RESEARCH ARTICLE

Designing Normative Open Virtual Enterprises

Emilia Garcia, Adriana Giret and Vicente Botti

Universitat Politecnica de Valencia, Valencia, Spain

{mgarcia,agiret,vbotti}@dsic.upv.es

(released April 2013)

*Corresponding author. Email: mgarcia@dsic.upv.es
Abstract: There is an increasing interest on developing virtual enterprises in order to deal with the globalization of the economy, the rapid growth of information technologies and the increase of competitiveness. In this paper we deal with the development of normative virtual enterprises (NOVEs). They are systems with a global objective that are composed of a set of heterogeneous entities and enterprises that exchange services following a specific normative context. In order to analyze and design systems of this kind the multiagent paradigm seems suitable because it offers a specific solutions for supporting the social and contractual relationships between enterprises and for formalising their business processes. This paper presents how the ROMAS methodology, an agent-oriented software methodology, can be used to analyse and design NOVEs. ROMAS offers a complete development process that allows to identify and formalise the structure of NOVEs, their normative context and the interactions among their members. The use of ROMAS is exemplified by means of a case study that represents an automotive supply chain.

Keywords: multiagent technology; open systems; normative systems; contracts; software methodology.
1. Introduction

In the past the production management systems used by most manufacturers were comprised of fixed planning and execution processes developed by the same enterprise (W.J. 1996, Peng et al. 1998). However enterprises’ structures and manufacturing processes should evolve in order to deal with the globalization of the economy, the rapid growth of information technologies and the increase of competitiveness. The concept and development of virtual enterprises (VE) are attracting increasing attention from both academic and industrial communities. As is described by Chu et al. (2002) the common defining characteristics of these systems are that they are distributed, collaborative, flexible and product-oriented. In these systems several autonomous enterprises collaborate to design and manufacture high quality and customized products ((X. and C.A. 1997, Deng and Zhang 1997)).

In this paper we deal with the analysis and design of normative open virtual enterprises (NOVE). The term NOVE is understood as a system with a global objective that is composed of a set of heterogeneous entities and enterprises that exchange services following a specific normative context. The composition of a NOVE is not fixed and it can change at runtime in order to adapt the NOVE to changes in the environment, to increase competitiveness or due to changes of interests of an enterprise of the system.

NOVEs are normative in the sense that every entity that is part of a virtual enterprise should follow specific regulations that bound their behaviour. These regulations are derived from legal restrictions and also from the internal best practices of each company. A normative context is understood in this paper as the set of norms that apply to a specific set of entities. For example, a virtual enterprise like "booking.com" can specify a set of norms that apply to all the entities that offer or demand accommodation. This would be one normative context. Besides each company that offers accommodation could add a set of specific norms for its own clients. The company itself would be another normative context. Since a NOVE integrates autonomous and heterogeneous enterprises that can be distributed all over the world, each member of a NOVE may have its own normative context. Thus, the normative context of a NOVE is the integration of all the normative contexts of its members. NOVEs are open in the sense that their structures are not fixed and they can change at runtime. It means that enterprises can dynamically become part or leave a NOVE at runtime.

Multiagent technology has emerged over the last decades as a new software engineering paradigm for building complex, adaptive, flexible systems in distributed heterogeneous environments (Wooldridge and Ciancarini 2001). Therefore, the multiagent paradigm seems intuitively a good approach to deal with the demands for global flexibility, cooperation and, at the same time, the local autonomy of each enterprise.

In this paper we propose an agent-oriented methodology to analyze and design NOVEs. This methodology, called Regulated Open Multiagent systems (ROMAS), was initially designed for developing multi-agent systems (Garcia et al. 2014). In this paper, we present how the ROMAS methodology can be used to analyse and design NOVEs. ROMAS offers a complete development process that allows to identify and formalise the structure of NOVEs, their normative context and the interactions among their members.

The rest of the paper is organised as follows: Section 2 analyzes the requirements for analyzing and designing NOVEs. Section 3 presents how multiagent systems (MAS) concepts and constructions support these requirements. Section 4 discusses to what ex-

---

1In this paper we use the term service as a functionality or a resource that is offered to other entities.
tent current MAS methodologies support the analysis and design of NOVEs. Section 5 presents our proposed methodology and exemplify it by means of a case study. Section 6 discusses the main contributions of this paper.

2. Normative open virtual enterprises design requirements

This section analyzes the common requirements and challenges that need to be dealt during the analysis and design of NOVEs.

**Integration of autonomous and social behavior.** Virtual enterprises can represent real-world enterprises or temporally associations of individual entities with a common objective. A particular challenge is that these systems are composed of disparate entities and enterprises that often fall under different spheres of control. Each enterprise is autonomous and has its own purposes. In this context, interactions can often take place among partners, suppliers and consumers that are managed by parties with conflicting goals, different policies, incompatible data representations, and so on.

A complex enterprise composed of many interacting components offering a wide range of functionality can be departmentalized. This fact simplifies the design, development, implementation and maintenance of the system. When the departments are independent, they could be considered as subsystems (enterprises) of a NOVE. In that sense, a complex enterprise could be designed as a NOVE.

In order to design a NOVE it is necessary to explicitly specify its network structure, the interactions among its members, and the desired functionality of its members. This desired functionality specifies what products should provide each entity and enterprise and under what conditions. If this desired functionality is well specified the NOVE could be composed of independent enterprises with autonomous and even private behaviour that have agreed to comply the requirements. Therefore, it is not necessary to specify the internal structure of all the members of a NOVE in terms of how they are going to provide their functionality. It is enough to specify what they are going to provide and under what conditions. This separation between what is provided and how it is going to be provided increases the flexibility of the design.

**Integration of heterogeneous enterprises.** Since we are dealing with heterogeneous and autonomous enterprises that can have different spheres of control and that can have been developed by different providers, interoperability problems may emerge. Interoperability is an issue that should be solved at implementation time. However, a design that considers this potential issue can facilitate the posterior implementation task and reduce the gap between design and implementation (García et al. 2012).

Some of the issues that must be solved are: (i) **Distributed Data** – the required data is spread widely across all organisations, frequently using different schemas; (ii) **Technical Interoperability** – different organisations often use different (potentially incompatible) technologies; (iii) **Process Interoperability** – different organisations often employ divergent (potentially incompatible) processes to achieve their goals; (iv) **Semantic Interoperability** – different organisations often use different vocabularies and coding schemes, making it difficult to understand the data of others.

Moreover, as these enterprises often operate with a range of aims and priorities in a very dynamic and changing environment, they may have to regularly update their internal processes and technology. However, it is possible that changes may take place without necessarily propagating to all other parts of the system. Interactions and interchanges of services and products should be standardized and formally described in order to isolate
the internal characteristics of the actors of the system from their interactions with the rest of the system (Jain and Dahiya 2012).

**Integration of the different normative contexts.** In the real-world our behavior is restricted by a set of norms derived from the law legislation and from the regulations of the institutions that we belong to or the environments where we interact. Most virtual enterprises are also normative, which means that the behaviour of their entities and companies must be bounded by rights and duties (Cardoso and Oliveira 2005, Davidsson and Jacobsson 2009). Norms provide users and members of a system with expectations about what other entities of the system will do or not do respecting the local autonomy of each enterprise. This ensures they can have confidence in the quality and correctness of what occurs in the system (Hollander and Wu 2011). Norms also avoid critical status of the systems to occur and ensure that the system follows the law regulations established in a specific domain or enterprise.

**Dynamic network structures.** We are trying to deal with large-scale flexible systems that can have many independent enterprises involved in various capacities worldwide. A common feature of large-scale systems is the expectation that more enterprises and entities will join the system (Luck et al. 2011). Open systems are systems that are able to interact with and integrate new entities and enterprises in the system at runtime (Dignum et al. 2007). In order to interact with external entities the system must use standards of communication in order to avoid interoperability problems. Open systems should clearly specify how an external entity can be integrated in the system. One of the most common approaches is to divide the functionality of the system in roles (Botti et al. 2011). Then, if any external entity wants to enter in the system it has to acquire a specific role inside the system. So, once a stakeholder enters in a normative system its behavior is restricted by the rights and duties of the roles that it is playing. The explicit specification of these rights and duties is necessary in order to allow entities to reason about the consequences of acquiring a specific role.

Furthermore, little trust exists between different enterprises, particularly those with conflicting goals and interests. Therefore, interchanges of services between internal or with external entities should be formalized (Meneguzzi et al. 2012). The details of an agreement between two entities is completely specified at runtime, however at design time it may be necessary to specify which kind of relationships are allowed and under which terms (Jakob et al. 2008, Vázquez-Salceda et al. 2009).

As a summary, we can conclude that the responsibility for forming a particular NOVE should be supported by software engineering methods that guide the specification of the NOVE requirements and the development of software that facilitates the creation of the NOVE and its members’ interactions. This software should reflect the social and normative contexts of the enterprises involved. Besides this software should specify the forbidden and enabled interactions among the members of a NOVE at the same time that maintains the flexibility and adaptability of their structure and objectives. The software should also respect the autonomy of each entity of the system and permit their interactions despite their differences in their technology. Therefore, the analysis and design of systems of this kind could be a complex task. The definition of the normative context of a NOVE is a critical task because an incorrect or incomplete definition could cause serious consequences such as the lack of robustness, security and privacy.
3. Multiagent systems for NOVEs

Multiagent systems (MAS) are systems composed of distributed computational components that are autonomous and social, known as agents (Wooldridge 2009). Agents are autonomous in the sense that they are capable of autonomous action, i.e., each individual entity can decide when and how to satisfy its objectives. Agents are social in the sense that they are capable of interacting with other agents in order to cooperate, coordinate, negotiate, and so on. This section revises the general requirements presented in the previous section and matches them with multiagent systems (MAS) constructions and concepts in order to show the suitability of MAS for developing these systems.

Integration of autonomous and social behavior. MAS use high-level abstraction concepts that are very close to real-life concepts such as agents and roles. Agents are computer systems that are autonomous, heterogeneous, reactive, proactive and social (Wooldridge and Ciancarini 2001). Moreover, nowadays the concept of organisation has become a key concept in MAS research (García et al. 2008a). In organisational multiagent systems, organisations may represent enterprises that exist in the real life, or groups of entities that interact between them in order to achieve a common objective and that can be seen from outside as a whole. These high-level abstraction concepts facilitate the communication with domain experts, thereby easing things such as requirements elicitation and verification (Dignum and Dignum 2006).

Integration of heterogeneous enterprises. Agents and organisations in MAS are assumed to be heterogeneous (Wooldridge and Ciancarini 2001). The interoperability problems are commonly solved by many MAS approaches by integrating service-oriented approaches into their architectures (Fernández et al. 2007, Dignum et al. 2009). Services standardize the interactions between heterogeneous entities without restricting the technology or the process followed in order to offer this functionality. Integrating agents and services thus improves flexibility, interoperability, and functionality (García et al. 2008b). 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. Services are a powerful interaction mechanism at implementation and also at design time. The use of services during the design time helps in the specification of different levels of abstraction. Services allow to specify what an entity offers or requires separately from the internal features of this entity and how it is going to offer this functionality (Julian et al. 2009).

Integration of the different normative contexts. As is explained in the previous section, we are dealing with systems that need to bound the behavior of their entities. These restrictions on the behavior are related to system specification requirements, legal documents and internal regulations of the enterprises involved. In order to adapt MAS systems to legal and restricted environments, agents’ social relationships, organisational behavior, interactions and service interchanges are regulated (Hollander and Wu 2011, Boella et al. 2009). Some MAS methodologies, architectures and platforms have been working on explicitly integrating the high-level abstraction of norm (Hollander and Wu 2011). The advantage of a norm-based design approach is that there is a ready way for developers to specify these regulations explicitly in the development process, such that they become part of the design. Implementing the system in a norm-aware platform can ensure their fulfilment, even if the system has been externally implemented by different providers.

Dynamic network structures. In practice, openness is enabled by a design specifying exactly how a new entity must behave in order to join the system (Dignum
et al. 2007). The integration of the concept of contract in MAS architectures facilitates the formalization of the rights, duties and restrictions that any entity acquires when enters in the system playing a specific role (Telang and Singh 2012, 2009). Contracts are flexible and expressive as they allow agents 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 (Vázquez-Salceda et al. 2009). Also, contracts allow formalising the interchanges between enterprises and entities. Therefore, we can conclude that the MAS constructions and concepts fits with the needs of the normative open virtual enterprises. A normative service-oriented MAS paradigm that includes the concept of contract would be suitable for developing NOVEs.

4. Related work

In the previous sections we have conclude that the MAS approach is suitable for developing NOVE’s. This section analyzes to what extent current agent methodologies support the analysis and design of these kinds of systems.

4.1. Agent-oriented methodologies

As is presented in the previous section, some multiagent approaches already integrate the concepts of agent, role, organization, service, norm and contract (DeLoach 2008, Dignum et al. 2003, Telang and Singh 2012, Vázquez-Salceda et al. 2009), however in order to support the analysis and design of a NOVE the development process should include a complete set of guidelines that formalize and facilitate the process. Selecting the most suitable organizational typology and distributing the functionality of the system in the most appropriate way can be a complex task in large and heterogeneous systems. Beyond the complexity of the task, a bad selection of the structure of the organization can be critical for the success of the system (Horling and Lesser 2004a). Some MAS methodologies provide specific guidelines (Argente et al. 2009, DeLoach and Garcia-Ojeda 2010). The social structure and coordination are usually represented in agent approaches by means of roles and structured organizations. However, only a small subset consider the normative context when selecting the organizational structure and a few of these approaches represent the social structure by means of norms in order to allow entities to dynamically reason and change this structure at runtime (Dignum 2003).

Just a few methodologies consider services as an important part of the analysis and design of the system and provide guidelines for specifying their interface as well as their internal functionality (Fernandez et al. 2007, Argente et al. 2009). Without these kinds of guidelines the designer should rely only in his/her expertise to specify the services and their attributes. This task could be very complex in dynamic, distributed, large systems.

The specification of a formal syntax for defining norms is well covered by most agent methodologies (DeLoach 2008, Telang and Singh 2009, Dignum 2003). However, just a few methodologies integrate the concept of contract in their metamodel (DeLoach 2008, Telang and Singh 2012). Some methodologies are integrating into their metamodel the specification of contractual agreements, however, the use of structural norms and contracts to define the structure of the system is only supported by a small subset of methodologies (Dignum et al. 2003). However, current methodologies largely omit any
guidance on how the normative context of a system should be analyzed and formalized. When specifically considering normative context, guidelines are necessary because norms can come from many sources and translating from those different sources to a semi-formal specification is non-trivial. In the literature, a few works provide guidelines for actually identifying the normative context of the system (Siena et al. 2009, Sacki and Kaiya 2008, Breaux 2009). But, each of these guidelines is focused on the identification of a specific type of norms and they are not integrated in a complete development process for social and open systems. No methodology integrates into the development process guidelines that completely support the identification of norms from the analysis of the requirements, nor from legal texts.

**Verification** using any development approach is important, but in normative open systems is even more so due to the high risk of incoherence resulting from interference between different normative contexts, and between the global goals of the system and the individual goals of each party. Although there is some work related to validation and verification of the designed models, it is still an open problem. Most work here is focused on offline verification of norms by means of model checking (Viganò and Colombetti 2007). 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 (Garcia-Ojeda et al. 2009) and Operetta (Okouya and Dignum 2008) respectively). However, these methodologies do not integrate the verification of the normative context within their phases of the methodology.

### 4.2. Applying multiagent technology for developing NOVE

The interest of applying multiagent technology for developing virtual enterprises is gaining more and more interest. Works as the ones presented by Presley et al. (Presley et al. 2001) and the one presented by Assimakopoulos et al. (Assimakopoulos and Theodosi 2003) discusses the advantages and disadvantages of modeling virtual enterprises using different approaches such as object-oriented techniques, agent-oriented methodologies, ontological models or systemic methodologies like Strategis Assumption Surfacing & testing (SAST). However as far as we know there is no previous study of how an agent-oriented methodology can be applied in order to analyze and design a virtual enterprise taking into account its normative environment. Therefore, there is a lack of support for designing NOVEs.

### 4.3. Discussion

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 of 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 systems 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.

In the literature, there are some approaches that offer partial solutions to the problems highlighted. However, the combination of this partial solutions in order to obtain a complete development process that developers can follow is not an easy task because each approach use different terminology, semantics and metamodel constructions. Therefore in the next section we present an agent methodology that tries to cover some of the open issues in the area.

5. ROMAS methodology for NOVEs

ROMAS (Regulated Open Multiagent Systems) is a methodology focused on the analysis and design processes for developing organisational multiagent systems where agents interact by means of services, and where social and contractual relationships are formalized using norms and contracts. ROMAS methodology for analyzing multiagent systems was presented in (Garcia 2013). This section is focused on showing how the ROMAS methodology can be used to analyze and design NOVEs. Therefore some guidelines and design products templates that do not differ from the original version of ROMAS are not included and can be consulted at (Garcia 2013). The main structure of a NOVE in ROMAS is presented in Figure 1. A NOVE is modeled as an organisation and an organisation is composed of:

- A set of global objectives that represent the general purpose of this organisation.
- A set of services that are offered or demanded from its environment. The exchanges of services can be formalized by means of the definition of a contract.
- A set of roles in which the functionality of the organisation is divided. These roles can be played by agents or by other organisations. The rights and duties that an agent or an organisation acquires when it is playing a role can be formalized by means of contracts. In this context, agents represent individual parties. Organisations represent other virtual enterprises, individual companies or specific departments of a company. The fact that one organisation can play a role inside another organisation is used to represent how a NOVE can be composed of several enterprises.
- A normative context that is the set of norms and contracts that all the members of this organisation should fulfill. Norms defined as permissions, obligations and prohibitions restrict the behavior of the entities of the system. Contracts are used to formalise the relationships between entities. Since one organisation (or virtual enterprise) can be composed of several organisations (enterprises), one NOVE can have several normative contexts (one for each organisation). In that sense, every individual member should follow the normative contexts of every organisation of which he is member.

In essence, ROMAS uses the concepts of organisation and role to specify the network structure of a NOVE. This network structure is flexible in the sense that any part of the structure can be substituted dynamically by another organisation or entity that fulfills the terms specified in the contract related to this functionality. Also if the objectives of the NOVE change the contracts that are related to each role can be changed in order to be coherent with the new objectives.

Since each company and department is represented by an organisation and each organisation can define its own normative context, the different regulations of each company can be represented.
The ROMAS methodology provides a well-defined meta-model that can be consulted in (García et al. 2011c).

5.1. Process lifecycle

Figure 2 shows the ROMAS process lifecycle. This section describes each phase of this process following the template proposed by the FIPA Design Process Documentation and Fragmentation Working Group (FIPA 2012) and specifies how ROMAS can be used to design NOVEs. The description of each phase is composed of a brief introduction, a table that explains the sequence of tasks to be executed and a list of the guidelines and work products used and produced during this phase.

**PHASE 1: System specification** 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. Table 1 describes the sequence of tasks that should be executed. The result of these tasks 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 NOVE by means of its main objectives, the external stakeholders with which the NOVE must interact and which services the NOVE must provide to these stakeholders.
1.1 Identify NOVE requirements

Following the guideline system description document, the requirements of the NOVE 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>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Identify NOVE requirements</td>
</tr>
<tr>
<td>1.2</td>
<td>Identify operational objectives</td>
</tr>
<tr>
<td>1.3</td>
<td>Identify use cases</td>
</tr>
<tr>
<td>1.4</td>
<td>Evaluate ROMAS suitability</td>
</tr>
</tbody>
</table>

Table 1. Phase 1: Activity tasks

- **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 analysing the requirements of the NOVE, 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 specification During this phase the analysis of the structure of the NOVE 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. Table 2 describes the sequence of tasks that should be executed. The result of these tasks are formalized by means of the following guidelines and work products:

- **Role identification guideline** It is a 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 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 NOVE. Besides, this diagram also describes the overview of the NOVE by means of its global *objectives* and how the NOVE interacts with its environment (which *services* offers and demands to/from the *stakeholders* and which *events* the NOVE is able to handle). The necessary information to fulfill these diagrams 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 (Argente et al. 2009) is recommended.

PHASE 3: Normative context specification 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 entities’ behavior inside the system. Table 3 describes the sequence of tasks that should be executed. The result of these tasks 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.

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

Table 2. Phase 2: Activity tasks
### Task Description

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Identify restrictions from requirements</td>
</tr>
<tr>
<td>3.2</td>
<td>Identify social contracts</td>
</tr>
<tr>
<td>3.3</td>
<td>Validate normative context</td>
</tr>
</tbody>
</table>

Table 3: Phase 3: Activity tasks

- **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 understood 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 specification** 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*. A task is understood here as the steps that an entity or an enterprise must perform in order to provide a functionality. A protocol is understood here as the sequence of predefined actions, negotiations or exchanges between two or more entities. 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. Table 4 describes the sequence of tasks that should be executed.

ROMAS provides the tools for specifying any interaction protocol such as the negotiation protocol of a contract. However, in contrast with other works like (Cardoso et al. 2012), ROMAS does not provide guidelines for recommending specific protocols regarding the requirements of the interaction. This is part of our future work.

**PHASE 5: Agents specification** During this phase each identified agent is described by means of an instance of the *internal view* metamodel. In the design of a NOVE all the autonomous and individual entities are considered as agents. On the other side, every enterprise or department is considered as an organization. Table 5 describes the sequence of tasks that should be executed in order to specify individual entities.
4.1 Describe ontology
System domain concepts are analysed. 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

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Describe ontology</td>
</tr>
<tr>
<td>4.2</td>
<td>Describe services</td>
</tr>
<tr>
<td>4.3</td>
<td>Describe tasks and protocols</td>
</tr>
</tbody>
</table>

5.1 Describe agent
Following the guideline agent description document, the development requirements of each agent are analysed.

5.2 Analyse objectives
Following the guideline objectives description document detailed in Phase 1, the agent’s objectives are analysed 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

<table>
<thead>
<tr>
<th>Task</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>Describe agent</td>
</tr>
<tr>
<td>5.2</td>
<td>Analyse objectives</td>
</tr>
<tr>
<td>5.3</td>
<td>Associate with system roles</td>
</tr>
<tr>
<td>5.4</td>
<td>Validate coherence</td>
</tr>
</tbody>
</table>

5.2 Case study: Automotive Supply Chain
In order to exemplify this methodology we have analyzed and designed a case study called CVE (Cars Virtual Enterprise). It is a virtual enterprise whose main purpose is manufacturing vehicles. This manufacturer may purchase half or even more of the content of the product from other firms. For example for the manufacture of a vehicle, it might buy the electrical systems from one company, the seats from another and manufacture itself the chassis. Issues of product design and production schedule must be negotiated between these firms in order to produce quality vehicles on time and at reasonable cost. Therefore we are dealing with a network structure where relationships and interchanges are dynamic and must be formalized.

Due to space restrictions, only the most representative diagrams and documents generated during the application of the methodology are presented. Figure 3 summarizes the graphical notation used in the ROMAS diagrams.
PHASE 1: System description During this phase, the requirements for developing the case study CVE are analysed by means of fulfilling an instance of the system description document. The requirements of this NOVE are expressed as objectives by means of the objective description document. Every global objective is analysed and decomposed in operational objectives. The graphical overview of this case study objectives is shown in Figure 4, where A means abstract objective and O means operational objective. The system has three global objectives (Design models, Manufacture vehicles, Sell vehicles) 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 CVE case study features following this guideline shows that ROMAS is suitable for the development of this system. CVE is a distributed system, composed of independent companies that interact between them. All the entities and companies of the system should behave and produce their products following the regulations and specifications established by the CVE.

PHASE 2: Organization description During this phase, the roles of the system are identified and the social structure of the NOVE is determined regarding the objectives of the system. The execution of the role identification guideline identifies twelve roles. 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 5 shows a simplification of this diagram, where the main system is represented by the organization CVE. The CVE divides its functionality and processes in three departments (Design, Manufacturing and Sells). Focusing on the Manufacturing department we can see that its functionality is divided in three main roles: (1) The Assembler role is responsible for integrating all the different parts of the vehicles and produce the final product. This role is played by an internal section of the company. (2) The Manager role is responsible for running the department and controlling the quality of the parts manufactured internally and externally and from the quality of the final product. (3) The Part provider role is responsible for providing a specific part of the vehicle. The provider of each part of the vehicle must fulfill specific requirements. In the diagram we can see that the seats and the electrical systems are
provided by companies that interact with the CVE. However, the chassis are produced by a section of the company CVE. ROMAS methodology guides in the design of the initial organizational structure. At runtime changes in the organizational structure can be derived from changes in the environment of the NOVE, changes in its objectives or because of the specific features of current partners. If a change into the organizational structure or the norms of the NOVE is needed this step of the methodology could be executed again.

**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 identified was that "The companies that
produce a part of the vehicle should sign a confidentiality agreement”. This norm will be attached to all the companies that play a role in the CVE system.

Following the social contracts guideline, one play role contract template diagram is defined for each role of the CVE in order to establish the rights and duties that any company, department or entity playing a role should fulfill. Therefore, twelve play role contract templates are formalized. As an example, Figure 6 shows a simplification of the play role contract template that any company that wants to provide seats for the CVE, i.e., any company that wants to play the role Seats provider should sign. In other words, this contract template specifies the features that any partner that would like to participate in the NOVE as providing seats must fulfill. The diagram shows four clauses attached to this contract template: The Confidentiality Agreement norm that specifies that non-provider can share the designs provided by the CVE. (2) The Produce on Time norm that obligates providers to deliver their products before a specific date or pay a monetary sanction. (3) The Quality requirements norm that obligates to produce the product fulfilling the specified quality requirements. (4) The Incompatibility norm that forbids the provider to produce seats for any other company during two years. This last norm is a softclause. It means that it is recommended that any contract of this type includes this norm but it is not mandatory and whether or not it is included depends on the negotiation at runtime between the manager and the provider.

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 7. It shows the description of the provider play role negotiation protocol. First, the manager sends to the user that tries to play the role provider the details about the terms and conditions of these contractual relationship. The provider analyses this information and if necessary propose a change in terms. This change can be accepted or rejected by the manager. If the manager rejects the change, he can finish the interaction or modify his proposal and send it again to the
provider. Once they have agreed the terms of their relationship, the manager sends the provider the specification of the contract, i.e., the rights and duties that the user will acquire if he becomes a provider to that NOVE. This contract cannot be negotiated, so the user can reject it and finish the interaction, or accept it and begin playing the role provider within this NOVE.

**PHASE 5: Agent description** During this phase agents representing individual entities that will be implemented as software programs are specified. Commonly to many NOVEs our case study designs the system high-level of abstraction point of view. Therefore, it does not consider the implementation of individual agents.

5.3. Discussion: From the design to the implementation

Although this paper is focus on the analysis and design of NOVEs, this section tries to give an overview of what is the role of the methodology in the life cycle of a virtual enterprise and how the works of each phase can be implemented.

The ROMAS methodology provides guidelines, a metamodel and a case tool (Garcia et al. 2011a) in order to analyse and design the initial snapshot of a NOVE. Following this methodology the desired features of a NOVE will be specified in terms of:

- **Objective decomposition diagram**: NOVE’s objectives
- **Organizational diagram**: The stakeholders and entities that can interact and conform the NOVE, and the social relationships between them.
- **Norms and contract template diagrams**: the rights and duties that any entity playing a specific role inside the NOVE must fulfill.
- **Interactions and protocols**: The designed possible interactions and the protocols that must be followed in order to perform every action inside the NOVE.
Internal view diagrams: The specification of the characteristics of entities and enterprises

This design can be used as a way of analyzing and specify a system or it can be translated into a multiagent systems application. This multiagent system would be a software system that would integrate intelligent agents with norms and contracts in order to automatize part of the functionality of the NOVE.

In order to generate this multiagent system application, the ROMAS case tool automatic generation code module would be executed in order to obtain code templates for the Magentix2 agent execution platform (Such et al. 2012).

At runtime the Manager and all the roles of the system will be represented by an intelligent agent that could be programmed to behave automatically, or it could be associated to an user interface in order to behave following direct human orders. This fact allows the human control over important decisions and tasks and the automatization of other functionality.

Magentix2 platform provides mechanisms to create and change the organizational structure dynamically at runtime. Therefore, if, for example, the Manager of a NOVE would realize that a specific type of partner cannot be found, the organizational structure of the NOVE could be changed at runtime. Currently there are lots of works about how an automatic intelligent agent can be aware of the structure of system at runtime and how this agent can propose changes on this structure in order to adapt the system to changes in the environment (Horling and Lesser 2004b, Di Marzo Serugendo et al. 2005).

6. Conclusions

ROMAS contributes to the state-of-the-art defining a methodology that guides developers during the analysis and design of NOVEs. ROMAS offers a complete set of guidelines 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 NOVE and it also takes into account the individual goals of each autonomous entity that interact with the system.

The experiences obtained through the analysis and design of the case study shows that ROMAS is a suitable approach for NOVEs. ROMAS can be used to analyze the current status of an enterprise in order to detect conflicts and detect possible improvements. ROMAS and the Magentix2 platform can be used to implement flexible and dynamic NOVEs or to simulate them. Simulations in this topic are very important because can test the possible outcomes of specific configurations of a NOVE.

In order to analyze and compare ROMAS with other approaches we have used a multiagent systems evaluation framework called MASEV Garcia et al. (2011b). This framework allows analyzing and comparing methods and tools for developing multiagent systems in terms of general requirements, guidelines offered for supporting the development process and the support for the integration of multiagent systems and services. After this analysis we can conclude that ROMAS offers an extensive support for developing NOVEs. Also we have observed that ROMAS differs from other proposals by providing specific guidelines for NOVEs such as guidelines for specifying their structure and their normative context. This evaluation is only a qualitative one. Therefore, as future work we plan to extend this evaluation by introducing quantitative measurements and comparisons.

One of the limitations of ROMAS, detected during the case study implementation, is that the integration of the normative context of each company rises many incoherences.
The computational cost of solving these incoherences automatically is too high and at the moment it has to be done by the domain expert.

There are still open issues in the analysis and design of normative open virtual enterprises such as the integration of guidelines for identifying the most appropriate interaction protocol regarding the interaction requirements and the integration of complete verification guidelines and tools. This can be very useful for the specification of the negotiation protocols and the composition of NOVEs. As future work we plan to improve the ROMAS methodology by dealing with these open issues.

7. Acknowledgments

This work is partially supported by the PROMETEOII/2013/019, TIN2012-36586-C03-01, FP7-29493, TIN2011-27652-C03-00, CSD2007-00022 projects, and the CASES project within the 7th European Community Framework Program under the grant agreement No 294931.

References


REFERENCES


