Conditional Dependence Networks in Requirements Engineering

Guido Boella[1] and Leendert van der Torre[2] and Serena Villata[1]

[1] Dipartimento di Informatica, University of Turin.
{boella,villata}@di.unito.it
[2] Computer Science and Communication, University of Luxembourg.
leendert@vandertorre.com

Abstract. In this paper we present a new model for the requirements analysis of a system. We offer a conceptual model defined following a visual modeling language, called dependence networks. TROPOS uses a visual modeling language called dependence networks in the requirements analysis of a system, and in this paper we propose a new conceptual model extending dependence networks with norms. This improvement allows to define a new type of dependence networks, called conditional dependence networks, representing a new modeling technique for the requirements analysis of a system. Our model, moreover, allows the definition of coalition depending on different kinds of networks. We illustrate our model using the scenario of virtual organizations based on a Grid network.

1 Introduction

The diffusion of software applications in the fields of e-Science and e-Research underlines the problem 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 provided numerous methods and methodologies allowing to treat more complex software systems. One of these methodologies is the TROPOS methodology [7], developed for agent-oriented design of software systems. The intuition of the TROPOS methodology [7] 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 [2], characterized by their own sets of goals, capabilities and beliefs. TROPOS covers five phases of the software development process: early requirements allowing the analysis and modeling of the requirements of the context in which the software system will be inserted, late requirements describing the requirements of the software system, architectural and detailed design of the system and, finally, the code implementation.

The TROPOS methodology [7] 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 [3]. These social concepts are used in the conceptual modeling of multiagent systems, for example in requirements analysis, as well as in formal analysis and agent based social simulation. For example, in the game theoretic approach of Shoham and Tennenholtz [17], social laws are constraints on sets of strategies. In this paper, we propose to add norms, presented thanks to the normative multiagent paradigm, both to the requirements analysis phases and to the conceptual meta-model. This paper addresses the following research question:

– How to apply a normative multiagent approach to the early and late requirements analysis?

The research question beaks down in the following sub-questions: which ontology have to be defined for the normative multiagent requirements engineering model? and how to model sanctions, contrary-to-duty and coalition's stability in dependence networks?.

Our approach is based, following the approach of TROPOS [7], on the semiformal language of visual modeling called dependence networks and it is composed by the following components. First, we present our ontology that defines the set of concepts composing our conceptual metamodel. 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 and obligations, sanctions, secondary obligations and conditional dependencies. Second, our model is defined as 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. Finally, we have a set of rules and constraints to guide the building of the main concepts of the metamodel, e.g. the formation of coalitions and their stability is constrained to the kind of dependencies linking its members. In TROPOS [7], the requirements analysis phase is split in two main phases, the early requirements and the late requirements. In our methodology, these two phases share the same conceptual and methodological approach, thus we refer to them just as requirements analysis. Dynamic dependence networks have been firstly introduced by Caire *et al.* [9] and then treated in Boella *et al.* [5] in which the existence of a dependency depends on the actions of the agents which can delete it.

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. This new kind of networks represent obligations as particular kinds of dependencies and these obligations are related to notions by means of sanctions if the obligation is not fulfilled and contrary to duty when the primary obligation, not fulfilled, actives a secondary obligation. Moreover, we introduce the notion of coalition and we propose to use methods of social order such as obligations and sanctions to efficiently achieve the maintenance of the stability and the cohesion of these groups. Our model is intended to support the requirements specification for high level open interaction system where heterogeneous and autonomous agents may interact.

Our aim is not to present an 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 constrain the modeling 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 means of 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 these agents is note since connected to a role and it can be changed by the other agents. 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 [7], but only the requirements analysis phases which involve dependence networks. Of course, our model is not intended for any type of software. For system software, e.g., a compiler, or embedded software, the operating environment of the system-to-be is an engineering artifact, with no identifiable stakeholders. In such cases, traditional software development techniques may be most appropriate. However, a large and growing percentage of software operates within open, dynamic organizational environments. This paper is organized as follows. Section 2 describes a Grid computing scenario . In Section 3, we present the dependency and the dynamic dependency modeling while in Section 4 we present a new kind of dependence network, called conditional dependence network. Related work and conclusions end the paper.

2 The Grid Scenario

The Grid Computing paradigm provides 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 and linkage of a wide range of data resources. 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. This information is typically established through the concept of a virtual organization (VO) [12]. A virtual organization allows the users, their roles and the resources they can access in a collaborative project to be defined. 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.

As presented by Zhao *et al.* [22], 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 [22], administrators can associate obligations with permissions, and require the fulfillment of the obligations when the permissions are exercised. The general idea of the RBAC 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. 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. 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 fixed 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 of role and its associated permissions, i.e. the role based access control policy needs a design able to assign roles and represents to 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 can model the concept of "local network" in virtual organizations. Finally, the presented modeling activities depict the system using structures similar to the Grid network itself.

3 Dependency and Dynamic Dependency Modeling

Figure 1 shows the ontology on which is based our model 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 1. Roughly, an institution is a structure of social order and cooperation governing the behavior of a set of individuals. Institutions are identified with a social purpose and permanence, with the enforcing of rules governing cooperative human behavior. The Figure depicts, following the legend of Figure 2, the three submodels which group the concepts of our ontology.

Such a decomposition is common in organizational theory, because an organization can be designed without having to take into account the agents that will play a role in it. 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.

As shown in Figure 1, the agent view is composed by concepts like agent, goal and skill or ability and they are represented by means of a social dependence networks in which nodes are the agents and the edges are the representation of goal-based dependencies. The institutional view, instead, is composed by the notion of role and its institutional goals, skills and facts. As for the agent view, also the institutional one is represented by means of a social institutional dependence network representing the norm-based dependency relations between roles. The role assignment view associates

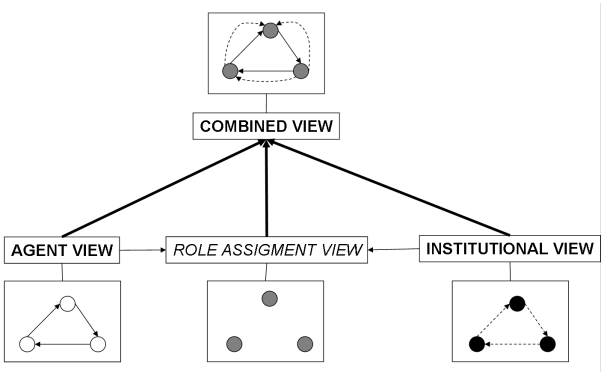

Fig. 1. The conceptual metamodel.

to each agent the roles it plays, depending on the organization in which the agent is playing. All these notions are unified in the combined view where the dependence network represents at the same time both goal-based dependencies and norm-based ones connecting the agents playing roles. In this way, early and late requirements can be based both on agents and on roles. Models are acquired as instances of a conceptual metamodel resting on the concepts presented in the following sections. For more details on the three conceptual submodels, see Boella *et al.* [5] and Boella *et al.* [4].

3.1 Dependence Networks

Figure 2 shows the components of our model. 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.

Dependence networks [18] represent our first modeling activity consisting in the identification of the dependencies among agents and among roles. In the early requirements phase, we represent the domain stakeholders using these networks while in the late requirements phase, the same kind of approach is followed representing the agents of the future system involved in the dependence network. Figure 2-(a) shows the graphical representation of the model obtained following this modeling activity, the *dependency modeling*. The legend 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). The legend considers dependencies among agents but they can be also among roles or agents assigned to roles.

3.2 Dynamic Dependence Networks

Concerning dynamic dependence networks[5], as shown in Figure 2-(a), here we distinguish "negative" dynamic dependencies where a dependency exists unless it is removed

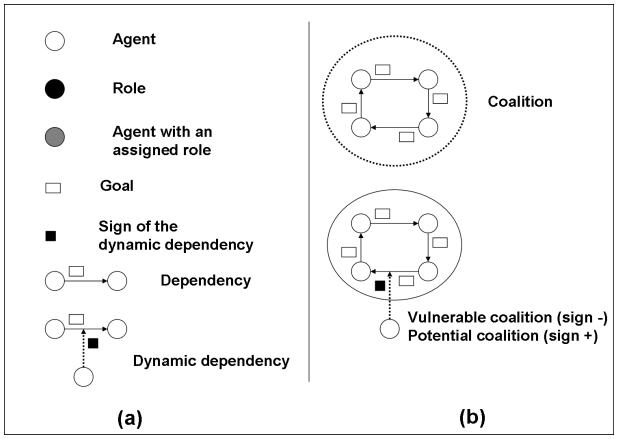

Fig. 2. Legend of the graphical representation of our model.

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. *Dynamic dependency modeling* represents our second modeling activity for requirements analysis. A formal definition of dynamic dependence networks is given in Boella *et al.* [4].

The legend of Figure 2-(a) 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). Figure 3 presents an example of dynamic dependence network on the Grid. In this figure, each node plays a role inside a virtual organization and a number of goal-based dependencies link the nodes to each other, making explicit the fact that a major number of goals can be achieved thanks to cooperation. The dynamic dependency depicted in the figure is related to the institutional goal of obtaining an authorization and it can be removed or added due to the institutional power of the role played by agent n_6. Thanks to our model, we can represent portions or complete virtual organizations, explicating what are the played roles, what are the goals of each node and what are its capabilities, both from the agent point of view and from the institutional one.

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. The graphical representation of coalitions is depicted in Figure 2-(b) 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). Definition 1 makes a distinction between *coalitions* which are actually formed, *vulnerable coalitions* which can be destroyed by

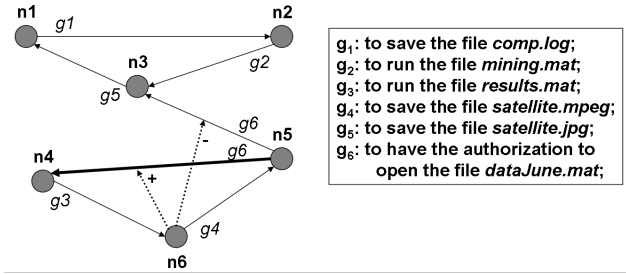

Fig. 3. An example of dynamic dependence network.

the deletion of dynamic dependencies and, *potential coalitions*, which can be formed depending on additions and deletions of dynamic dependencies.

Definition 1 (Coalition). *Let A be a set of agents and G be a set of goals. A coalition function is a partial function $C : A \rightarrow 2^A \times 2^G$ 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 $\langle A, G, dyndep^-, dyndep^+, \geq \rangle$ 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 $G' \in dep(a, B)$. 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 $G' \in \cup_D dyndep^-(a, B, D)$. 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 $G' \in \cup_D (dyndep^-(a, B, D) \cup G' \in dyndep^+(a, B, D))$ 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.*

Figure 3 presents two different coalitions. On the one hand, we have an *actual* 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.

4 Conditional Dependency Modeling

In this section, we answer to the question *how to model sanctions, contrary-to-duty and coalition's stability in dependence networks* by defining the conditional dependency modeling. 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" [6]. In this paper, we represent general regulative norms. 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.

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 [16]. 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 obligations is given by the so called "gentle murder" by Forrester [11] which says "do not kill, but if you kill, kill gently". A contrary-to-duty obligation is not a type of norm. A regulative norm represented by a rule "*if a then obliged x*" is a contrary-to-duty if there is another norm of the kind "*forbidden a*". Note that this is not a property of the norm "*if a then obliged x*" and thus not a type of norm.

4.1 Conditional Dependence Networks

The introduction of norms in dependence networks is based on the necessity to adapt the requirements analysis phases to model norm-based systems. An example of application of this kind consists in the introduction of obligations in virtual Grid-based organizations [22] where obligations, as shown in Section 2, are used to enforce the authorization decisions. On the one hand, in approaches like [22], obligations are considered simply as tasks that have to be fulfilled when an authorization is accepted/denied while, on the other hand, in approaches like [15], the failure in fulfilling the obligation incurs a sanction but there is no secondary obligation.

The introduction of obligations brings us to introduce a new kind of goal, the normative one. These goals originate from norms and they represent the obligation itself. We define a new set of normative concepts, based on Boella *et al.* [2] 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 obligation, sanctions and contrary-to-duty obligations. The UML diagram of Figure 4 provides a unified vision of the presented concepts of the ontology representing our conceptual metamodel.

Definition 2 (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;*

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

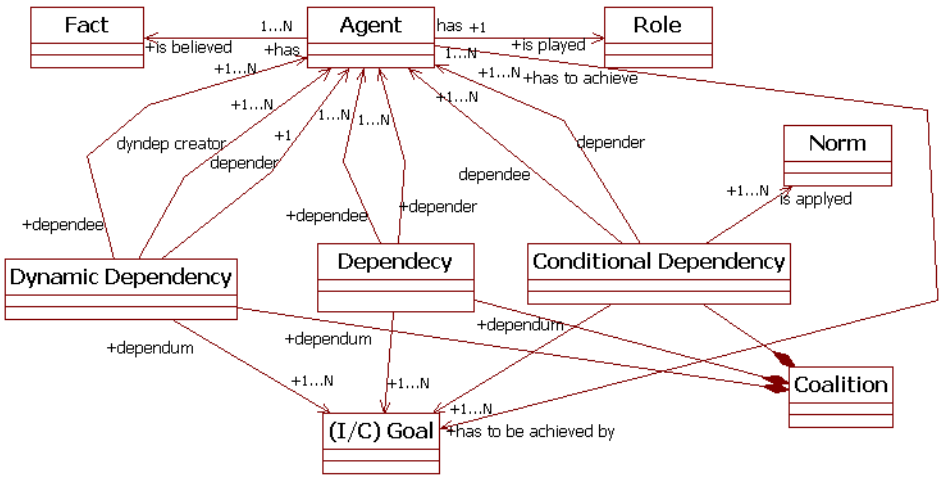

Fig. 4. The UML class diagram specifying the main concepts of the metamodel.

- *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* $rl \in RL$, $ctd(n, rl) \in power(\{rl\})$.

We relate norms to goals following a twofold direction. First, we associate with each norm n a set of institutional goals $oblig(n) \subseteq RG$. Achieving these normative goals 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 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 (primary obligation) another norm (secondary obligation) represented by a set of institutional goals $ctd(n) \subseteq RG$ that have to be fulfilled if the primary obligation is violated.

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

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

Definition 3 (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 and* 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 obligation-based dependency 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 dependencies 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 dependencies which represent their secondary obligations. Assumption:* $OBL_1, OBL_2 \in$ *odep and* $OBL_1 \cap OBL_2 = \emptyset$.

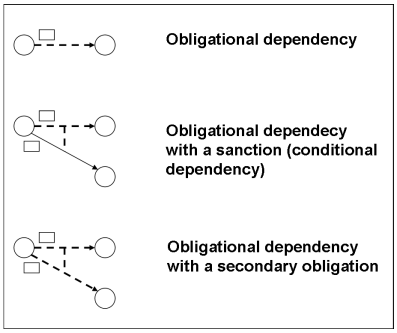

Fig. 5. Legend of the graphical representation of the *conditional dependency modeling*.

Figure 5 gives a graphical representation of the *conditional dependency modeling*. It describes the obligation-based dependency (depicted as a dashed arrowed line), the obligation-based dependency with the associated sanction expressed as conditional dependency (depicted as a dashed arrowed line representing the obligation connected to a common arrowed line representing the sanction by a dashed line) and the obligation-based dependency with the associated secondary obligation (depicted as a dashed arrowed line representing the primary obligation connected to another dashed arrowed line representing the secondary obligation by a dashed line). The two functions *ctddep* and *sandep* are graphically represented as the dashed line connecting the obligation to the sanction or to the secondary obligation.

Example 1. Considering Grid's nodes of Figure 3, we can think to add two constraints under the form of obligations and we build the following conditional dependence network $CDN = \langle A, G, cdep, odep, sandep, ctddep \rangle$ depicted in Figure 6:

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 save 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 save 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 save 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$\{((\{n_4\}, \{n_5\}) = \{\{g_8\}\}, (\{n_4\}, \{n_6\}) = \{\{g_8\}\})\}$;

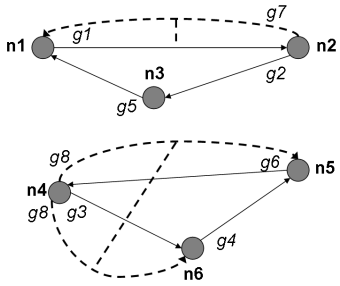

Fig. 6. Conditional Dependence Network of Example 1.

Example 1 is depicted in Figure 6 which shows the network in the step after the deletion and the insertion of the two dynamic dependencies of Figure 3. In Figure 6, following the definition of coalition, we have two coalitions composing, e.g., two local groups of a virtual organization. The first one is composed by nodes n_1, n_2, n_3 and

the other one is 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 means the deletion of a dependency concerning a goal of the agent that has to fulfill the obligation. We represent sanctions as avoiding the achievability of a goal by the punished agent but a sanction would be represented also by imposing something unpleasant, for example an additional goal, on an agent. In this paper, we concentrate the discussion only on the first point and the second one is left for future research. 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 coalition but, if it does not fulfill this obligation then it has to share these results with another member of its coalition.

Figure 7 shows the graphical representation of how an obligation in a conditional dependence network can evolve toward the application of a sanction or of a secondary obligation. 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 can be removed. 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 the deletion of this conditional dependency associated to a goal that the agent would achieve. 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 7 are depicted only the conditional dependencies and the obligational dependencies and not all the other kinds of possible dependencies present in the network.

Summarizing, 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 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. These obligations can be of different kinds depending on the involved agents. For example, we can have a primary obligation linked to two secondary obligations: a first case con involve the same agents, e.g., agent a has to pay agent b for a service but he does not do the payment thus the secondary obligation is to pay to agent b an additional cost, and second case can involve a third agent, e.g., agent a continues to not pay you thus a third agent c is obliged to punish it for example with the deletion of all the services he has to perform for this agent.

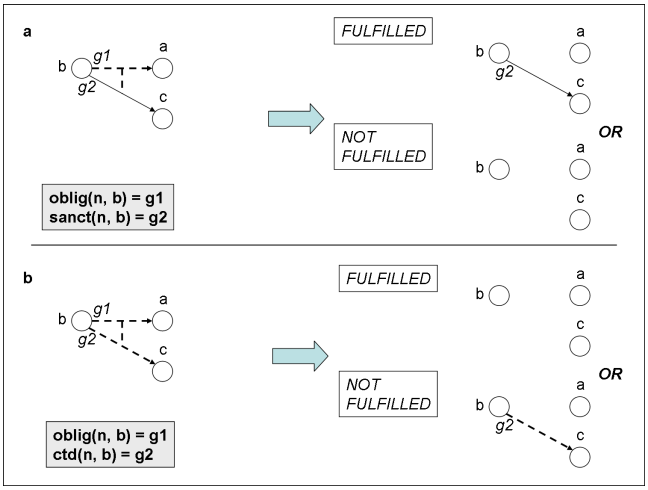

Fig. 7. The evolution of conditional dependence networks.

4.2 Coalitions in Conditional Dependence Networks

In Section 3, we presented a definition of coalition based on the structure of dynamic dependence networks. In these dynamic coalitions we deal with conditional goals but there is not the presence of obligations intended as sets of dependencies linked together by a relation of the kind obligation-sanction or primary obligation-secondary obligation. Conditional dependence networks have to be taken into account when a system is described in terms of coalitions, vulnerable coalitions and potential coalitions since they can change depending on the conditional dependencies set by obligations. A coalition has to consider sanctions and secondary obligations, according to these constraints:

Definition 4 (Constraints for Coalitions in Conditional Dependence Networks). *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 2. Let us consider conditional dependence network of Example 1, depicted in Figure 6. Applying these constraints, we have that if the obligation on goal g_7 is fulfilled then the 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 coalition, if the obligation on goal g_8 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 coalition composed by agents n_4, n_5 and n_6.

5 Related work

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 [21]. The i* model offers actors, goals and actor dependencies as primitive concepts. 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 supports a more refined analysis of system dependencies and encourages a uniform treatment of system's requirements. As stated throughout the paper, the most important inspiration source for our model is the TROPOS methodology [7] that spans the overall software development process, from early requirements to implementation. Other approaches to software engineering are those of KAOS [10], GAIA [20], AAII [14] and MaSE [13] and AUML [1]. The comparison of these works is summarized in Figure 8.

	Early requirements	Late requirements	Architectural design	Detailed design
i*	X	X		
Kaos		X		
GAIA		X	X	
AAII and MaSE			X	X
AUML				X
TROPOS	X	X	X	X

Fig. 8. Comparison among different software engineering methodologies.

The main difference between these approaches and our one consists in the introduction of the notion of obligation with its related concepts of contrary-to-duty and sanction to the requirements analysis and in the graphical modeling language based on dependencies among agents. Moreover, these approaches do not consider the notion of coalition, as group of actors with a common set of goals and the possible constraints on their structure.

6 Conclusions

This paper provides a detailed account of a new requirements analysis model based on the normative multiagent paradigm, following the TROPOS methodology [7]. The paper presents and discusses the early and late requirements phases of systems design [19]. We present the key concepts of the ontology of our methodology, agents, roles, skills, goals, as shown by the UML diagram of Figure 4. We divide our ontology in three submodels: the agent model, the institutional model, and the role assignment model. The modeling activities based on this ontology, the dependency and the dynamic dependency modeling, use a visual language in order to model the stakeholders and their relationships. Moreover, we introduce in the ontology the notion of coalition for dependence networks. The modeling of normative concepts is an improvement to requirements analysis since it allows, first, to constrain the construction of the requirements modeling and, second, to represent systems, as for example Grid-based systems, in which there are explicit obligations regulating the behaviour of the components composing it. We define a new modeling activity, called conditional dependency modeling, to support in the early and late requirements analysis the representation of obligations and contrary-to-duty obligations. This representation is realized as tuples of dependencies related to each other where an obligation is viewed as a particular kind of dependency, related to the dependencies due to sanctions and secondary obligations. Moreover, we model the requirements analysis phases also in a context in which there is the possible presence of coalitions in conditional dependence networks.

Concerning future work, 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. Finally, we are improving our conditional dependency modeling by adding also the representation of prohibitions.

References

1. 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.
2. 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.
3. G. Boella, L. van der Torre, and H. Verhagen. Introduction to normative multiagent systems. *Computational and Mathematical Organization Theory*, 12:71–79, 2006.
4. G. Boella, L. van der Torre, and S. Villata. Changing institutional goals and beliefs of autonomous agents. In Bui et al. [8], pages 78–85.
5. G. Boella, L. van der Torre, and S. Villata. Social viewpoints for arguing about coalitions. In Bui et al. [8], pages 66–77.
6. G. Boella and L. W. N. van der Torre. Regulative and constitutive norms in normative multiagent systems. In D. Dubois, C. A. Welty, and M.-A. Williams, editors, *KR*, pages 255–266. AAAI Press, 2004.

7. P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems*, 8(3):203–236, 2004.
8. T. D. Bui, T. V. Ho, and Q.-T. Ha, editors. *Intelligent Agents and Multi-Agent Systems, 11th Pacific Rim International Conference on Multi-Agents, PRIMA 2008, Hanoi, Vietnam, December 15-16, 2008. Proceedings*, volume 5357 of *Lecture Notes in Computer Science*. Springer, 2008.
9. P. Caire, S. Villata, G. Boella, and L. van der Torre. Conviviality masks in multiagent systems. In L. Padgham, D. C. Parkes, J. Müller, and S. Parsons, editors, *AAMAS (3)*, pages 1265–1268. IFAAMAS, 2008.
10. A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Sci. Comput. Program.*, 20(1-2):3–50, 1993.
11. J. W. Forrester. Gentle murder, or the adverbial samaritan. *Journal of Philosophy*, 81:193–197, 1984.
12. I. T. Foster. The anatomy of the grid: Enabling scalable virtual organizations. In *CCGRID*, pages 6–7. IEEE Computer Society, 2001.
13. J. C. García-Ojeda, S. A. DeLoach, Robby, W. H. Oyenan, and J. 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.
14. D. Kinny, M. P. Georgeff, and A. S. Rao. A methodology and modelling technique for systems of bdi agents. In W. V. de Velde and J. W. Perram, editors, *MAAMAW*, volume 1038 of *Lecture Notes in Computer Science*, pages 56–71. Springer, 1996.
15. N. H. Minsky and A. Lockman. Ensuring integrity by adding obligations to privileges. In *ICSE*, pages 92–102, 1985.
16. H. Prakken and M. J. Sergot. Contrary-to-duty obligations. *Studia Logica*, 57(1):91–115, 1996.
17. Y. Shoham and M. Tennenholtz. On social laws for artificial agent societies: Off-line design. *Artif. Intell.*, 73(1-2):231–252, 1995.
18. J. S. Sichman and R. Conte. Multi-agent dependence by dependence graphs. In *AAMAS*, pages 483–490. ACM, 2002.
19. A. van Lamsweerde. *Requirements Engineering: From System Goals to UML Models to Software Specifications*. Wiley, 2009.
20. 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.
21. E. Yu. Modeling organizations for information systems requirements engineering. In *First IEEE International Symposium on Requirements Engineering*, pages 34–41, 1993.
22. G. Zhao, D. W. Chadwick, and S. Otenko. Obligations for role based access control. In *AINA Workshops (1)*, pages 424–431. IEEE Computer Society, 2007.