	Following the guideline *Organizational norms*, the system analyst formalizes the norms described in the requirements that regulate the agent behavior. This norms refine the *organizational view* diagram of the organization associated to these norms.
3.2	Identify social contracts	Following the guideline *Social contracts*, the social contracts of the system are identified and formalized by means of the *contract template view diagram*.
3.3	Validate normative context	Following the guideline *Normative context validation*, the coherence among systems norms and between them and the social contracts of the system is validated.

Table 3. Phase 3: Activity tasks

entities' behavior inside the system. The result of these activities are formalized by means of the following guidelines and work products:

- *Organizational norms guideline* This guideline specifies a step-by-step process to identify and formalize restrictions on the behavior of entities gained from the analysis of system requirements. These normative restrictions are associated with specific features of the system, and are usually well known by domain experts but not formally expressed in any document. This guideline helps the system analyst to obtain the necessary information from the domain expert.

- *Social contracts guideline* This guideline specifies a step-by-step process to identify and formalize social contracts inside a specific organization regarding the information detailed in the *role description document*, the roles' *internal view diagrams* and the structure of the organization. Social contracts are used to formalize two kinds of social relationship: (1) *play role contract template*, which specifies the relationship between an agent playing a role and its host organization; and (2) *social relationship contract template*, which specifies the relationship between two agents playing specific roles. Social order thus emerges from the negotiation of contracts over the rights and duties of participants.

- *Contract template view diagram* One contract template diagram is created for each social contract identified with the previous guideline.

- *Normative context validation guideline* This guideline specifies a step-by-step process to validate the coherence of the normative context. The validation of the normative context is understand as the verification that there are no norms in conflict, i.e, that the normative context is coherent. The system analyst can follow the steps of the guideline to perform a manual verification. However, this guideline is implemented in the ROMAS case tool as a plug-in, so it can be executed automatically from the tool.

PHASE 4: Activity description As is presented in Table 4, during this phase each task, service and protocol identified in previous phases of the methodology is described by means of instances of the *activity model view*. Phase 2 identifies the tasks, services and protocols that each role should implement. In phase 3, the contract templates are identified. For each contract identified a negotiation, execution and conflict resolution protocol should be specified.

	Task	Description
4.1	Describe ontology	System domain concepts are analyzed. These concepts will be used to define the inputs, outputs and attributes of tasks, protocols and services.
4.2	Describe services	Define service profile attributes for each service. One *activity view* diagram is created for specifying each service implementation. If there are services that should be published to other members of the system or to external stakeholders, the *organizational view diagram* of the system should be refined by adding a *BulletinBoard*. This abstraction is an artifact where authorized entities can publish and search services.
4.3	Describe tasks and protocols	Specify each task and protocol by means of an instance of the *activity view diagram*. If any protocol should end up with a formal commitment between two entities a contract is specified.

Table 4. Phase 4: Activity tasks

PHASE 5: Agents description As is presented in Table 5, during this phase each identified agent is described by means of an instance of the *internal view* metamodel.

	Task	Description
5.1	Describe agent	Following the guideline *agent description document*, the development requirements of each agent are analyzed.
5.2	Analyze objectives	Following the guideline *objectives description document* detailed in Phase 1, the agent's objectives are analyzed and decomposed in operational objectives.
5.3	Associate with system roles	Identify which roles the agent must play in order to achieve its objectives. This analysis is performed by matching the agent objectives with the roles functionality. Therefore, the *objective description document* of the agent is compared with the analysis of the roles presented in the *roles description documents*.
5.4	Validate coherence	Validate that the normative context of the agent does not avoid any of its objectives to be satisfied. Validate that the agent is able to fulfill its commitments defined by its signed contracts. Validate that there is no incoherence between the normative context of the agent and the normative context of the organizations to which it pertains.

Table 5. Phase 5: Activity tasks

3.3 Case study: Conference management system

The conference manager system (CMS) is a system to support the management of scientific conferences which involves several aspects from the main organization issues to paper submission and peer review, which are typically performed by a number of people distributed all over the world. This section presents the analysis and design of this system using the ROMAS methodology. Due to space limitations only some of the work products produced during the process are presented.

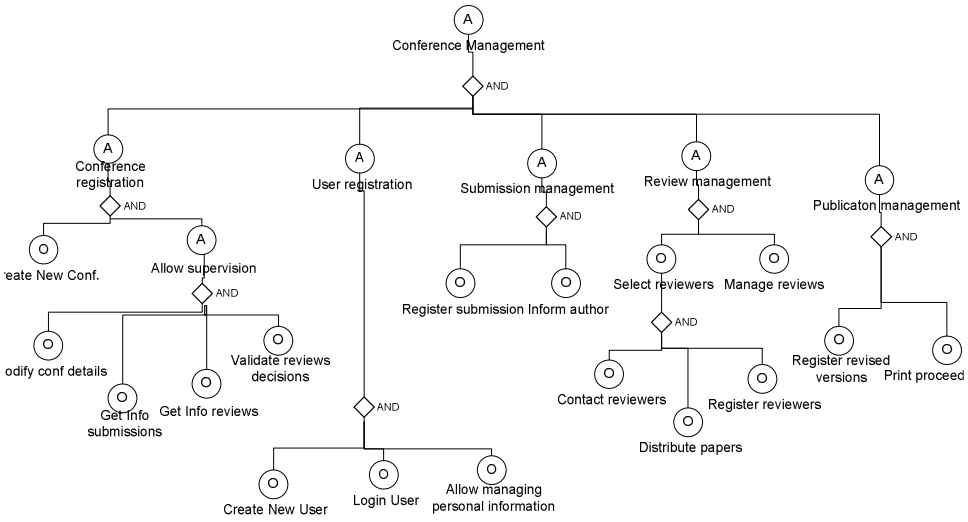


Fig. 2. Case study: Objective decomposition diagram

PHASE 1: System description During this phase, the system requirements are analyzed by means of fulfilling an instance of the *system description document*. The requirements of the system are expressed as objectives by means of the *objective description document*. Every global objective is analyzed and decomposed in operational objectives. The graphical overview of the CMS case study objectives is shown in Figure 2, where A means abstract objective and O means operational objective. The system has five global objectives (*Conference registration, User registration, Submission management, Review management and Publication management*) that are decomposed into operational objectives.

After the analysis of the objectives of the system, its use case diagrams are created and the *ROMAS suitability guideline* is executed. The analysis of the CMS case study features following this guideline shows that ROMAS is suitable for the development of this system. CMS is a distributed system, composed by intelligent systems with social relationships between them. The entities of the system should behavior following the regulations of the system. The rights and duties that an entity acquires when participates in the system should be formalized. For example, reviewers should know before acquiring the commitment of reviewing a paper, the specific terms of this commitment (deadlines, benefits,...). Therefore, a contract-base approach is recommendable for developing this case study.

PHASE 2: Organization description During this phase, the roles of the system are identified and their social structure determined regarding the objectives of the system. The execution of the *role identification guideline* identifies seven roles: (1) The *User* role is an entity of the system that must be registered in order to access to the system. On the contrary of the rest of the roles, this role is not related to any specific conference. (2) The *Author* role is an entity attached to a

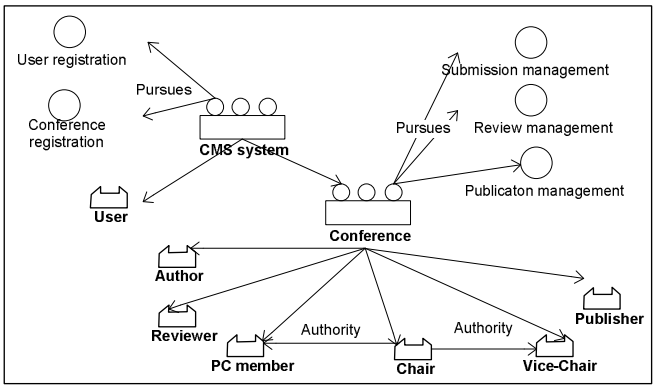


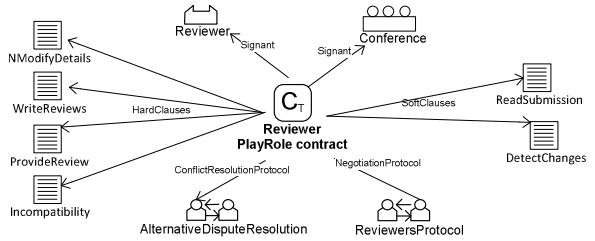
Fig. 3. Case study: Organizational diagram

specific conference in which this role can submit papers and receive information about the status of its papers. (3) The *Chair* role is the main responsible from a conference. This role is able to create a conference and share the responsibility from selecting the reviewers, validate the revisions and update the conference details with the *Vice-Chair* role. (5) The *PC member* role is responsible from managing the reviews, can participate in the selection of the reviewers and have access to the information about submissions and reviews for a specific conference. (6) The *Reviewer* role is responsible from submit the reviews to the system. (7) The *Publisher* role is responsible from managing the revised versions of the papers and print the proceedings.

This roles are described by means of seven instances of the *role description document* and graphically specified by means of seven instances of the *internal view diagram*. Taking into account the features of each role and the requirements of the system the social structure of the system is defined and graphically represented by means of a instance of the *organizational view diagram*. Figure 3 shows this diagram, where the main system is represented by the organization *CMS system* and it can contain others suborganizations called *conference organizations* that represent each conference. Representing each conference as a virtual organization allows each conference to define its own internal legislation and to refine the functionality assigned to each entity of the system.

PHASE 3: Normative context description During this phase, the norms of the system are derived from the requirements, as well as the social contracts between entities. First, the *organizational norms guideline* is executed and several norms arise. As an example of how the identification of a norm can modify the functionality offered by the system, one of the norms that the guideline had indentified was that *"Each conference should describe its internal normative"*. This norm will be attached to every conference, therefore, the task of defining the internal normative should be added to a role inside the conference. In this case, this task has been added to the *chair* responsibilities.

Following the *social contracts guideline*, one *play role contract template* is defined for each role of the organization in order to establish the rights and



NORM ID	NORM DESCRIPTION (Deontic,Target,Activation,Expiration,Action,Sanction,Reward)
NModifyDetails	(FORBIDDEN, IChair, Modify(ConferenceDB),-,-,-,-)
WriteReviews	(PERMITTED, Reviewer, -,-, WriteAccess(Conference database),-,-)
ReadSubmission	(PERMITTED, Reviewer,-, - , UseService(Get Info Submission),-,-)
DetectChanges	(OBLIGED, Reviewer, -,-, Detect(ChangesConference event),-,-)
ProvideReview	(OBLIGED, Reviewer, -,-, ProvideService(Execute review),-,-)
Incompatibility	(FORBIDDEN, Reviewer, reviewer=author,-, UseService(Get Info Submission),-,-)

Fig. 4. Phase 3: Case study - Reviewer play role contract template

duties that any agent playing this role should fulfill. Therefore, seven *play role contract* templates are formalized: one for role *user* of the main organization, and six for each role described inside the *Conference* organization (*author*, *reviewer*, *PC member*, *Chair*, *Vice-chair*, *Publisher*). That means that the rights and duties that an agent that tries to play a role inside a conference can be different depending on how each conference negotiate these contracts. For example, one conference can establishes that a PC member cannot submit a paper while another conference do not add any restriction about this. Since every agent that intends to play an specific role inside the system must sign a *play role contract*, every agent will be aware of its rights and duties inside the organization in advance. As an example, Figure 4 shows the *play role contract template* that any entity that wants to play the role reviewer should sign. There are six clauses attached to this contract template that specify an entity playing this role is not allowed to modify the details about a conference unless it is also the chair of this conference (*NModifyDetails* norm), and neither to access to the submission information about a paper in which he is also author (*Incompatibility* norm). This entity would have permission to access to the reviews database (*WriteReviews* norm) and to use the service *Get Info Submission* (*ReadSubmission* norm). This entity would be obliged to detect when the conference details have changed (*DetectChanges* norm) and to provide the service *Execute review* (*ProvideReview* norm).

PHASE 4: Activity description During this phase, the interactions between entities and the protocols associated to contracts are described by means of instances of the *activity view diagram*. An example is presented in Figure 5. It shows the description of the *reviewer play role negotiation protocol*. First, the chair sends to the user that tries to play the role reviewer the details about the conference (deadlines, topics of interests, ...). The user analyzes this information and if necessary propose a change in the review deadlines. This change can be accepted or rejected by the chair. If the chair rejects the change, he can finish the interaction or modify his proposal and send it again to the user. Once they have agreed the conference details, the chair send the user the specification of

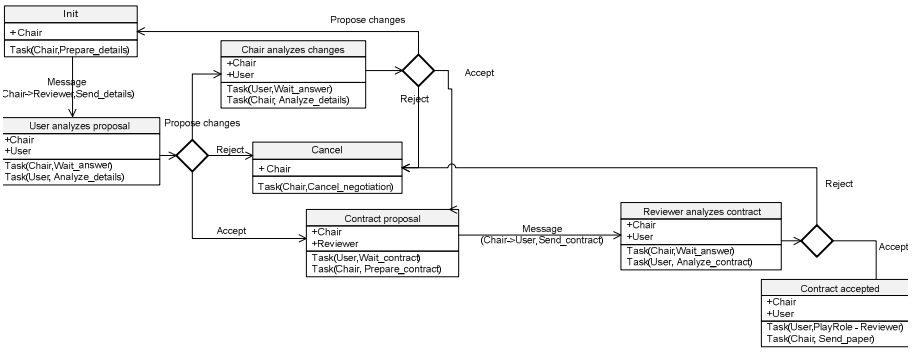


Fig. 5. Phase 4: Case study - Reviewer play role negotiation protocol

the contract, i.e., the rights and duties that the user will acquire if he becomes a reviewer. This contract cannot be negotiated, so the user can reject it and finish the interaction or accept it and begin playing the role reviewer within this conference.

PHASE 5: Agent description During this phase agents representing specific entities that will be implemented in the system are specified. As an example, we present the design of an agent called *PHD student*. First an *agent description document* is created to analyze this agent. After that, each identified objective is analyzed following the guideline *objective description document*. The analysis of the objectives in our running example shows that the main objective of the PHD student agent, *Improve CV*, is decomposed in: *Submit thesis draft*, *Increase number of publications* and *Collaborate in conferences*. The first objective is not related to any objective of the system, so it cannot be achieved inside the conference management system. The second objective, *Increase number of publications*, could be achieved if the agent joined conferences as an author. The authors' play role contract template establish that any agent that wants to join a conference as an author should submit an abstract of the paper. Since this agent is able to create papers that could submit to a conference he can play the role *author*. The third objective, *Collaborate in conferences*, could be achieve by being the PC member of a conference. However, after the validation step it is shown that this agent cannot play the role PC member because any agent that wants to play this role must be a doctor and this agent is a PHD student. Figure 6 shows the internal view diagram of this agent.

4 Conclusions and future work

This paper contributes to the state-of-the-art defining a methodology that guides developers during the analysis and design of systems of this kind. It offers a complete guideline that guides developers from the initial requirement analysis to the definition of concrete tasks and interactions. The whole development process is guided by the global goals of the system and it also takes into account the individual goals of each autonomous entity that interact with the system.

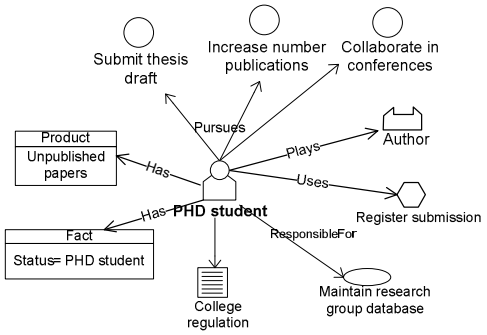


Fig. 6. Phase 5: Case study - PHD student agent description

The use of contracts to define the social and contractual relationships between entities allows the system to operate with expectations of the behavior of others, but providing flexibility in how they fulfil their own obligations. ROMAS methodology deals with some of the open issues detected in Section 2: (1) It integrates the concepts of agent, role, organization, norm and contract during the whole development process. (2) ROMAS not only allow the formalization of the normative context of a system by means of norms and contracts, but also offers guidelines for identifying and formalizing them. (3) It integrates the verification of the normative context of the system in the development process. (4) ROMAS methodology is supported by a case tool that provides automatic verification of parts of the normative context by means of model checking techniques [11]. (5) The description of the methodology using a FIPA standard method allow the reuse of parts of the ROMAS methodology into other development process, as well as, the use of other methodologies fragments into ROMAS. It also facilitates the comparison between methodologies. It can reduce the time that a system analyst needs to learn a new methodology.

As future work we plan to integrate into ROMAS guidelines for identifying the most appropriate interaction protocol regarding the interaction requirements.

Acknowledgments. This work is partially supported by the TIN2009-13839-C03-01, TIN2011-27652-C03-01, CSD2007-00022, COST Action IC0801, FP7-294931 and the FPU grant AP2007-01276 awarded to Emilia Garcia.

References

1. E. Argente, V. Botti, and V. Julian. Gormas: An organizational-oriented methodological guideline for open mas. In *AOSE*, pages 85–96, 2009.
2. T. Breaux. Exercising due diligence in legal requirements acquisition: A tool-supported, frame-based approach. In *Proc. IEEE Int. Requirements Engineering Conference*, pages 225–230, 2009.
3. S. A. DeLoach. Developing a multiagent conference management system using the o-mase process framework. In *Proc. Int. Conf. on Agent-oriented software engineering VIII*, pages 168–181, 2008.

4. S. A. DeLoach and J. C. Garcia-Ojeda. O-mase; a customisable approach to designing and building complex, adaptive multi-agent systems. *Int. J. Agent-Oriented Softw. Eng.*, 4(3):244–280, 2010.
5. S. A. DeLoach, L. Padgham, A. Perini, A. Susi, and J. Thangarajah. Using three aose toolkits to develop a sample design. In *International Journal Agent-Oriented Software Engineering*, volume 3, pages 416–476, 2009.
6. F. Dignum, V. Dignum, J. Padget, and J. Vázquez-Salceda. Organizing web services to develop dynamic, flexible, distributed systems. In *iiWAS '09*, pages 225–234, 2009.
7. V. Dignum. *A model for organizational interaction:based on agents, founded in logic*. PhD thesis, Utrecht University, 2003.
8. V. Dignum, J. Meyer, F. Dignum, and H. Weigand. Formal Specification of Interaction in Agent Societies. *Formal Approaches to Agent-Based Systems*, 2699, 2003.
9. R. F. Fernandez, I. G. Magarinyo, J. J. Gomez-Sanz, and J. Pavon. Integration of web services in an agent oriented methodology. *Journal International Transactions on Systems Science and Applications*, 3:145–161, 2007.
10. FIPA. Design process documentation template standard specification. *http://fipa.org/specs/fipa00097/index.html*, 2012.
11. 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.
12. E. Garcia, A. Giret, and V. Botti. Regulated open multi-agent systems based on contracts. In *Information Systems Development*, pages 243–255, 2011.
13. J. C. Garcia-Ojeda, S. A. DeLoach, and Robby. agenttool process editor: Supporting the design of tailored agent-based processes. In *Annual ACM Symposium on Applied Computing*, pages 8 – 12, 2009.
14. D. Greenwood, M. Lyell, A. Mallya, and H. Suguri. The ieee fipa approach to integrating software agents and web services. *AAMAS Industrial Track*, 2007.
15. B. Horling and V. Lesser. A survey of multi-agent organizational paradigms. *Knowl. Eng. Rev.*, 19(4):281–316, 2004.
16. D. Okouya and V. Dignum. Operetta: A prototype tool for the design, analysis and development of multi-agent organizations (demo paper). In *AAMAS*, pages 1667–1678, 2008.
17. N. Oren, S. Panagiotidi, J. Vázquez-Salceda, S. Modgil, M. Luck, and S. Miles. Towards a formalisation of electronic contracting environments. *COIN*, pages 156–171, 2009.
18. J. Pavon, J. Gomez-Sanz, and R. Fuentes. *The INGENIAS Methodology and Tools*, article IX, pages 236–276. Idea Group Publishing, 2005.
19. M. Saeki and H. Kaiya. Supporting the elicitation of requirements compliant with regulations. In *CAiSE '08*, pages 228–242, Berlin, Heidelberg, 2008.
20. A. Siena, J. Mylopoulos, A. Perini, and A. Susi. Designing law-compliant software requirements. In *Proceedings of the 28th International Conference on Conceptual Modeling*, ER '09, pages 472–486, Berlin, Heidelberg, 2009. Springer-Verlag.
21. P. R. Telang and M. P. Singh. Conceptual modeling: Foundations and applications. chapter Enhancing Tropos with Commitments, pages 417–435. 2009.
22. J. Vázquez-Salceda, R. Confalonieri, I. Gomez, P. Storms, S. P. Nick Kuijpers, and S. Alvarez. Modelling contractually-bounded interactions in the car insurance domain. *DIGIBIZ 2009*, 2009.
23. F. Viganò and M. Colombetti. Symbolic model checking of institutions. In *ICEC*, pages 35–44, 2007.