A Normative Multiagent Approach to Requirements Engineering

Serena Villata

Department of Computer Science, University of Turin, Italy

Abstract. In this paper we present a new model for the requirements analysis of a system. This is a new model based on the multiagent systems paradigm with the aim to support the requirements analysis phase of systems design. This model offers a structured approach to requirements analysis, based on conceptual models defined following a visual modeling language, called dependence networks. The main elements of this visual language are the agents with their goals, capabilities and facts, similarly to the TROPOS methodology [11]. The normative component is present both in the ontology and in the conceptual metamodel, associating agents to roles they play inside the system and a set of goals, capabilities and facts proper of these roles. This improvement allows to define different types of dependence networks, called dynamic dependence networks and conditional dependence networks, representing the different phases of the requirements analysis of the system. This paper presents a requirements analysis model based on normative concepts such as obligation and institution. Our model is a model of semiformal specification featured by an ontology, a meta-model, a graphical notation and a set of constraints. Our model, moreover, allows the definition of the notion of coalition for the different kinds of network.

1 Introduction

The diffusion of software applications in the fields of e-Science and e-Research underlines the necessity to develop open architectures, able to evolve and include new software components. In the late years, the process of design of these software systems became more complex. The definition of appropriate mechanisms of communication and coordination between software components and human users motivates the development of methods with the aim to support the designer for the whole development process of the software, from the requirements analysis to the implementation.

The answer to this problem comes from software engineering that provides numerous methods and methodologies allowing to treat more complex software systems. One of these methodologies is the TROPOS methodology [11], developed for agent-oriented design of software systems. The intuition of the TROPOS methodology [11] is to couple, together with the instruments offered by software engineering, the multiagent paradigm. In this paradigm, the entities composing the system are agents, autonomous by definition, characterized by their own sets of goals, capabilities and beliefs. The multiagent paradigm allows the cooperation among the agents with the aim to obtain common and personal goals. In this way, multiagent systems offer a solution for open, distributed and complex systems and the approach combining software engineering and

multiagent systems is defined Agent-Oriented Software Engineering (AOSE). TROPOS [11] covers five phases of the software development process: the early requirements allowing to analyze and model the requirements of the context in which the software system will be inserted, late requirements describing the requirements of the software system, architectural design and detailed design aiming to design the architecture of the system and, finally, the code implementation.

The TROPOS methodology [11] is based on the multiagent paradigm but it does not consider the addition of a normative perspective to this paradigm. Since twenty years, the design of artificial social systems is using mechanisms like social laws and norms to control the behavior of multiagent systems [6]. These social concepts are used in the conceptual modeling of multiagent systems, for example in requirement analysis, as well as in formal analysis and agent based social simulation. For example, in the game theoretic approach of Shoham and Tennenholtz [29], social laws are constraints on sets of strategies. Together with the rationality assumptions of classical game theory, this leads to the analysis of, for example, stable or minimal social laws, which can be used to choose the best alternative among a set of available social laws. More recently, institutions have emerged as a new mechanism in the design of artificial social systems, which are used in conceptual modeling of multiagent organizations in agent oriented software engineering [37]. Roughly speaking, institutions are structures and mechanisms of social order and cooperation governing the behavior of a set of individuals. They are needed to enforce the global behaviour of the society and to assure that the global goals of the society are met. However, the formal analysis of the institutions is challenging due to the complexity of its dynamics. For example, the agents may change the roles they are playing, or the institution itself may change over time due to the behavior of the agents. In this paper, we propose to add institutions and norms, presented thanks to the normative multiagent paradigm, to the requirements analysis phase. This paper addresses the following research question:

– How to develop a model for requirements analysis based on the normative multiagent paradigm?

Our approach is based, following the approach of TROPOS [11], on a semiformal language of visual modeling and it is composed by the following parts. First, we present an ontology that defines the set of concepts used in the modeling. The elements composing the ontology are agents, goals, facts, skills, dependencies, coalitions with the addition of the normative notions of roles, institutional goals, institutional facts, institutional skills, dynamic dependencies, obligations, sanctions, secondary obligations and conditional dependencies. Second, our meta-model is specified by UML diagrams. These diagrams and the graphical notation establish how to graphically depict the elements composing models. Our model is a directed labeled graph whose nodes are instances of the metaclasses of the metamodel, e.g., agents, goals, facts, and whose arcs are instances of the metaclasses representing relationships between them such as dependency, dynamic dependency, conditional dependency. In TROPOS [11], the requirements analysis is split in two main phases, the early requirements and the late requirements. In our model, these two phases share the same conceptual and methodological approach, thus we call both of them only requirements analysis.

We provide an abstract notion of institution and a definition of a new modeling, called dynamic dependency modeling, based on the structure of dynamic dependence networks. These networks, as classical dependence networks, depict the dependencies among the agents. The dependencies reflect the relation between the goals of agents and agents who have the power to achieve them. In the institutional perspective, institutional powers cannot be captured by the existing dependence networks formalism, since they introduce a dynamic component. Institutional powers can change the norms and permissions of agents playing roles, and, thus, by exercising a power an agent transforms a dependence structure into a new one by adding or removing dependencies thanks to the institutional level of the ontology. Thus, power is seen as the base of the change that is applied to the network describing the system, differently from what expresses by Jones and Sergot [21] and Grossi [20]. Moreover, we introduce the normative issue of obligations, representing them directly in dependence networks. This introduction allows the definition of a third kind of modeling called conditional dependency modeling based on the structure of conditional dependence networks. Conditional dependence networks represent obligations as particular kind of dependencies and these obligations are related to notions as sanctions, if the obligation is not fulfilled, and as contrary-to-duty when the primary obligation, not fulfilled, actives a secondary obligation.

A coalition is an alliance among agents, during which they cooperate in joint action, each one following his own self-interest. We define the notion of coalition in dependence networks, based on the idea that to be part of a coalition, every agent has to contribute something, and has to get something out of it. Since the processes involving coalitions dynamics are complex and costly social behaviors, the idea is that agents have to maintain the stability of their own coalition, paying attention to the possible actions that can be performed by the other agents to strategically increase their profit, mining the coalition or, even worse, destroying the coalition itself. To maintain stability, coalitions have to change dynamically. The possibility to represent coalitions is relevant for systems design and, in particular, for the requirements analysis where the different components of the system can have the necessity to cooperate in a preferential way with a specific subset of other components. The aim of requirements analysis in this context consists in the definition of models able to represent these groups and to provide methods to maintain the stability and the cohesion of these groups.

Our model is not intended to support all analysis and design activities in software development process, from application domain analysis down to the system implementation as in the TROPOS methodology [11]. Moreover, the treatment of a topic like contrary to duty does not concern any connection with deontic logic approaches to solve and analyze this structure such as in Prakken and Sergot [24].

This paper is organized as follows. Section 2 describes a Grid computing scenario as case study for the design of virtual organizations for e-Science and e-Research. In Section 3, we present the core concepts of the ontology and their inter-relations. In Section 4, we define the structure of dynamic dependence networks and we introduce the notion of coalition in this kind of network. Section 5 presents conditional dependence network, introducing some constraints that have to be set for representing coalitions in the conditional dependency modeling. Related work and conclusions end the paper.

2 The Grid Scenario

Grids and the Grid Computing paradigm provide the technological infrastructure to facilitate e-Science and e-Research. Grid technologies can support a wide range of research including amongst others: seamless access to a range of computational resources, linkage of a wide range of data resources, exploitation of shared instruments such as astronomical telescopes or specialized resources such as visualization servers. Historically, much of the focus and effort of Grid computing was based upon addressing access to and usage of large scale high performance computing (HPC) resources such as cluster computers. These access models are typified by their predominantly authentication-only based approaches which support secure access to an account on a cluster. It is often the case that research domains and resource providers require more information than simply the identity of the individual in order to grant access to use their resources. The same individual can be in multiple collaborative projects each of which is based upon a common shared infrastructure. Knowing in what context a user is requesting access to a particular resource is essential information for a resource provider to decide whether the access request should be granted or not. This information is typically established through the concept of a virtual organization (VO) [18]. A virtual organization allows the users, their roles and the resources they can access in a collaborative project to be defined.

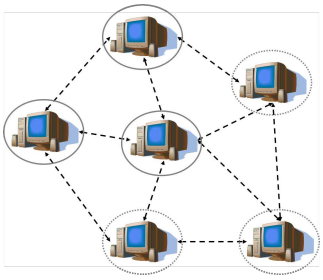

Fig. 1. A Grid composed by six nodes and the interconnections among them.

In the context of virtual organizations, there are numerous technologies and standards that have been put forward for defining and enforcing authorization policies for access to and usage of virtual organizations resources. Role based access control (RBAC) is one of the more well established models for describing such policies. In the RBAC model, virtual organization specific roles are assigned to individuals as part of their membership of a particular virtual organization. Possession of a particular role, combined with other context information, such as time of day and amount of resource being requested, can then be used by a resource gatekeeper to decide whether an access request is allowed or not. One of the key advantages is that whilst individuals in a virtual organization may come and go, the role itself is unlikely to change as much. Consequently RBAC based approaches are considered more scalable and manageable. The key advantage of RBAC-based security models compared to other approaches is

that privileges and access is determined by roles and memberships a user holds and not merely on identity. Indeed the common philosophy underlying the Grid is that all resource providers are expected to be autonomous, i.e. they may allow/deny access requests at their own discretion. Nevertheless, a crucial consideration in establishing a virtual organization is whether a common understanding of the various roles and their associated privileges needs to be established throughout the entire virtual organization or not.

There are two primary models for defining roles specific to a virtual organization: the centralized and decentralized models [18]. In the centralized model, all sites agree in advance on the definition and names of the roles that are applicable to their particular virtual organization, and the privileges that will be assigned to them. A single virtual organization administrator is then appointed who will typically assign these roles to individuals on a case by case basis when users ask to be granted particular roles or permissions in the virtual organization. The decentralized virtual organization role model is more aligned with the original dynamic collaborative nature of the Grid. In this model there is no central virtual organization administrator. Instead, each resource site has its own local administrator who is completely responsible for determining which virtual organization members can access the local virtual organization resources. Each site administrator determines the roles and the associated privileges that are required to access and use the local resources. Consequently, they can decide which other administrators (at this and other virtual organization sites) are trusted to assign which roles to which virtual organization users. In this way they may each delegate to each other the responsibility of user-role assignments throughout the virtual organization. This model allows for more dynamic collaborations to occur. Thus rather than all sites having to agree on virtual organization-wide roles and develop associated policies, the decentralized model allows a resource administrator to directly provide end users and trusted end user administrators with the privileges they need to enable access to his resource.

Role based access control systems make access control decisions based on the roles that users hold. Traditional output of the access control decisions are *Granted* and *Denied*, which dictate whether the requests are authorized or not. As presented by Zhao et al. [38], obligations are requirements and tasks to be fulfilled, which can be augmented into conventional systems to allow extras information to be specified when responding to authorization requests. For example in [38], administrators can associate obligations with permissions, and require the fulfillment of the obligations when the permissions are exercised. The base model associated users with roles, and roles with permissions. Users, being members of roles, acquired all permissions associated with the roles. The hierarchical model enhanced the base model by allowing senior roles to acquire permissions of their junior roles. The general idea of the role based access control model is that, permissions are associated with functional roles in organizations, and members of the roles acquire all permissions associated with the roles. Allocation of permission to users is achieved by assigning roles to users. An obligation is associated with privileges, and when an operation is performed, the obligation associated to the privilege which authorizes the operation is activated. Obligations are requirements to be performed by a specific deadline. Failure of the fulfilling an obligation will incur a sanction.

Some of the main features of a node in a Grid are reliability, degree of accepted requests, computational capabilities, degree of faults and degree of trust for confidential data. These different features set up important differences among the nodes and the possible kinds of coalitions that can be formed and maintained. In this scenario, as in the following examples, we do not consider the way the coalitions are formed but we are interested in coalitions' evolution. We think of already formed coalitions and we discuss the notion of stability and the possible ways to regulate these coalitions thanks to the use of obligations. The idea is that coalitions emerge thanks to the preferred relationships among the different nodes, e.g., each node maintains a sort of list of the more trusted nodes forming a coalition with it. Reciprocity-based coalitions can be viewed as a sort of virtual organizations in which there is the constraint that each node has to contribute something, and has to get something out of it.

The scenario of virtual organizations based on Grid networks represents a case study able to underline the benefits of a normative multiagent paradigm for requirements analysis. First of all, in the normative multiagent paradigm as well as in the common multiagent one, the autonomy of agents is the fix point of all representations, i.e., the Grid philosophy imposes the autonomy of the nodes composing it. Second, the normative multiagent paradigm allows a clear definition of the notion or role and its associated permissions, i.e. the role based access control policy needs a design able to assign roles and represents all the consequent constraints based on them. Third, the normative multiagent paradigm allows the introduction at requirements analysis level of obligations able to model the system. Fourth, the concept of coalition and the constraints introduced by this concept to the early and late requirements model can design the concept of "local network" in virtual organizations. Finally, the modeling activities of dependency modeling, dynamic dependency modeling and conditional dependency modeling depict the system using structures similar to the Grid network itself.

3 Institutional MAS: agents, roles and assignments

Since last years, many factors have caused a great increase of the complexity of software systems. Applications such as e-commerce, e-services, e-science, e-research are clear example of this kind. The software for these applications has to be based on open architectures and it has to evolve over time to integrate new hardware components and answer to the necessity of new requirements. Our model is addressed to the representation of the requirements of the system using the normative multiagent paradigm. This model is based firstly on an ontology containing a number of concepts related to each other. We divide our ontology in three submodels: the agent model, the institutional model, and the role assignment model, as shown in Figure 2. The Figure depicts the three submodels which group the concepts of our ontology.

Such a decomposition is common in organizational theory, because the organization can be designed without having to take into account the agents that will play a role in it. Also, if another agent starts to play a role, for example if a node with the role of simple user becomes a VO administrator, then this remains transparent for the organizational model. Likewise, agents can be developed without knowing in advance in which institution they will play a role.

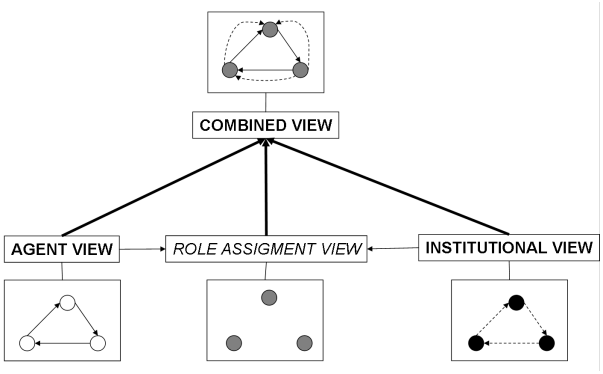

Fig. 2. The conceptual metamodel.

The notion of agent and all its features as goals and capabilities are used in the conceptual modeling as in TROPOS [11]. In our model, we add to these notions those related to the institution such as the notion of role and all its institutional goals, capabilities and facts. Both these notions, combined in the combined view, are used in the conceptual modeling and to each agents it is possible to assign different roles depending on the organization in which the agent is playing. Adding the institution, to each agent are associated both a number of physical features and a role with all its institutional features. In this way, early and late requirements can be based both on agents and on roles. The models are acquired as instances of a conceptual metamodel resting on the concepts presented in the following subsections. We present our three submodels as definitions and each definition contains the concepts belonging to this particular subset of the ontology.

3.1 Agent View

The representation of the system from a material point of view, called agent view, can be imagined as composed by a set of agents, each of them with its associated sets of skills and goals and a set of actions, a set of facts describing the world and a set of rules that allow the application of an action by an agent that can perform it and the consequences of the action on the system. The definition of the agent view is as follows:

Definition 1 (Agent view).
$\langle A, F, G, X, goals : A \rightarrow 2^G, skills : A \rightarrow 2^X, rules : 2^X \rightarrow 2^G \rangle$ *consists of a set of agents* A*, a set of facts* F*, a set of goals* G*, a set of actions* X*, a function* $goals$ *that relates with each agent the set of goals it is interested in, a function* $skills$ *that describes the actions each agent can perform, and a set of rules, represented by the function* $rules$ *that relates sets of actions with the sets of goals they see to.*

Example 1. Considering a virtual organization on a Grid with a role based access control policy, the agent view is used to describe the set of legitimate users of the system, represented inside the Grid as nodes. Each user is provided by a set of actions he can do,

represented by the set X, e.g., to save a file on his file system or to start a computation on his personal computer, and by a set of goals he would fulfill, represented as the set G, e.g., he wants to reserve half of his available memory for his data or he has to obtain the result of a computation in two hours. These actions X can be compared to the operations that are recognized by the system. Functions $goals$ and $skills$ link each agent with the actions he can perform and with the goals he would obtain. Function $rules$ is a sort of action-consequence function, relating sets of actions with the goals they allow to fulfill, e.g., to obtain the results of a computation in two hours, the user has to start the computation on his personal computer.

3.2 Institutional View

A social structure is modeled as a collection of agents, playing roles regulated by norms where “interactions are clearly identified and localized in the definition of the role itself" [37]. The notion of role is notable in many fields of Artificial Intelligence and, particularly, in multiagent systems where the role is viewed as an instance to be adjoined to the entities which play the role. According to Ferber [16], “A role describes the constraints (obligations, requirements, skills) that an agent will have to satisfy a role, the benefits (abilities, authorizations, profits) that an agent will receive in playing that role, and the responsibilities associated to that role". In TROPOS [11], the role is one of the three specification of the concept of actor and it is an abstract characterization of the behaviour of the social actor inside the specific context of the application domain. In our model the notion of role is inserted into the submodel called institutional view. Our characterization of roles is less rigid that in [37]. There is not just one level of interaction and of dependency (the ’formal’ one), but, on the contrary, there are two layers, one exploiting the other, but also diverging: the personal goals, skills, dependencies and the role goals, skills, dependencies. The institutional view is defined as follows:

Definition 2 (Institutional view).
$\langle RL, IF, RG, X, igoals : RL \rightarrow 2^{RG}, iskills : RL \rightarrow 2^{X}, irules : 2^{X} \rightarrow 2^{IF} \rangle$ *consists of a set of role instances RL, a set of institutional facts IF, a set of public goals attributed to roles RG, a set of actions X, a function $igoals$ that relates with each role the set of public goals it is committed to, a function $iskills$ that describes the actions each role can perform, and a set of institutional rules, represented by the function $irules$, that relates a set of actions and the set of institutional facts they see to.*

Example 2. The institutional view represents in the Grid scenario a sort of meta-model for the role based access control policy. In fact, this view represents all the possible roles that can be instantiated in the system and all the possible actions and goals related to each of these roles. For example, we can think to a Grid system with the two basic roles of virtual organization administrator and virtual organization member. These two roles are different depending on the actions they can perform. For example, the VO administrator has the possibility to assign to the VO members the privileges they need to enable the access to its resource. Our approach gives the opportunity to define not only the capabilities of a particular role but also the definition of institutional goals associated to roles, differently from other approaches such as [38] [18]. The institutional view is

a way to represent permissions of the users of the system. Users, being assigned to a particular role, acquire all permissions (in this view represented as rules by the function $irules$) associated to the role. In this way, the allocation of permissions to users is achieved by assigning roles to users. In the Grid computing field, a permission is an approval of performing an operation on a specific target. In our model, we represent a permission in a virtual organization as the actions that a role can perform and what goals these actions allow to achieve. For example, a user asks for storing a file on the file system of another node. This user is associated to a role, since he belongs to a virtual organization regulated by a role based access control policy. The request can be processed either by the local VO administrator or by the user that has received the request. If the user requesting the service has a role that can perform this action, the request is accepted and the file is saved. In this case, we consider for simplicity the case in which the request is always accepted if the role has the permission to do it without thinking of malicious behaviours.

3.3 Role Assignment View

In TROPOS [11], the position of the actor represents the set of roles played by a single agent. In our model, we introduce the third submodel,the role assignment view, which links the agent view and the institutional view to each other, by relating agents to roles.

Definition 3 (Assignment view).
$\langle A, RL, roles\colon RL \rightarrow A\rangle$ consists of a set of agents A, a set of role instances RL, and a function $roles$ assigning a role to its player in A.

Example 3. The assignment view relates each agent with the role it is associated with. In virtual organizations, this kind of assignments is done by the VO administrator, in the centralize model, and by the VO local administrators, in the decentralized model. In our model, there is not a constraint on what kind of agent has the power to assign roles and thus privileges to the users. The assignment view can be eventually restricted to one of the two cases of centralized and decentralized model.

3.4 Combined View

In our model, the system is provided with two distinct views, the material one, called the agent view, and the institutional one, called institutional view, that aims to regulate the behaviour of the agents and to present the permissions associated to each role. Usually, in a multiagent system each agent is related to a set of facts and goals the other agents cannot change since all agents are autonomous. All these features are presented in the concepts of the agent view. But a multiagent system is composed by a multitude of agent that, thanks to their existence inside a social structure, are provided by new sets of facts and goals, the institutional ones, representing permissions. The combined view unifies the agent view and the institutional view thanks to the assignment view providing thus the combined and unified conceptual metamodel:

Definition 4 (Combined view).
Let $\langle A, RL, roles : RL \rightarrow A\rangle$ *be a role assignment view for the agents and role instances defined in the agent view* $\langle A, F, G, X,$ *goals*$: A \rightarrow 2^G,$ *skills*$: A \rightarrow 2^X, rules : 2^X \rightarrow 2^G\rangle$ *and institutional view* $\langle RL, IF, RG, X, igoals : RL \rightarrow 2^{RG}, iskills : RL \rightarrow 2^X, irules : 2^X \rightarrow 2^{IF}\rangle$. *The role playing agents are* $RPA = \{\langle a, r\rangle \in A \times RL \mid r \in roles(a))\}$. *The combined view associates with the role playing agents the elements of the agent and institutional view.*

Example 4. The agents start with their sets of personal beliefs and goals and, only after their insertion inside a social structure, they enlarge their sets of goals and beliefs. In particular, the set of goals is enlarged with new normative goals that represent the responsibilities of the agent inside its social structure while the set of beliefs is enlarged with new normative beliefs representing the set of constitutive norms of the systems, norms based on the collective acceptance of the society representable by means of an institutional ontology.

3.5 Dependency Modeling

Our model is a directed labeled graph whose nodes are instances of the metaclasses of the metamodel, e.g., agents, goals, facts, and whose arcs are instances of the metaclasses representing relationships between them such as dependency, dynamic dependency, conditional dependency. The building of a model involves many activities contributing to the process of definition of the model itself. Our modeling is based on the theory of the social power and dependence pioneered by Castelfranchi [14] as starting point and then developed in the context of coalition formation by Sichman [30] and Sauro [26]. The theory of social power and dependence is an attempt to transfer theories developed initially in the field of sociology to the field of multiagent systems and to refine them. This theory models the potential interactions among the agents which lead to the achievement of a shared goal, i.e. cooperation, or the reciprocal satisfaction of their own goals, i.e. social exchange. This involves the development of a social reasoning mechanism that analyzes the possibility to profit from mutual-dependencies, e.g., the case in which two agents depend on each other for the satisfaction of a shared goal, or reciprocal-dependencies, e.g., the case in which two agents depend on each other for the satisfaction of two different goals.

In a multiagent system, since an agent is put into a system that involves also other agents, he can be supported by the others to achieve his own goals if he is not able to do them alone. This leads to the concept of power representing the capability of a group of agents (possibly composed only by one agent) to achieve some goals (theirs or of other agents) performing some actions without the possibility to be obstructed. The power of a group of agents is defined as follows:

Definition 5 (Agents' power).
$\langle A, G, power : 2^A \rightarrow 2^{2^G}\rangle$ *where* A *is a set of agents,* G *is a set of goals. The function* $power$ *relates with each set* $S \subseteq A$ *of agents the sets of goals* $G_S^1, \ldots, G_S^m$ *they can achieve.*

Example 5. In the Grid scenario, the simplest kind of example of power consists in the power of the local or global administrator to give to common users the possibility to access a resource. Particularly, if we consider a role based access control policy, the Grid administrator has the power to give to the common users, under request, a new role which makes them able to access to a resource. Other kinds of power are, for example, the power to perform a heavy computation or to store a great amount of data.

The notion of power brings to the definition of a structure with the aim to show the dependencies among agents. In order to define these relations in terms of goals and powers, we adopt, as said, the methodology of dependence networks developed by Conte and Sichman [31]. In this model, an agent is described by a set of prioritized goals, and there is a global dependence relation that explicates how an agent depends on other agents for fulfilling its goals. For example, $dep(\{a, b\}, \{c, d\}) = \{\{g_1, g_2\}, \{g_3\}\}$ expresses that the set of agents $\{a, b\}$ depends on the set of agents $\{c, d\}$ to see to their goals $\{g_1, g_2\}$ or $\{g_3\}$. For each agent we add a priority order on its goals, and we say that agent a gives higher priority to goal g_1 than to goal g_2, written as $\{g_1\} \succ (a) \{g_2\}$, if the agent tries to achieve goal g_1 before it tries to achieve g_2. In other words, it gives more attention to g_1 than to g_2. A dependence network is defined as follows:

Definition 6 (Dependence Networks (DN)).
A dependence network is a tuple $\langle A, G, dep, \geq \rangle$ *where:*

- A *is a set of agents;*
- G *is a set of goals;*
- $dep : 2^A \times 2^A \rightarrow 2^{2^G}$ *is a function that relates with each pair of sets of agents all the sets of goals on which the first depends on the second.*
- $\geq$: $A \rightarrow 2^G \times 2^G$ *is for each agent a total pre-order on goals which occur in his dependencies:* $G_1 \geq (a) G_2$ *implies that* $\exists B, C \subseteq A$ *such that* $a \in B$ *and* $G_1, G_2 \in depend(B, C)$.

Dependence networks represent our first modeling activity, the *dependency modeling*, consisting in the identification of the dependencies among the agents and among the roles. In the early requirements phase, we model the dependencies among the agents and the roles associated to the agents of the organization. In this way, we represent the domain stakeholders and we model them using the multiagent paradigm with the addition of the normative component with its related concepts. These dependencies are based both on goals and institutional goals. In the phase of late requirements, the same kind of approach is followed but the agents involved in the dependence network are those of the future system. A graphical representation of the model of the *dependency modeling* is built following the legend of Figure 3 which describes the agents (depicted as white circles), the roles (depicted as black circles), the agents assigned to roles (depicted as grey circles), the agents'/roles' goals (depicted as white rectangles) and the dependency among agents (one arrowed line connecting two agents with the addition of a label which represents the goal on which there is the dependency). For simplicity, the legend considers the dependency only among agents but these dependencies can be also among roles or agents assigned to roles.

We present a first example of modeling a virtual organization based on a Grid network containing only the notions of the agent view.

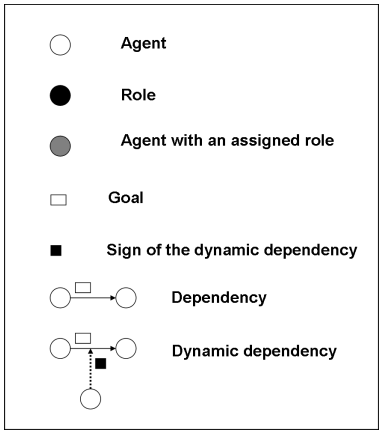

Fig. 3. The legend of the graphical representation of the modeling activities of *dependency* and *dynamic dependency*.

Example 6. Considering a Grid composed by the nodes of Figure 1, we can imagine to view each node as an agent and we can form the following dependence network $DN = \langle A, G, dep, \geq \rangle$, depicted in Figure 4:

1. Agents $A = \{n_1, n_2, n_3, n_4, n_5, n_6\}$;
2. Goals $G = \{g_1, g_2, g_3, g_4, g_5, g_6\}$;
3. *dep*$(\{n_1\}, \{n_2\}) = \{\{g_1\}\}$: agent n_1 depends on agent n_2 to achieve the goal $\{g_1\}$: to store the file *comp.log*;
   dep$(\{n_2\}, \{n_3\}) = \{\{g_2\}\}$: agent n_2 depends on agent n_3 to achieve the goal $\{g_2\}$: to run the file *mining.mat*;
   dep$(\{n_3\}, \{n_1\}) = \{\{g_5\}\}$: agent n_3 depends on agent n_1 to achieve the goal $\{g_5\}$: to store the file *satellite.jpg*;
   dep$(\{n_4\}, \{n_6\}) = \{\{g_3\}\}$: agent n_4 depends on agent n_6 to achieve the goal $\{g_3\}$: to run the file *results.mat*;
   dep$(\{n_6\}, \{n_5\}) = \{\{g_4\}\}$: agent n_6 depends on agent n_5 to achieve the goal $\{g_4\}$: to store the file *satellite.mpeg*;
   dep$(\{n_5\}, \{n_3\}) = \{\{g_6\}\}$: agent n_5 depends on agent n_3 to achieve the goal $\{g_6\}$: to have the authorization to open the file *dataJune.mat*;

Example 6 shows the dependence network based on a simple Grid example composed by six agents. The kind of dependencies are all related to the agent view and they always refer to material goals and not to the institutional ones, except for goal g_6. Using dependence networks as methodology to model a system advantage us from different points of view. First, they are abstract, so they can be used for example for conceptual modeling, simulation, design and formal analysis. Second, they are used in high level design languages, like TROPOS [11], so they can be used also in software implementation.

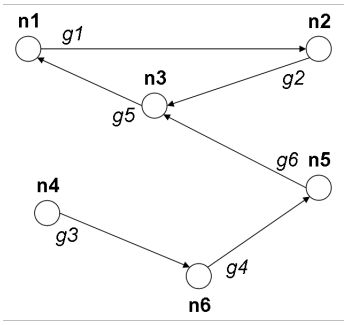

Fig. 4. Dependence Network of Example 6.

4 Dynamic Dependency Modeling

In this section, we answer to the following subquestions: *How to extend dependence networks to build a new modeling activity able to model the dynamics intrinsic to the notion of institutional view?* And, *how to model coalitions in dependence networks?*

In multiagent environments, autonomous agents may need to cooperate in order to fulfill their goals. Each group of agents may have different degrees of efficiency in the achievement of its own goals due to differing capabilities of its members. A requirements analysis model has to consider also the possible presence of groups of agents collaborating to each other. We call these groups coalitions. In this section, we introduce the concept of coalition in our conceptual metamodel.

4.1 Dynamic Dependence Networks

In Section 3, we introduce the different views composing our conceptual metamodel. On the one hand, we have the agent view where one of the main features is that, since agents are autonomous by definition, no goals and skills can be added to an agent. On the other hand, we have the institutional view where the institutional goals, skills and rules can be added to a role, always maintaining the assumption of agents' autonomy. The main changes that can occur thanks to the introduction of the institutional view during the system's evolution are the addition or deletion of an $igoal$, of an $iskill$ and of an $irule$. These additions and deletions change the number of dependencies and the agents involved in them, passing from a dependence network to another one. This change can be represented by means of dynamic dependence networks.

We extend Sichman and Conte's [31] theory with conditional dependencies, in which agents can create or destroy dependencies by introducing or removing powers and goals of agents. Goals can be introduced if they are conditional, or when the agent can create normative goals by creating obligations for the other agents. Otherwise, if an $iskill$ or an $irule$ is introduced, we have the representation of permissions since these additions allow the role to perform a wider number of actions to achieve its goals.

Dependence networks are used to specify early requirements in the TROPOS methodology [11], and to model and reason about the interactions among agents in multiagent

systems. Dynamic dependence networks have been firstly introduced by Caire et al. [12] and then treated in Boella et al. [7], in which a dependency between agents depends on the actions of other agents and, in particular, agents can delete the goals of the other ones. Here we distinguish "negative" dynamic dependencies where a dependency exists unless it is removed by a set of agents due to removal of a goal or ability of an agent, and "positive" dynamic dependencies where a dependency may be added due to the power of a third set of agents. The *Dynamic dependency modeling* represents the second activity modeling for requirements analysis of the system.

Definition 7 (Dynamic Dependence Networks (DDN)).
A dynamic dependence network is a tuple $\langle A, G, dyndep^{-}, dyndep^{+}, \geq \rangle$ *where:*

- A *is a set of agents;*
- G *is a set of goals;*
- $dyndep^{-} : A \times 2^{A} \times 2^{A} \rightarrow 2^{2^{G}}$ *is a function that relates with each triple of a agent and two sets of agents all the sets of goals in which the first depends on the second, unless the third deletes the dependency.*
- $dyndep^{+} : A \times 2^{A} \times 2^{A} \rightarrow 2^{2^{G}}$ *is a function that relates with each triple of a agent and two sets of agents all the sets of goals on which the first depends on the second, if the third creates the dependency.*
- $\geq$: $A \rightarrow 2^{G} \times 2^{G}$ *is for each agent a total pre-order on goals which occur in his dependencies:* $G_1 \geq (a) G_2$ *implies that* $\exists B, C \subseteq A$ *such that* $a \in B$ *and* $G_1, G_2 \in dyndep^{-}(a, B, C)$ *or* $G_1, G_2 \in dyndep^{+}(a, B, C)$.

The static dependencies are defined by $dep(a, B) = dyndep^{-}(a, B, \emptyset)$.

A graphical representation of the model of the *dynamic dependency modeling* activity is built following the legend of Figure 3 which describes the sign of the dynamic dependency (depicted as a black square) and the dynamic dependency among agents (depicted as one arrowed line connecting two agents with the addition of a label which represents the goal on which there is the dependency and another arrowed dotted line with the sign's label connecting an agent to the arrowed plain line that can be deleted or added by this agent).

Example 7. Considering a Grid composed by the nodes of Figure 1 and the dependence network of Example 6, we can form the following dynamic dependence network $DDN = \langle A, G, dyndep^{-}, dyndep^{+}, \geq \rangle$, depicted in Figure 5:

1. Agents $A = \{n_1, n_2, n_3, n_4, n_5, n_6\}$;
2. Goals $G = \{g_1, g_2, g_3, g_4, g_5, g_6\}$;
3. $dep(\{n_1\}, \{n_2\}) = \{\{g_1\}\}$: agent n_1 depends on agent n_2 to achieve the goal $\{g_1\}$: to store the file *comp.log*;
   $dep(\{n_2\}, \{n_3\}) = \{\{g_2\}\}$: agent n_2 depends on agent n_3 to achieve the goal $\{g_2\}$: to run the file *mining.mat*;
   $dep(\{n_3\}, \{n_1\}) = \{\{g_5\}\}$: agent n_3 depends on agent n_1 to achieve the goal $\{g_5\}$: to store the file *satellite.jpg*;
   $dep(\{n_4\}, \{n_6\}) = \{\{g_3\}\}$: agent n_4 depends on agent n_6 to achieve the goal $\{g_3\}$: to run the file *results.mat*;

dep$(\{n_6\},\{n_5\}) = \{\{g_4\}\}$: agent n_6 depends on agent n_5 to achieve the goal $\{g_4\}$: to store the file *satellite.mpeg*;
dyndep$^-(n_5,\{n_3\},\{n_6\}) = \{\{g_6\}\}$: agent n_5 does not depend on agent n_3 to achieve the goal $\{g_6\}$ (to have the authorization to open the file *dataJune.mat*), if it is deleted by agent n_6;
dyndep$^+(n_5,\{n_4\},\{n_6\}) = \{\{g_6\}\}$: agent n_5 depends on agent n_4 to achieve the goal $\{g_6\}$ (to have the authorization to open the file *dataJune.mat*), if it is created by agent n_6;

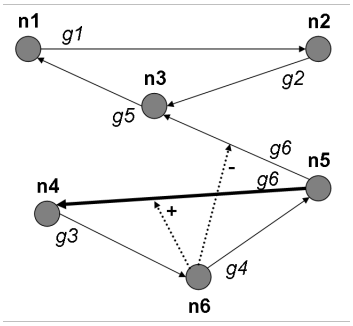

Fig. 5. Dynamic Dependence Network of Example 7.

Example 7 presents the dynamic dependence network of the Grid scenario. We can note that in this network each agent has its associated role since all the nodes are grey ones. Suppose to have a Grid network composing a virtual organization where the local VO administrator is agent n_6. Agent n_6 has delegated the power to give the authorization to access to the files of the VO to agent n_3 but now, since, for example, this node became not safe, the VO administrator has to delegate this power to another node and it chooses node n_4. The dynamic dependence network reflects these actions and thus we have one dynamic dependency for the deletion and another one for the addition.

4.2 Coalitions in Dynamic Dependence Networks

In a multiagent system, we can characterize three different notions of coalitions. A coalition can be defined in dependence networks, based on the idea that to be part of a coalition, every agent has to contribute something, and has to get something out of it. Roughly speaking, a coalition can be formed when there is a cycle of dependencies (the definition of coalitions is more complicated due to the fact that an agent can depend on a set of agents, see below). We show how dependence networks can be used in the requirements analysis for coalitions' evolution, by assuming that goals are maintenance goals rather than achievement goals, which gives us automatically a longer term and more dynamic perspective.

A coalition can be represented by a set of dependencies, represented by $C(a,B,G)$ where a is an agent, B is a set of agents and G is a set of goals. Intuitively, the coalition

agrees that for each $C(a, B, G)$ part of the coalition, the set of agents B will see to the goal G of agent a. Otherwise, the set of agents B may be removed from the coalition or be sanctioned. The three notions of coalition defined below make a distinction between the coalitions which cannot be attacked by the others with addition or removal of dynamic dependencies and thus which are actually formed, vulnerable coalitions of which the existence can be destroyed by the deletion of dynamic dependencies and, finally, potential coalitions, those coalitions which can be formed depending on additions and deletions of dynamic dependencies.

Definition 8 (Coalition).
Let A be a set of agents and G be a set of goals. A coalition function is a partial function $C : A \times 2^A \times 2^G$ such that $\{a \mid C(a, B, G)\} = \{b \mid b \in B, C(a, B, G)\}$, the set of agents profiting from the coalition is the set of agents contributing to it.

Let $\langle A, G, \textit{dyndep}^-, \textit{dyndep}^+, \geq \rangle$ be a dynamic dependence network, and dep the associated static dependencies.

1. *A coalition function C is a coalition if $\exists a \in A, B \subseteq A, G' \subseteq G$ such that $C(a, B, G')$ implies $G' \in \textit{dep}(a, B)$. These coalitions which cannot be destroyed by addition or deletion of dependencies by agents in other coalitions.*
2. *A coalition function C is a vulnerable coalition if it is not a coalition and $\exists a \in A, D, B \subseteq A, G' \subseteq G$ such that $C(a, B, G')$ implies $G' \in \cup_D \textit{dyndep}^-(a, B, D)$. Coalitions which do not need new goals or abilities, but whose existence can be destroyed by removing dependencies.*
3. *A coalition function C is a potential coalition if it is not a coalition or a vulnerable coalition and $\exists a \in A, D, B \subseteq A, G' \subseteq G$ such that $C(a, B, G')$ implies*

$$G' \in \cup_D(\textit{dyndep}^-(a, B, D) \cup G' \in \textit{dyndep}^+(a, B, D))$$

Coalitions which could be created or which could evolve if new abilities or goals would be created by agents of other coalitions on which they dynamically depend.

Example 8. Example 7 presents two different coalitions. On the one hand, we have a *real* coalition composed by agents n_1, n_2 and n_3. On the other hand, we have a potential coalition, such as a coalition which could be formed if agent n_6 really performs the dynamic addition making agent n_5 dependent on agent n_4.

These three notions of coalition represent the constraints for coalitions based on the *dynamic dependency modeling*. The graphical representation of the coalition model is depicted in Figure 6 which describes coalitions (depicted as sets of agents and dependencies included in a dotted circle) and vulnerable and potential coalitions (depicted as sets of agents and dependencies in a circle in which one or more of these dependencies can be added or deleted by another agent with a labeled dynamic dependency). There are various further refinements of the notion of coalition. For example, Boella et al. [5] look for minimal coalitions. In this paper we do not consider these further refinements.

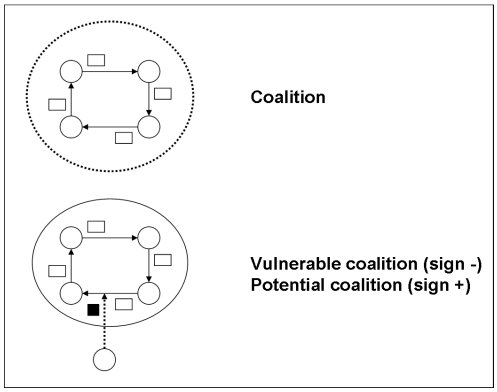

Fig. 6. The legend of the graphical representation of the modeling activities of *dynamic dependency* representing coalitions, potential coalitions and vulnerable coalitions.

5 Conditional Dependency Modeling

In this section, we answer to the subquestions *how to introduce obligations in dependence networks* and *how to define new constraints for the coalitions' representation for this new kind of networks*.

Normative multiagent systems are "sets of agents (human or artificial) whose interactions can fruitfully be regarded as norm-governed; the norms prescribe how the agents ideally should and should not behave. [...] Importantly, the norms allow for the possibility that actual behavior may at times deviate from the ideal, i.e., that violations of obligations, or of agents' rights, may occur" [13]. An obligation is a requirement which must be fulfilled to take some course of action, whether legal or moral. The notion of conditional obligation with an associated sanction is the base of the so called regulative norms. Obligations are defined in terms of goals of the agent and both the recognition of the violation and the application of the sanctions are the result of autonomous decisions of the agent. The association of obligations with violations or sanctions is inspired by Anderson's reduction of deontic logic to alethic logic [1].

A well-known problem in the study of deontic logic is the representation of contrary-to-duty structures, situations in which there is a primary obligation and what we might call a secondary obligation, coming into effect when the primary one is violated [24]. A natural effect coming from contrary-to-duty obligations is that obligations pertaining to a particular point in time cease to hold after they have been violated since this violation makes every possible evolution in which the obligation is fulfilled inaccessible. A classical example of contrary-to-duty obligation is given by the so called "gentle murder" by Forrester [17] which says "do not kill, but if you kill, kill gently".

The introduction of norms in dependence networks to present a new modeling activity is based on the necessity of designing systems based on norms, particularly obligations. An example of these real applications is due to the introduction of obligations in virtual Grid-based organizations [38] where obligations, as shown in Section 2, are used to enforce the authorization decisions. Our model introduces obligations and as-

sociates to the violation of these obligations, sanctions and secondary obligations. This is a new design model since, in approaches like [38], obligations are considered simply as tasks that have to be fulfilled when an authorization is accepted/denied while in approaches like [23], the failure in fulfilling the obligation incurs a sanction but there is no secondary obligation.

The first step toward the introduction of obligations directly in dependence networks is to refine the two notions of goal introduced in Section 3. Physical goals are those goals proper of the agent, e.g., in the Grid scenario these are the personal goals of the users of the system, while institutional goals represent those goals associated to a particular role and not to a single agent, e.g., in the Grid scenario, a VO member node has the goal to obtain an authorization to access to a particular file of another node. The introduction of obligations underlines the necessity to introduce a new kind of goal, the normative goal. These goals originate from norms and they represent the obligation itself. We define a new set of normative concepts, based on Boella et al. [4] model of obligations, and we group them in a new view, called the normative view. The normative view is composed by a set of norms N and three main functions, $oblig$, $sanct$ and ctd representing obligations, sanctions and contrary-to-duty obligations. A portion of our metamodel concerning some of the main concepts is shown the UML class diagram of Figure 12.

Definition 9 (Normative View).
Let the institutional view $\langle RL, IF, RG, X, igoals : RL \rightarrow 2^{RG}, iskills : RL \rightarrow 2^{X}, irules^{1} : 2^{X} \rightarrow 2^{IF} \rangle$*, the normative view is a tuple* $\langle RL, RG, N, oblig, sanct, ctd \rangle$ *where:*

- RL *is a set of roles,* RG *is a set of institutional goals,* N *is a set of norms;*
- *the function* $oblig : N \times RL \rightarrow 2^{RG}$ *is a function that associates with each norm and role, the institutional goals the agent must achieve to fulfill the norm. Assumption:* $\forall n \in N$ *and* $rl \in RL$, $oblig(n, rl) \in power(\{rl\})$[2].
- *the function* $sanct : N \times RL \rightarrow 2^{RG}$ *is a function that associates with each norm and role, the institutional goals that will not be achieved if the norm is violated by role* rl*. Assumption: for each* $B \subseteq RL$ *and* $H \in power(B)$ *that* $(\cup_{rl \in RL} sanct(n, rl)) \cap H = \emptyset$.
- *the function* $ctd : N \times RL \rightarrow 2^{RG}$ *is a function that associates with each norm and role, the institutional goals that will become the new institutional goals the role* rl *has to achieve if the norm is violated by* rl*. Assumption:*$\forall n \in N$ *and* $rl \in RL$, $ctd(n, rl) \in power(\{rl\})$.

In Figure 7 the new conceptual metamodel is provided. In this enlarged version of the conceptual metamodel the notions of obligation, sanction and secondary obligation are added.

To model obligations, we introduce a set of norms, we associate with each norm the set of agents that has to fulfill it, and for each norm we represent how to fulfill

[1] $irules$ associate sets of institutional actions with the sets of institutional facts to which they lead.

[2] Power relates each role with the goals it can achieve.

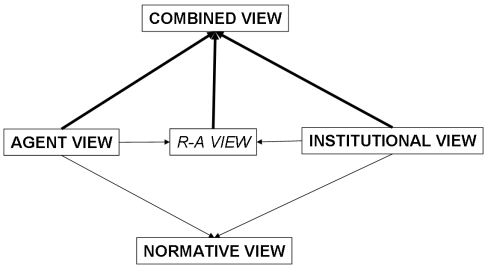

Fig. 7. The new conceptual metamodel.

it, and what happens when it is not fulfilled. In particular, we relate norms to goals in the following two ways. First, we associate with each norm n a set of institutional goals $oblig(n) \subseteq RG$. Achieving these normative goals $oblig(n)$ means that the norm n has been fulfilled; not achieving these goals means that the norm is violated. We assume that every normative goal can be achieved by the group, i.e., the group has the power to achieve it. Second, we associate with each norm a set of institutional goals $sanct(n) \subseteq RG$ which will not be achieved if the norm is violated (i.e., when the goals resulted from the norm are not achieved) and it represents the sanction associated with the norm. We assume that the group of agents does not have the power to achieve these goals. Third, we associate with each norm (called primary obligation) another norm (called secondary obligation) represented as a set of institutional goals $ctd(n) \subseteq RG$ that has to be fulfilled if the primary obligation is violated.

Current work on normative systems' formalizations is declarative in nature, focused on the expressiveness of the norms, the definition of formal semantics and the verification of consistency of a given set. Our approach to norms, using the methodology of dependence networks, is different and is based on the definition of conditional dependence networks. Our aim is not to present a new theorem that, using norms semantics, checks whether a given interaction protocol complies with norms. We are more interested in considering, in the context of requirements analysis, how agents' behaviour is effected by norms and in analyzing how to constraint the design of coalitions' evolution thanks to a normative system. There are two main assumptions in our approach. First of all we assume that norms can sometimes be violated by agents in order to keep their autonomy. The violation of norms is handled by sanctions and contrary-to-duty mechanisms. Second, we assume that, from the institutional perspective, the internal state of the external agents is neither observable nor controllable but the institutional state or public state of the external agents is note since linked to the role associated to the external agent and it can be changed by the agents having this power. Thus, we cannot avoid a forbidden action associated to a goal by a particular rule and we cannot impose an obligatory action in the goals of the agents.

In Section 4, we introduce dynamic dependence networks as a development of the model of dependence networks. In dynamic dependence networks, an agent creates the dependency either creating the obligation, i.e., he creates a new institutional goal for another agent, or creating the power to achieve a goal. In this section, we define a new

modeling activity, called *conditional dependency modeling*, to support the early and late requirements analysis of a system representing obligations and, in particular, sanctions and contrary-to-duty obligations. Conditional dependence networks are defined as follows:

Definition 10 (Conditional Dependence Networks (CDN)).
A conditional dependence network is a tuple $\langle A, G, cdep, odep, sandep, ctddep \rangle$ *where:*

- A *is a set of agents;*
- G *is a set of goals;*
- *cdep* : $2^A \times 2^A \rightarrow 2^{2^G}$ *is a function that relates with each pair of sets of agents all the sets of goals on which the first depends on the second.*
- *odep* : $2^A \times 2^A \rightarrow 2^{2^G}$ *is a function representing a dependency based on obligations that relates with each pair of sets of agents all the sets of goals on which the first depends on the second.*
- *sandep* $\subseteq (OBL \subseteq (2^A \times 2^A \times 2^{2^G})) \times (SANCT \subseteq (2^A \times 2^A \times 2^{2^G}))$ *is a function relating obligations to the dependency which represent their sanctions. Assumption:* $SANCT \in$ *cdep and* $OBL \in$ *odep.*
- *ctddep* $\subseteq (OBL_1 \subseteq (2^A \times 2^A \times 2^{2^G})) \times (OBL_2 \subseteq (2^A \times 2^A \times 2^{2^G}))$ *is a function relating obligations to the dependency which represent their secondary obligations. Assumption:* $OBL_1, OBL_2 \in$ *odep and* $OBL_1 \cap OBL_2 = \emptyset$.

The graphical representation of the model of the *conditional dependency modeling* activity is built following the legend of Figure 8 which describes the obligation-based dependency (depicted as a striped arrowed line), the obligation-based dependency with the associated sanction expressed as conditional dependency (depicted as a striped arrowed line representing the obligation connected to a common arrowed line representing the sanction by a striped line) and the obligation-based dependency with the associated secondary obligation (depicted as a striped arrowed line representing the primary obligation connected to another striped arrowed line representing the secondary obligation by a striped line). The two functions *ctddep* and *sandep* are graphically represented as the striped line connecting the obligation to the sanction or to the secondary obligation.

Example 9. Considering Grid's nodes of Example 7, depicted in Figure 5, we can add two constraints for the requirements analysis phase under the form of obligations and we can build the following conditional dependence network $CDN = \langle A, G,$ *cdep*, *odep*, *sandep*, *ctddep*$\rangle$, depicted in Figure 9:

1. Agents $A = \{n_1, n_2, n_3, n_4, n_5, n_6\}$;
2. Goals $G = \{g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8\}$;
3. *cdep*$(\{n_1\}, \{n_2\}) = \{\{g_1\}\}$: agent n_1 depends on agent n_2 to achieve the goal $\{g_1\}$: to store the file *comp.log*;
   dep$(\{n_2\}, \{n_3\}) = \{\{g_2\}\}$: agent n_2 depends on agent n_3 to achieve the goal $\{g_2\}$: to run the file *mining.mat*;
   dep$(\{n_3\}, \{n_1\}) = \{\{g_5\}\}$: agent n_3 depends on agent n_1 to achieve the goal $\{g_5\}$: to store the file *satellite.jpg*;

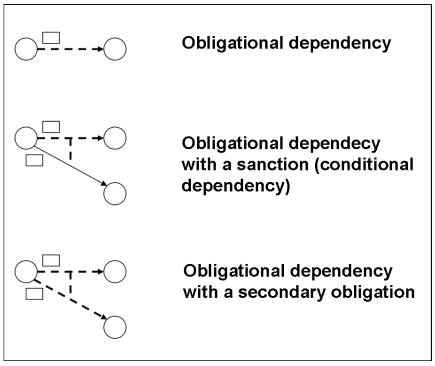

Fig. 8. The legend of the graphical representation of the modeling activity of *conditional dependency*.

dep$(\{n_4\},\{n_6\}) = \{\{g_3\}\}$: agent n_4 depends on agent n_6 to achieve the goal $\{g_3\}$: to run the file *results.mat*;
dep$(\{n_6\},\{n_5\}) = \{\{g_4\}\}$: agent n_6 depends on agent n_5 to achieve the goal $\{g_4\}$: to store the file *satellite.mpeg*;
dep$(\{n_5\},\{n_4\}) = \{\{g_6\}\}$: agent n_5 depends on agent n_4 to achieve the goal $\{g_6\}$: to have the authorization to open the file *dataJune.mat*;
odep$(\{n_2\},\{n_1\}) = \{\{g_7\}\}$: agent n_2 is obliged to perform goal $\{g_7\}$ concerning agent n_1 : to run the file *mining.mat* with the highest priority;
odep$(\{n_4\},\{n_5\}) = \{\{g_8\}\}$: agent n_4 is obliged to perform goal $\{g_8\}$ concerning agent n_5 : to share results of the running of file *dataJune.mat* with agent n_5;
odep$(\{n_4\},\{n_6\}) = \{\{g_8\}\}$: agent n_4 is obliged to perform goal $\{g_8\}$ concerning agent n_6 : to share results of the running of file *dataJune.mat* with agent n_6;
sandep$\{((\{n_2\},\{n_1\}) = \{\{g_7\}\},(\{n_1\},\{n_2\}) = \{\{g_1\}\})\}$;
ctddep$\{((\{n_4\},\{n_5\}) = \{\{g_8\}\},(\{n_4\},\{n_6\}) = \{\{g_8\}\})\}$;

Example 9 shows the subsequent step after the deletion and the insertion of the two dynamic dependencies of Example 7. In this situation, following the definition of coalition, we can imagine to have two local coalitions composing a virtual organization, the first one composed by nodes n_1, n_2, n_3 and the other composed by nodes n_4, n_5 and n_6. Since these two subsets of the virtual organization have to work with a good cohesion then it is possible to insert some constraints, made clear by obligations. The first obligation consists in giving the highest priority to, for example, a computation for an agent composing the same local coalition as you. This first obligation is related to a sanction if it is violated. This link is made clear by the function *sandep* and it represents the deletion of a dependence concerning a goal of the agent that has to fulfill the obligation. The second obligation, instead, is related to a secondary obligation and it means that the agent has to share the results of a computation with a member of its local coalition but, if it does not fulfill this obligation then it has to share these results with another member of the local coalition.

In this new kind of network, if a goal, set by an obligation, is not fulfilled then the conditional dependency related to this obligation has two possible developments: if a

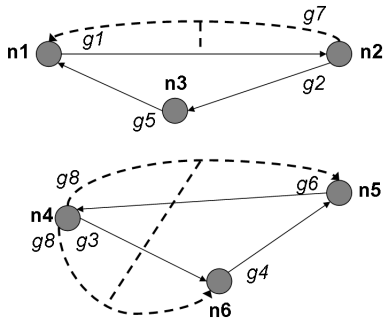

Fig. 9. Conditional Dependence Network of Example 9.

sanction is associated to the norm, a goal cannot be achieved and thus the conditional dependency related to that goal has to be deleted or, if a contrary-to-duty obligation, which means a secondary obligation, is associated to the norm then the conditional dependency related the goals set by this secondary obligation has to be added. We represent obligations, sanctions and contrary-to-duty obligations as tuples of dependencies related to each other. An obligation is viewed as a particular kind of dependency and it is related to other dependencies: dependencies due to sanctions and dependencies due to secondary obligations. In the first case, we have that sanctions are common dependencies, already existing inside the system that, because of their connection with the obligation, can be deleted. In particular, if the obligation is not fulfilled, then the dependency related to the obligation with the role to be its sanction is deleted. In the second case, instead, a primary obligation is related to a number of secondary obligations. A graphical representation of the evolutions of conditional dependence networks is provided in Figure 10:

In the first case, if the obligation is fulfilled and it is linked to a sanction then the obligation can be removed and also the connection among the obligation and the sanction. The only dependency that remains in the network is the one related to the sanction that passes from being a conditional dependency to a common dependency. If the obligation is not fulfilled then it is deleted and the deletion involves also the conditional dependency representing the sanction. The sanction consists exactly in the deletion of this conditional dependency. In the second case, if the obligation is fulfilled and it is linked to a secondary obligation then the obligation is deleted and also the secondary obligation is deleted since there is no reason to already exists. If the obligation, instead, is not fulfilled then the primary obligation is deleted but the secondary obligation not. Note that in Figure 10 are depicted only the conditional dependencies and the obligational dependencies and not all the other kinds of possible dependencies of the network.

Two case studies: transactions and personal norms. In this section, we analyze two particular case studies using our representation of obligations. The first one consists in transactions. A transaction is an agreement or communication carried out between separate entities, often involving the exchange of items of value, such as information, goods,

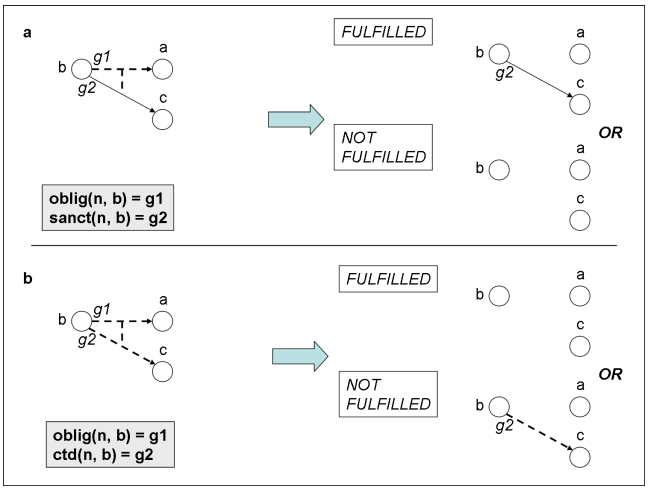

Fig. 10. The evolution of conditional dependence networks.

services and money. This is the basic idea underlying norm emergence. Let us consider the case of two agents a and b, where a is the buyer and b is the seller. If we consider two goals such as $g1$: book sent by the seller b to the buyer a and $g2$: money transferred from the buyer a to the seller b, we have the dependence network depicted in Figure 11-(b). The two agents depend on each other to achieve their goals, the seller is waiting for its payment and the buyer is waiting for its good. When introduced, our representation of obligations allows to arrive to a very simplified version of the network in which each agent depends on itself to not violate the obligation. The dependence network derived after the norm creation is much more simpler than the previous one representing however the same concepts. This simplified version of the network, representing obligations, can be used for the requirements analysis phases, allowing to individuate in a simpler way the obligation present in it, without the necessity to take into account all the sets of dependencies on goals of the network.

The second case study makes more explicit this necessity to simplify the dependence network with the aim to individuate the obligations in the case of personal norms. Everybody's life is regulated by personal norms like *not kill* and *not leave trash on the roads*. These norms are referred to every person and it seems that everyone depends on the others to achieve these goals that can be represented as goals of the whole society. It is similar to the social delegation cycle: do not do the others what you do not want them to do to you. In this case, we can represent the dependence network as a full connected graph since every agent depends on all the other agents, for example to not be killed. The simplification brought by the representation of obligations is relevant, as can be seen in Figure 11-(a).

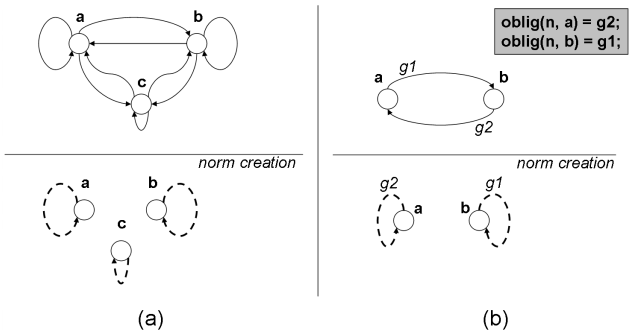

Fig. 11. Case studies: personal norms and transactions.

5.1 Coalitions in Conditional Dependence Networks

In this section, we answer to the subquestion: *What constrains are set by obligations in the conditional dependency modeling concerning coalitions*. In Section 4, we present three different kinds of coalitions: existing coalitions composed by common dependencies, vulnerable coalitions composed by one or more arcs linked to a dynamic dependency of removal and, finally, potential coalitions composed by one or more arcs linked to a dynamic dependency of addition. The new kind of dependence networks, conditional dependence networks, has to be taken into account when a system is described in terms of coalitions. This means that coalitions, vulnerable coalitions and potential coalitions can change depending on the conditional dependencies set by the obligations of the system. A coalition has to consider also sanctions and secondary obligations, according to these constraints:

Definition 11 (Constraints for Conditional Dependency Modeling). *Let A be a set of agents and G be a set of goals. A coalition function is a partial function $C \subseteq A \times 2^A \times 2^G$ such that $\{a \mid C(a, B, G)\} = \{b \mid b \in B, C(a, B, G)\}$, the set of agents profiting from the coalition is the set of agents contributing to it.*

Introducing conditional dependence networks, the following constraints arise:

- $\forall(dep_1, dep_2) \in$ *sandep*, $dep_2 \notin C$ *if and only if* $dep_1 \notin C$. *If the obligation, associated to the dependency dep_1 is not part of the coalition C then also the sanction dep_2 associated to the obligation is not part of the coalition C. If the obligation, associated to the dependency dep_1 is part of the coalition C then also the sanction dep_2 associated to the obligation is part of the coalition C.*
- $\forall(dep_1, dep_2) \in$ *ctddep*, $dep_2 \in C$ *if and only if* $dep_1 \notin C$. *If the primary obligation, associated to the dependency dep_1 is not part of the coalition C then the secondary obligation dep_2 is part of the coalition C. If the primary obligation, associated to the dependency dep_1 is part of the coalition C then the secondary obligation dep_2 is not part of the coalition C.*

Example 10. Let us consider the conditional dependence network of Example 9. Applying these constraints, we have that if the obligation on goal g_7 is fulfilled then the

local coalition composed by agents n_1, n_2 and n_3 already exists since the dependency associated to the sanction is not deleted. If the obligation on goal g_7 is not fulfilled then the obligation is deleted but also the sanction is deleted and the coalition does not exist any more. Concerning the second local coalition, if the obligation is fulfilled then both the primary and the secondary obligation are removed but if the primary obligation is not fulfilled then the secondary obligation is part of the local coalition composed by agents n_4, n_5 and n_6.

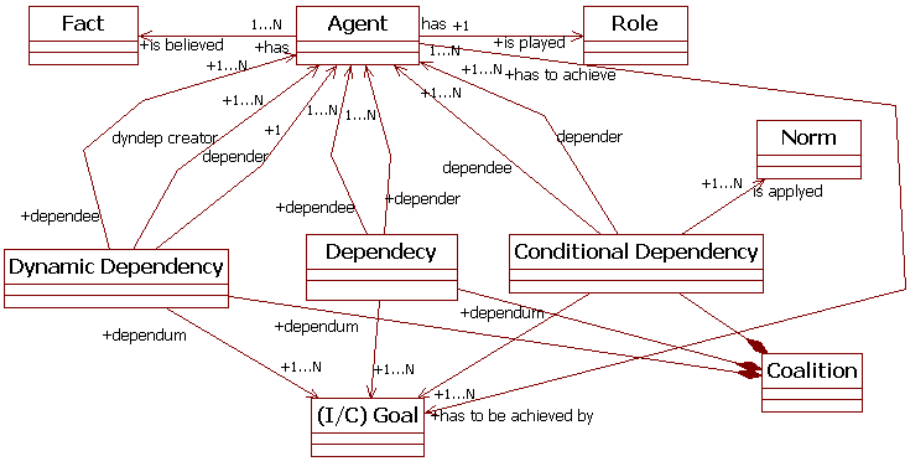

Fig. 12. The UML class diagram specifying the main concepts of the metamodel.

5.2 Regulation of Stability

In game theoretical approaches [28], stability may be taken into account when distributing the payoff of the coalition among its members. Roughly speaking, payoffs should be divided in a fair way to maintain stability. The core, for example, provides a concept of stability for coalitional games and a payoff is in the core only if no coalition has an incentive to break off from the grand coalition and form its own group. Other approaches of the same kind are provided by the other solution concepts such as the Shapley value and the nucleous. Given a previously formed coalitional configuration, game theory usually concentrates on checking its stability or its fairness and on the calculation of the corresponding payments. But game theory rarely takes into consideration the special properties of a multi-agent environment such as, for example, goal-based agents. Coalitions change dynamically due to rapid changes in the tasks and resource availability, and therefore relying on the initial configurations is misleading.

In this section, we present a first step toward the definition of a notion of stability for coalitions individuated in the context of one of our three modeling activities.

The importance of the definition of a notion of stability for the modeling analysis, particularly for the requirements analysis phases, is related to the issues of security and efficiency. For example, in the Grid scenario, it is very important to have the guarantee that the two subsets composing the virtual organization are stable in the sense that they represent secure and efficient “group" of nodes with a great internal cohesion. This approach has the aim to present the problem of coalitions' stability from a different point of view respect the point of view presented in game theoretical approaches. The main difference is in the notion of agent used in the model, such as not an agent viewed only as a utility maximizer but a goal-based agent. In this sense, our definition of agent is more complex and with many facets than the agents presented in game theory. Starting from Section 4, where we distinguished among three different kinds of coalitions, we can define coalitions' stability in the following way:

Definition 12 (Coalitions' stability). *A coalition C is called stable if $\exists a \in A, B \subseteq A, G' \subseteq G$ such that $C(a, B, G')$ implies $G' \in dep(a, B)$ and $\neg(\exists a \in A, D, B \subseteq A, G' \subseteq G$ such that $C(a, B, G')$ implies $G' \in \cup_D dyndep^-(a, B, D))$. A coalition is stable if it is formed by dependencies relying its members and there is not the possibility to delete one these dependencies by another agent, inside or outside the coalition itself.*

Conditional dependencies add new possibility to see to the stability of a coalition. In fact, we can claim that one of the main interests of the agents involved in a coalition is to maintain its stability. This maintenance can be achieved using norms such as obligations to regulate the behaviour of the members of the coalition. The use of obligations can follow two different lines and their development is left for future work:

- Obligations to regulate dynamic dependencies: this first kind of obligation is addressed to each member of a coalition with the aim to avoid, imposing a sanction or a secondary obligation, the mining of the stability of a coalition. These kind of obligations are of the type *If an agent, member of a coalition, has the power to delete one or more of the dependencies constituting the coalition itself then it is obliged to not do this deletion.* This norm is addressed only to those agent belonging to the coalition since, as in real cases, it is always possible for an external agent or coalition to attack another coalition with the aim to decrease its influence. This obligation can be linked to sanctions and secondary obligations of different kinds, such as for example the secondary obligation to create another dependency with the aim to strengthen the coalition. It is also possible to impose a sanction to the agent, for example deleting all the dependencies in which it depends on other agents, preventing him to achieve its goals.
- Obligations to regulate agents' behaviour: this second kind of obligations is not related to the dependencies and dynamic dependencies describing the system, but it is addressed to the regulation of the behaviour of the agents depending on their membership to a coalition. These kind of obligations are of the type *If an agent belongs to a coalition then it has to satisfy first those requests coming from the other members of the coalition and, only after, requests coming from outsiders.* These rules aim to strengthen the unity of the coalition and to improve the work inside it.

6 Related work

The related work section is divided into three subsections: 1) works on agent-based software engineering, 2) works on coalition formation and coalitions' evolution taking into account both game theoretical approaches and social networks ones, 3) works on normative multiagent systems and institutions. The second section presents also a number of works devoted to the definition of the notion of stability for coalitions.

6.1 Agent-based software engineering

The idea of focusing the activities that precede the specification of software requirements, in order to understand how the intended system will meet organizational goals, is not new. It has been first proposed in requirements engineering, specifically in Eric Yu's work with his i* model [36]. This model has been applied in various application areas, including requirements engineering, business process re-engineering, and software process modeling. The i* model offers actors, goals and actor dependencies as primitive concepts [35]. The rationale of the i* model is that by doing an earlier analysis, one can capture not only the what or the how, but also the why a piece of software is developed. This, in turn, supports a more refined analysis of system dependencies and encourages a uniform treatment of the system's functional and non-functional requirements. As stated in the introduction and in the paper, the most important example for our model consists in the TROPOS methodology [11] that aspires to span the overall software development process, from early requirements to implementation.

Other approaches to software engineering are those of KAOS [15] which covers only the late requirements phase, GAIA [34] which covers both the late requirements phase and the architectural design, AAII [22] and MaSE [19] which cover the two phases of architectural and detailed design, and AUML [2] which covers only the detailed design. The main difference between these approaches and our approach is in the use at the same time of the normative multiagent paradigm based on both the notion of institution and the notion of obligation, the graphical modeling language based on dependencies among agents and the covering of the very early phases of requirements analysis.

6.2 Coalitions' formation and evolution

One of the most important issues in the field of multiagent systems concerns the description and formalization of coalition formation. Although there were many approaches defining coalition formation, to represent different perspectives. Two representative examples are given by the model of Shehory and Kraus [27] and the one of Sichman and Conte [30][31].

The approach of Shehory and Kraus [27] is based on the assumption that autonomous agents in the multiagent environments may need to cooperate in order to fulfill tasks. They present algorithms that enable the agents to form groups and assign a task to each group, calling these groups coalitions. However, Shehory and Kraus' work considers tasks which are not related to the individual goals of the agents in the coalition and it does not consider the motivations for agents to enter the coalition, nor the dependencies

existing among the agents. They only address cases in which dependencies among tasks are due to competing resources' requirements or execution precedence order.

Sichman [30], instead, introduces a different point of view. He presents coalition formation using a dependence-based approach based on the notion of social dependence introduced by Castelfranchi [14]. This model introduces the notion of dependence situation, which allows an agent to evaluate the susceptibility of other agents to adopt his goals, since agents are not necessarily supposed to be benevolent and therefore automatically adopt the goals of each other. In this dependence-based model, coalitions can be modeled using dependence networks.

A definition of coalitions inspired by dependence networks is given by Boella et al. [5]. The authors represent a potential coalition as a labeled AND-graph of dependencies among agents. These AND-graphs consist of a set of nodes which denotes the agents involved in the coalition and a set of labeled arcs.

Coalitions' stability The work that, to our knowledge, gives a first definition of stability is the paper of Zlotkin and Rosenschein [39]. In a task oriented domain, a coalition can coordinate by redistributing their tasks among themselves. It seems intuitively reasonable that agents in a coalition game should not suffer by coordinating their actions with a larger group. In other words, if you take two disjoint coalitions, the utility they can derive together should not be less than the sum of their separate utilities, at the worst, they could coordinate by ignoring each other. This property is called superadditivity. This work introduce the notion of stability of a coalition using the concept of superadditivity. The stability condition relates to the payoff vector that assigns to each agent a utility. There are three levels of stability conditions: individual, group and coalition rationality. Individual rationality means that no individual agent would like to opt out of the full coalition, group rationality means that the group as a whole would not prefer any other payoff vector over this vector and coalition rationality means that no group of agents should have an incentive to deviate from the full coalition and create a subcoalition for each subset of agents. To ensure stability, they need to find a consensus mechanism that is resistant to any coalition manipulation. Another work on this issue is from Sandholm and Lesser [25]. In this paper, the optimal coalition structure and its stability are significantly affected by the agents algorithms performance profiles and the unit cost of computation.

6.3 Normative multiagent systems and institutions

An example of normative multiagent system introducing obligations has been done by Boella and van der Torre [8] and [3]. In this work, to model obligations they introduce a set of norms, associated with each norm the set of agents that has to fulfill it and what happens when it is not fulfilled. In particular, they relate norms to goals in the following two ways. First, each norm is associated to a set of goals. Achieving these normative goals means that the norm has been fulfilled; not achieving these goals means that the norm is violated. They assume that every normative goal can be achieved by the group, that means that the group has the power to achieve it. The second point is that each norm is associated to another set of goals which will not be achieved if the norm is violated,

this is the sanction associated to the norm. They assume that the group of agents does not have the power to achieve these goals, otherwise they would avoid the sanction.

An approach to the application of the notion of institution to multiagent systems is defined in Sierra et al. [32]. Electronic Institutions (EIs) provide the virtual analogue of human organizations in which agents, playing different organizational roles, interact to accomplish individual and organizational goals. EIs introduce sets of artificial constraints that articulate and coordinate interactions among agents. In this approach, roles are defined as patterns of behavior and are divided into institutional roles (those enacted to achieve and guarantee institutional rules) and non-institutional roles (those requested to conform to institutional rules). Like us, the purpose of their normative rules is to affect the behavior of agents by imposing obligations or prohibitions.

Another approach to EIs is given by Bogdanovych et al. [9]. In this approach they propose the use of 3D Virtual Worlds to include humans into software systems with a normative regulation of interactions. The normative part can be seen as defining which actions require an institutional verification assuming that any other action is allowed. Inside the 3D Interaction Space, an institution is represented as a building where the participants are represented as avatars. Once they enter the building their actions are validated against the specified institutional rules. In the last two works, unlike us, the concept of institution is presented by a practical approach without a formal definition of the concept of institution and a description of its dynamics while they are similar to our one in the establishment of a different level of the organization related to the institution.

The problem of dynamic institutions is treated in Bou et al. [10] as an extension to EIs definition with the capability to decide in an autonomous way how to answer dynamically to changing circumstances through norm adaptation and changes in institutional agents. The assumption for EIs to adapt is that EIs seek specific goals. The paper presents the normative transition function that maps a set of norms into another one. As our approach, agents participating in the system have social interactions mediated by the institution and the consequences of these interactions is a change in the institutional state of an agent. The difference with our approach consists in the definition of the institution as an entity with own goals, the running example given into the paper is that of the institution of the Traffic Regulation Authority with the goal to decrease the number of accidents below a given threshold, and states.

An approach is presented in Vazquez-Salceda et al. [33] where they propose the Organizational Model for Normative Institutions (OMNI) framework. OMNI brings together some aspects from two existing frameworks: OperA and HARMONIA. OperA is a formal specification framework that focuses on the organizational dimension while HARMONIA is a formal framework to model especially highly regulated electronic organizations from an abstract level to the final protocols that implement norms. In OMNI, roles are often dependent on other roles for the realization of their objectives. Societies establish dependencies and power relations between roles, indicating relationships between roles. These relationships describe, like in our approach, how actors can interact and contribute to the realization of the objectives of each other.

7 Conclusions

This paper provides a detailed account of a new requirements analysis model based on the normative multiagent paradigm, following the TROPOS methodology [11]. The paper presents and discusses the requirements analysis phase of systems design. The first part of the paper presents the key concepts of our model dividing them into three submodels, one representing the agents and their mentalistic notions of goals and facts, the second representing the roles and their associated notions of institutional goals and facts and, finally, the third representing the mapping between agents and roles. The second part of the paper presents our graphical representations for the three modeling activities by which our model is composed. The three modeling activities are called *dependency modeling*, *dynamic dependency modeling* and *conditional dependency modeling* and they are based on the notions of institution, obligation, sanction and secondary obligation. The addition of normative concepts as the last ones is a relevant improvement to requirements analysis since it allows first to constraint the construction of the requirements modeling and second to represent systems, as in the Grid scenario, in which there are explicit obligations regulating the behaviour of the components composing it. Moreover, the model is defined also to model the requirements analysis phases in a context in which there is the possible presence of coalitions and we present the first step toward the definition of the notion of coalitions' stability for our modeling activities.

Concerning future work, we are concentrating our efforts on the definition of the notion of coalitions' stability in this model. We are interested in representing the coalitions' evolution process by means of our modeling techniques and in defining more powerful constraints on coalitions with the aim to maintain, thanks to the application of norms, coalitions' stability during this evolution process. In our opinion, this would be a relevant improvement to the studies concerning coalitions' stability because of the application, at the same time, of a social network approach, providing measures and graph-based methods, and a normative multiagent approach, providing mechanisms like social laws and norms. Moreover, our model in its current form is also not suitable for agents requiring advanced reasoning mechanisms for plans, goals and negotiations. Further extensions will be required to the model to address this class of software applications. Finally, we are improving our conditional dependency modeling by adding also the representation of prohibitions.

References

1. A. Anderson. The logic of norms. *Logic et analyse*, 2, 1958.
2. B. Bauer, J. P. Müller, and J. Odell. Agent UML: A formalism for specifying multiagent software systems. *Software Engineering and Knowledge Engineering*, 11(3):207–230, 2001.
3. G. Boella. Obligations and cooperation: two sides of social rationality. In R. Falcone e C. Castelfranchi H.Hexmoor, editor, *Agent Autonomy*, pages 57–78. Kluwer, 2002.
4. G. Boella, P. Caire, and L. van der Torre. Autonomy implies creating one's own norms norm negotiation in online multi-player games. *KAIS*, 18:137–156, 2009.
5. G. Boella, L. Sauro, and L. van der Torre. Strengthening admissible coalitions. In *ECAI*, pages 195–199, 2006.
6. G. Boella, L. van der Torre, and H. Verhagen. Introduction to normative multiagent systems. *Computational and Mathematical Organization Theory*, 12:71–79, 2006.

7. Guido Boella, Leendert van der Torre, and Serena Villata. Social viewpoints for arguing about coalitions. In The Duy Bui, Tuong Vinh Ho, and Quang-Thuy Ha, editors, *PRIMA*, volume 5357 of *Lecture Notes in Computer Science*, pages 66–77. Springer, 2008.
8. Guido Boella and Leendert W. N. van der Torre. Power in norm negotiation. In Ngoc Thanh Nguyen, Adam Grzech, Robert J. Howlett, and Lakhmi C. Jain, editors, *KES-AMSTA*, volume 4496 of *Lecture Notes in Computer Science*, pages 436–446. Springer, 2007.
9. Anton Bogdanovych, Marc Esteva, Simeon J. Simoff, Carles Sierra, and Helmut Berger. A methodology for developing multiagent systems as 3d electronic institutions. In *AOSE*, volume 4951 of *Lecture Notes in Computer Science*, pages 103–117. Springer, 2007.
10. Eva Bou, Maite López-Sánchez, and Juan A. Rodríguez-Aguilar. Adaptation of autonomic electronic institutions through norms and institutional agents. In Gregory M. P. O'Hare, Alessandro Ricci, Michael J. O'Grady, and Oguz Dikenelli, editors, *ESAW*, volume 4457 of *Lecture Notes in Computer Science*, pages 300–319. Springer, 2006.
11. Paolo Bresciani, Anna Perini, Paolo Giorgini, Fausto Giunchiglia, and John Mylopoulos. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.
12. Patrice Caire, Serena Villata, Guido Boella, and Leendert van der Torre. Conviviality masks in multiagent systems. In Lin Padgham, David C. Parkes, Jörg Müller, and Simon Parsons, editors, *AAMAS (3)*, pages 1265–1268. IFAAMAS, 2008.
13. J. Carmo and A.J.I. Jones. Deontic logic and contrary-to-duties. *Handbook of Philosophical Logic*, pages 203–279, 1996.
14. C. Castelfranchi. The micro-macro constitution of power. *Protosociology*, 18:208–269, 2003.
15. A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Sci. Comput. Program.*, 20(1-2):3–50, 1993.
16. J. Ferber, O. Gutknecht, and F. Michel. Fom agents to organizations: An organizational view of multi-agent systems. In *AOSE*, pages 214–230, 2003.
17. J. W. Forrester. Gentle murder, or the adverbial samaritan. *Journal of Philosophy*, 81:193–197, 1984.
18. I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *Int. J. of Supercomputer Applications*, 15, 2001.
19. Juan C. García-Ojeda, Scott A. DeLoach, Robby, Walamitien H. Oyenan, and Jorge Valenzuela. O-mase: A customizable approach to developing multiagent development processes. In *Agent-Oriented Software Engineering VIII, 8th International Workshop*, volume 4951 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2007.
20. D. Grossi. *Desigining Invisible Hancuffs. Formal Investigations in Institutions and Organizations for Multi-agent Systems.* SIKS Dissertation Series 2007-16, PhD Thesis, 2007.
21. A. J. I. Jones and M. Sergot. A formal characterization of institutionalised power. *Logic Journal of IGPL*, 2003.
22. David Kinny, Michael P. Georgeff, and Anand S. Rao. A methodology and modelling technique for systems of bdi agents. In Walter Van de Velde and John W. Perram, editors, *MAAMAW*, volume 1038 of *Lecture Notes in Computer Science*, pages 56–71. Springer, 1996.
23. N. H. Minsky and A. Lockman. Ensuring integrity by adding obligations to privileges. In *ICSE*, pages 92–102, 1985.
24. Henry Prakken and Marek J. Sergot. Contrary-to-duty obligations. *Studia Logica*, 57(1):91–115, 1996.
25. T. W. Sandholm and V. R. Lesser. Coalition formation among bounded rational agents. Technical report, CMPSCI, 1995.
26. L. Sauro. *Formalizing admissibility criteria in coalition formation among goal directed agents*. PhD thesis, University of Turin, 2005.

27. O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artif. Intell.*, 101:165–200, 1998.
28. Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2008.
29. Yoav Shoham and Moshe Tennenholtz. On social laws for artificial agent societies: Off-line design. *Artif. Intell.*, 73(1-2):231–252, 1995.
30. J. S. Sichman. Depint: Dependence-based coalition formation in an open multi-agent scenario. *Artificial Societies and Social Simulation*, 1(2), 1998.
31. Jaime Simão Sichman and Rosaria Conte. Multi-agent dependence by dependence graphs. In *AAMAS*, pages 483–490. ACM, 2002.
32. Carles Sierra, John Thangarajah, Lin Padgham, and Michael Winikoff. Designing institutional multi-agent systems. In Lin Padgham and Franco Zambonelli, editors, *AOSE*, volume 4405 of *Lecture Notes in Computer Science*, pages 84–103. Springer, 2006.
33. J. Vázquez-Salceda, V. Dignum, and F. Dignum. Organizing multiagent systems. *Autonomous Agents and Multi-Agent Systems*, 11(3):307–360, 2005.
34. M. Wooldridge, N. R. Jennings, and D. Kinny. The GAIA methodology for agent-oriented analysis and design. *Autonomous Agents and Multi-Agent Systems*, 3(3):285–312, 2000.
35. E. Yu. Modeling organizations for information systems requirements engineering. In *First IEEE International Symposium on Requirements Engineering*, pages 34–41, 1993.
36. E. Yu. *Modelling Strategic Relationships for Process Reengineering*. PhD thesis, University of Toronto, 1995.
37. F. Zambonelli, N. Jennings, and M. Wooldridge. Developing multiagent systems: The gaia methodology. *IEEE Transactions of Software Engineering and Methodology*, 12:317–370, 2003.
38. Gansen Zhao, David W. Chadwick, and Sassa Otenko. Obligations for role based access control. In *AINA Workshops (1)*, pages 424–431. IEEE Computer Society, 2007.
39. G. Zlotkin and J. S. Rosenschein. Coalition, cryptography, and stability, mechanisms for coalition formation in task oriented domains. In *AAAI*, pages 432–437, 1994.