FORMALIZING VIRTUAL ORGANIZATIONS

Sergio Esparcia and Estefanía Argente
Grupo de Tecnología Informática - Inteligencia Artificial, Departamento de Sistemas Informáticos y Computación
Universidad Politécnica de Valencia, Camino de Vera s/n, 46022 Valencia, Spain

Keywords: Virtual organizations, Formalization, Multiagent systems.

Abstract: This work presents a formalization of Virtual Organizations, which are designed by means of their structural entities, such as roles, organizational units or norms, and the dynamic entities that change through time like agents and groups. Entities are grouped by means of the Organizational Dimensions, explicitly represented in the proposed formalization. Additionally, a study of existing formalizations of Organization Centered Multiagent Systems is presented.

1 INTRODUCTION

During last years, the development of Multiagent systems (MAS) has turned from an agent centered perspective to an organization centered perspective. Organizations describe system functionality, structure, environment and dynamics. In Organization Centered MAS (OCMAS), the organization exists as an explicit entity of the system (Picard et al., 2009), defined by its designers following a top-down approach. In OCMAS, agents are aware of the organization in which they are participating and they are provided with a representation of it. Agents can use this knowledge to reason about it and to establish relationships and interactions to reach their objectives.

OCMAS can be defined and described by means of a formal approach. To formalize them it is necessary to introduce concepts taken from maths and logic theories, such as LAO (Dignum and Dignum, 2007), whose syntax to define a system follows the temporal logic language CTL (Emerson, 1991). Other proposals not only provide a formal way to describe an OCMAS, but also a language to describe it, such as the proposal in (Grossi et al., 2005), which employs a multimodal propositional logic language to model agent organizations, based on Kripke models. Formal approaches are very useful in order to obtain a clear definition of OCMAS, improving the study and analysis of the different issues regarding them. Additionally, these formalizations are commonly used to check the correctness and integrity of an OCMAS, by means of techniques like model checking (Clarke, 1999).

However, formal approaches are not always able to represent all the concepts that compose an agent organization. Using the current proposals it is not possible to completely define a paradigm for developing agent systems such as Virtual Organizations (VO) (Foster et al., 2001), which are sets of individuals and institutions that need to coordinate resources and services. Thus, they are open systems (Gonzalez-Palacios and Luck, 2007) formed by the grouping and collaboration of heterogeneous entities, and allowing model systems at a high level of abstraction. They include the integration of organizational and individual perspectives and also the dynamic adaptation of models to organizational and environmental changes.

VOs can be structured by means of the Organizational Dimensions (Criado et al., 2009), which should be considered when modeling an organization. These dimensions describe all the entities that compose the organization, distributed by the functionality of the entities that they are providing to it. These dimensions are: *structural*, *functional*, *dynamical*, *environment* and *normative*. Current formal proposals only define a subset of the dimensions and concepts presented in the Organizational Dimensions. Thus, it seems necessary to provide with a formalization that clearly models the Organizational Dimensions, making a clear difference between them.

The objective of this work is to present a formal framework to define a VO, taking the Organizational Dimensions as a basis. The rest of this work is structured as follows: Section 2 describes the Organizational Dimensions. Section 3 describes formal frameworks related with our work. Section 4 presents the *Virtual Organization Formalization* (VOF), a formal

Esparcia S. and Argente E..
FORMALIZING VIRTUAL ORGANIZATIONS.
DOI: 10.5220/0003156500840093
In *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence* (ICAART-2011), pages 84-93
ISBN: 978-989-8425-41-6

framework to define VOs. Section 5 presents a discussion between our proposal and the analyzed frameworks. Finally, section 6 gives our conclusions.

2 ORGANIZATIONAL DIMENSIONS

When modeling an organization, the following dimensions should be taken into account (Criado et al., 2009): (i) *structural*, describing the entities that populate the system; (ii) *functional*, which details the functions, goals and services of the organization; (iii) *dynamical*, which considers the interactions between the elements and their effects; (iv) *environment*, describing the elements that surround the system; and (v) *normative*, which defines the mechanisms used by the society to influence the behavior of its members.

The **Structural Dimension** comprises all the elements of the organization that are independent from the agents that are part of it. Thus, it is based on roles, groups and their patterns of interrelationship (inheritance, compatibility, communication, and so on). Additionally, the topology of the system is established.

The **Functional Dimension** specifies the global goals of the organization, its offered functions and services, the goals followed by different components of the organization and the tasks and plans that must be executed to reach these goals.

The **Dynamical Dimension** specifies how the organization evolves through time, detailing the way in which agents enter and leave it, how they adopt certain roles according to their capabilities and abilities, and how they can participate in the units or groups of the organization where they are admitted. This dimension also details the interactions that take place between internal and external entities.

The **Environment Dimension** describes how agents are connected with other types of entities such as artifacts, applications or resources; and how agents can perceive and act on the environment.

Finally, the **Normative Dimension** determines the set of defined actions and rules to manage the behavior of the members of the organization. Norms are widely used to limit the autonomy inside societies and to solve coordination problems, specially when it is not possible to exercise a total social control.

3 RELATED WORK

Based on different logics and formal methods, some proposals to model OCMAS have been defined, each giving its particular vision and adapting its formalization to the specific kind of system that they are looking to build. In this section, a set of relevant proposals on this field has been reviewed: OperA (Dignum, 2003), LAO (Dignum and Dignum, 2007), Process-Oriented Modeling Framework (POMF) (Popova and Sharpanskykh, 2006), MOISE^{Inst} (Gâteau et al., 2005), MACODO (Haesevoets et al., 2009), PopOrg (da Rocha Costa and Dimuro, 2008) and the proposals from (Grossi et al., 2005) and (Jonker et al., 2007). All these proposals are analyzed following the Organizational Dimensions described in the previous section, in order to check whether they are taking into account the entities and concepts from each dimension.

Table 1 compares these proposals, analyzing the organizational elements that they take into account. Next, we will depict in detail the contents of this table, describing each studied proposal.

OperA proposes an Organizational Model to describe organizations that defines the social, normative, interaction and communicative structures of the society. The *Social Structure* of OperA is related with the Structural Dimension, since it contains roles, groups and dependency relations between roles. Also, its Social Structure is related with the Functional Dimension since it takes into account the objectives associated with roles. The *Normative Structure* is obviously related with the Normative Dimension, as both consider norms. The *Interaction Structure* models the activity of the system, which is considered as the dynamics taken from the Dynamical Dimension. Finally, the *Communicative Structure* manages communication between agents, like interactions in the Dynamical Dimension. Nevertheless, this framework does not model the environment of an OCMAS.

The *logic for agent organizations* (LAO) is an extension of CTL logic. The Functional Dimension is completely represented in LAO, including agents, objectives, groups, the topology of the system (establishing links between agents), and roles, which are represented by means of capabilities and abilities, elements taken from this dimension. LAO additionally defines different states of the world where the system is located (related to the Environment Dimension) and its transitions (related to the Dynamical Dimension). LAO is a very complete proposal, since it takes into account a large subset of the elements of the Organizational Dimensions, but it does not provide a formalization for the Normative Dimension.

The *Process Oriented Modeling Framework* (POMF) is structured by means of four views. The main one is the *process oriented view*, where tasks, processes and workflows are defined. This view in-

Table 1: Comparison between different formal representations.

Organizational concepts								
	OperA	**MOISE**	**PopOrg**	**LAO**	**POMF**	**MACODO**	**Grossi**	**Jonker**
Structural Dimension								
Roles	✓	✓	✓	✓	✓	✓	✓	✓
Groups	✓	✓		✓		✓		
Agents		✓	✓	✓	✓	✓	✓	✓
Relations	✓	✓	✓	✓			✓	✓
Topology				✓				
Functional Dimension								
Capabilities			✓	✓	✓	✓		
Abilities			✓	✓				
Services			✓		✓			
Objectives	✓	✓		✓	✓		✓	
Dynamical Dimension								
Interactions	✓	✓	✓				✓	✓
Dynamics	✓	✓	✓	✓		✓		✓
Environment Dimension								
Environment		✓		✓		✓		✓
Resources					✓			
Normative Dimension								
Norms	✓	✓				✓	✓	
Syntax				CTL	L_{PR}	Z	*Org*	*TTL*
Semantics	LCR			CTL*	T_{PR}		*Org*	

cludes the concept of service from the Functional Dimension, being a workflow divided into processes that are split into tasks. It also includes the resources of the Environment Dimension. The *organization oriented view* includes the role entity, which describes the set of capabilities of the organizational processes in a concrete workflow that are then assigned to agent entities, defined in the *agent view*, where groups of agents are not able to be modeled. Therefore, the organization oriented view is related with both Structural and Functional Dimensions and the agent view is related with the Structural Dimension. Finally, the *performance oriented view* describes the organizational goals, such as the Functional Dimension does. However, POMF does not provide a formalization for neither Dynamical nor Normative Dimensions.

MOISE^{Inst} is composed of four specifications, distributed in a similar way to the Organizational Dimensions. The *Structural Specification* (SS) defines the roles that agents will play, including the relations between them, and an additional structural level named group, where the roles belong to and the interactions are carried out. The SS contains elements from the Structural Dimension, but it does not model the topology of the system. The *Functional Specification* (FS), related with the Functional Dimension, only defines here the goals that the system must achieve. The *Contextual Specification* (CS) defines the different contexts that influence the organizational dynamics and the transitions between them. This specification defines the environment, taken from the Environment Dimension, and its dynamics, just like the Dynamical Dimension does. Unfortunately, the CS does not model the resources populating the environment or the interactions between agents. Finally, the *Normative Specification* (NS) defines the rights and duties of roles and groups inside the organization, which are known as norms in the Normative Dimension. All agents that have adopted a role from the SS compose the *Organizational Entity* (OE), which is the element of the system that controls the dynamic elements of the organization, including agents and all events that they generate, such as their interactions.

The organizational formalization proposed in (Grossi et al., 2005) pursues to represent the organizational structure. This formal method takes the concepts of *role*, establishing relations between them; and *agent* from the Structural Dimension. The roles of the organization are conceived around three basic notions: objectives, norms and information. Objectives are the only elements related with the Functional Dimension that are presented in this proposal; and the Normative Dimension is taken into account using norms. Regarding information, knowledge about the current state of the organization can be given to agents by other agents, so this is a type of interaction from the Dynamical Dimension. Since this proposal is focused on modeling the organization structure, it does not take into account the Environment Dimension.

In (Jonker et al., 2007), a framework for modeling and providing a formal analysis of organizations is defined, based on a generic representation of them

by means of a set of roles. Apart from the roles, this proposal also formalizes two concepts from the Structural Dimension: agents and relations between roles. These relations enable the interactions from the Dynamical Dimension, which is completed by taking into account the dynamics of the organization. One of the main advantages of this work is that it is able to explicitly model the environment of the Environment Dimension, although it does not model environmental resources. The main lack of this formalization is that it does not formalize any concept from the Functional and Normative Dimensions, so designers are not able to model concepts such as objectives and norms.

The MACODO framework is centered on the dynamics of organizations with self-organization concepts, offering a model with concepts related to the Functional Dimension, such as roles (establishing the concept of position, similar to a job offer), contracts of roles (an agreement between an agent and an organization for a concrete position to control the access to an available role), agents (including their context and local environment, which is related to the Environment Dimension), and organizations (groups of agents defined by a set of open role positions and current role contracts). Relations of hierarchy and communication between roles are not considered. A role is described as a set of capabilities, which is the only entity from the Functional Dimension that MACODO takes into account. Since MACODO is focused on self-organization, the dynamics of the system from the Dynamical Dimension, including changes in its context or in its set of agents, are formalized. To control the activities that the organization carries out, MACODO is enhanced with a set of laws, similar to norms from the Normative Dimension. Although MACODO does not model other relevant organizational concepts such as objectives, it deals with elements from all the Organizational Dimensions.

Finally, to manage the structural dynamics of a MAS, PopOrg is a model based on two basic concepts: the *population* of an organization and its *structure*. The population of a MAS is its set of agents, as well as the behaviors and actions (which represent the capabilities and abilities from the Functional Dimension); and the exchange processes (services from the Functional Dimension) that agents are able to carry out. Therefore, the population of a PopOrg organization mainly takes concepts from the Functional Dimension plus agents from the Structural Dimension. Moreover, the structure of the organization is composed by roles and the links between them, which are elements that belong to the Structural Dimension. To relate the population and the structure, PopOrg has a third element called *implementation* that relates the roles with the agents, and the links with the exchange processes. Also, PopOrg stores the different states that the system goes through during its execution. Unfortunately, PopOrg does not model any of the entities from the Environment or Normative Dimensions.

Generally, all the analyzed formalizations present a good approach to define an organization in a formal way. Nevertheless, none of the proposals takes into account all the entities from the Organizational Dimensions, which are not clearly depicted in these frameworks. Thus, it seems interesting to provide with an explicit description of the Organizational Dimensions, which are useful for representing the organization elements. Therefore, in section 4 our proposal to model organizations will be presented, which models an organization clearly defining its dimensions. This proposal also integrates some features taken from some proposals presented in this section.

4 FORMAL DESCRIPTION OF A VIRTUAL ORGANIZATION

In this section the concept of Virtual Organization (VO) will be defined in a formal way, taking into account its organizational dimensions. This formalization, named VOF (*Virtual Organization Formalization*), will be focused on three elements: (i) the Organizational Specification (*OS*), which details the set of static elements of the organization; (ii) the *Organizational Entity* (*OE*), which represents the instantiation of the elements in *OS*; and (iii) the *Organizational Dynamics* (ϕ), which relates elements from *OS* with elements from *OE*.

Definition 1. *A Virtual Organization is defined, at a given time t, as a tuple* $O^t = \langle OS^t, OE^t, \phi^t \rangle$ *where:*

- *OS refers to the Organizational Specification. It is defined as* $OS = \langle SD, FD, ED, ND \rangle$ *where:*
  - *SD is the Structural Dimension.*
  - *FD is the Functional Dimension.*
  - *ED is the Environment Dimension.*
  - *ND is the Normative Dimension.*
- *OE refers to the Organizational Entity, which represents the dynamic elements of the system.*
- ϕ *allows to relate OS with OE, thus defining the Dynamic Dimension, together with the OE.*

The VO will change through time modifying its states, occurred after a change in the environment, and it will change from one state to another by means of a transition. The following subsections will describe in detail these three elements.

4.1 Organizational Specification

The Organizational Specification details the set of 'static' elements of the organization, containing organizational units, roles, norms, and the rest of elements that build the dimensions of a Virtual Organization. The Organizational Structure is composed by: (i) the *Structural Dimension*, which contains roles, organizational units and their relationships; (ii) the *Functional Dimension*, describing objectives, functionalities and services of an organization; (iii) the *Environment Dimension*, which describes the artifacts and workspaces from the environment of the organization; and (iv) the *Normative Dimension*, which defines the norms that rule the system.

4.1.1 Structural Dimension

The *Structural Dimension* describes the components of the system and their relations. It allows defining the static components of an organization, i.e. all the elements that are independent from the entities that are finally executed. In a more specific way, it defines the organizational units and the structural elements, roles and relationships between roles.

Definition 2. *The Structural Dimension (SD) of a Virtual Organization is defined as* $SD = \langle R, OU, Relations \rangle$ *where:*

- *R refers to the roles of the organization.*
- *OU is the set of organizational units.*
- *Relations is a set of relationships, defined as* $Relations = \langle SocialRelations, StructRelations, DimRelations \rangle$ *where:*
  - ***SocialRelations*** *refers to the social relationships between roles, which can be formalized as:*

$$SocialRelations = \begin{cases} inf: & R \rightarrow R \\ col: & R \rightarrow R \\ sup: & R \rightarrow R \\ comp: & R \rightarrow R \end{cases}$$

    where: ***inf*** *(information) refers to the information relations, which allows communications between roles;* ***col*** *(collaboration) allows a role to monitor the activities of other roles;* ***sup*** *(supervision) defines that an agent playing a specific role can transfer or delegate one or some of his objectives to a subordinate role; and* ***comp*** *(compatibility) depicts that an agent playing a specific role can also play another compatible role in the organization at the same time.*

  - ***StructRelations*** *refers to the structural relationships defined by the structure of the organization, which can be formalized as:*

$$StructRelations = \begin{cases} RoleHier: & R \rightarrow R \\ Contains: & OU \rightarrow 2^{OU} \\ Roles: & OU \rightarrow 2^{R} \end{cases}$$

    where: ***RoleHier*** *represents the hierarchy between roles of the organization;* ***Contains*** *defines the topology of the organization by means of relations between organizational units; and* ***Roles*** *defines the roles that are located inside an organizational unit.*

  - ***DimRelations*** *allows relating this dimension with others, through the element OU, and can be formalized as:*

$$DimRelations = \begin{cases} Norms: & OU \rightarrow 2^{N} \\ Services: & OU \rightarrow 2^{S} \\ Goals: & OU \rightarrow 2^{G} \\ Workspaces: & OU \rightarrow 2^{WS} \end{cases}$$

    where: ***Norms*** *defines the norms, described in the normative dimension, which rule an OU;* ***Services*** *relates an OU with the services that contains;* ***Goals*** *describe the objectives that are necessary to be reached inside an OU; and* ***Workspaces*** *details the workspaces (see Definition 4) where an OU can be located.*

Properties of the Relations. The social relation inf is symmetrical, since a role can provide information to a second role, and viceversa; transitive, since agents can build an information chain, and reflexive as an agent can send information to himself. The relations *col* and sup are both asymmetrical, since an agent cannot monitor or supervise the agent which is monitoring or supervising him; reflexive, because an agent can collaborate or supervise himself; and transitive, allowing to create a command chain inside the organization.

The compatibility relation (*comp*) has reflexive and transitive properties, because a role is compatible with itself and a role is compatible with the roles that have a compatibility relation with its compatible roles. It is interesting to notice that the *comp* relation is not symmetrical (e.g. $comp(r_1, r_2)$ not always implies $comp(r_2, r_1)$). For example, the relation $comp(Professor, Teacher)$ is correct, because a professor can work as a teacher in every moment, but a teacher might not be capable of playing the role of professor. Finally, the relations *RoleHier* and *comp* are related, since an agent playing a specialized role is capable of playing its generalized role. Formally:

$$\forall r_1, r_2 \in R : RoleHier(r_1, r_2) \rightarrow comp(r_2, r_1) \quad (1)$$

Let $r_1, r_2 \in R$ be two roles belonging to *OS*. The information, collaboration and supervision relations define the following relations in an implicit way:

$$sup(r_1, r_2) \rightarrow col(r_2, r_1) \quad (2)$$

$$col(r_1, r_2) \rightarrow inf(r_1, r_2) \wedge comp(r_2, r_1) \quad (3)$$

This means that a supervision relation between two agents implies that a supervised agent will collaborate with a supervisor agent to help him to reach his objectives. Also, a collaboration relation between two roles implies that an information link between them exists and the second role of the relation is compatible with the first one.

The relation *Contains* from the *StructRelations* set has the following properties: (i) asymmetrical, since an OU cannot be contained in another OU that contains it; (ii) transitive, because it is considered that an OU contained inside another OU is also contained inside the predecessors of the OU that contains it; and (iii) irreflexive, since an OU cannot contain itself. In a similar way, the *RoleHier* relation has the same properties of the *Contains* relation, because a role cannot have an inheritance relation with itself, the relations between roles are transitive to allow defining a complete role hierarchy and a subordinated role cannot be the supervisor of its supervisor.

Properties of the Entities. Firstly, an organizational unit is contained inside another OU, this implies that the roles from this OU are compatible with those of its predecessor OU. Formally:

$$\forall OU_1, OU_2 \in OU : Contains(OU_1, OU_2) | \forall r_1 \in \quad (4)$$

$$Roles(OU_1) \wedge \forall r_2 \in Roles(OU_2) \rightarrow comp(r_2, r_1)$$

It should be noted that the *Roles* relation is recursive: the roles that an OU offers are not only its own roles, but also these from its predecessor OUs. Formally:

$$\forall o \in OU : \forall r \in Roles(o) \rightarrow r \in Roles(o) \vee \quad (5)$$

$$r \in Roles(o_1) : o \in Contains(o_1)$$

Properties of the OU. The relations between organizational units allow defining three different types of structures of an organization:

- *'hierarchy'*. A hierarchy implies that there is a supervisor role, with supervision relations to all the other members of its same organizational unit (OU). Formally, $\exists r \in Roles(OU) : \forall r_i \neq r \in Roles(OU) \rightarrow sup(r, r)$. If a designer wants to make his system tighter, he can also prohibit communications between subordinated roles.
- *'team'*. In this kind of structure, all roles have coordination relations between them. Formally, it is defined as $\forall r_1, r_2 \in Roles(OU) : col(r_1, r_2)$.
- *'plain'*. This structure establishes information relationships between roles. Formally, $\exists r_1, r_2 \in Roles(OU) : inf(r_1, r_2)$.

4.1.2 Functional Dimension

The *Functional Dimension* details the specific functionality of the system, based on services, tasks and objectives, as well as the interactions of the system, activated by means of objectives or service usage. It allows defining the functionality of organizational units, roles and agents of the MAS, including services and objectives that these entities offer or consume.

Definition 3. *The Functional Dimension (FD) from the Organizational Structure of a Virtual Organization is defined as* $FD = \langle G, S, Ta, FuncRel \rangle$ *where:*

- *G represents the goals followed by the organization.*
- *S is the set of services that the system offers or requires.*
- *Ta are the tasks that compose the services.*
- $FuncRel = \langle GT, Client, Provider, Obtains, Achieves, Task, Invoke, Plan \rangle$ *is the set of relations of this dimension, where:*
  - $GT : G \rightarrow 2^G$ *is the Goal Tree of the organization, describing the dependencies between different goals of the organization.*
  - $Client : S \rightarrow 2^R$ *relates a service with the set of roles that use it.*
  - $Provider : S \rightarrow 2^R$ *relates a services with the set of roles that offer it.*
  - $Obtains : S \rightarrow 2^G$ *describes the set of goals that can be achieved by a service, thus defining the functionality of the system.*
  - $Achieves : Ta \rightarrow 2^G$ *defines the set of goals that are reached when a task is executed.*
  - $Task : S \rightarrow 2^{Ta}$ *shows how services are split in different tasks.*
  - $Invoke : S \rightarrow 2^S$ *describes the dependencies between services, showing which services need to be invoked by other services to complete their functionality, thus allowing the composition of services.*
  - $Plan : G \rightarrow 2^S$ *represents the sequence of services that must be followed in order to achieve a goal.*

Properties of the Relations. The Goal Tree relation is irreflexive, asymmetrical and transitive, since a goal cannot be related with itself, neither with its predecessor but it can be related with the successors of its successors.

It must be assured that the provider of a service must be a role contained in the same OU as the service. Formally:

$$\forall o \in OU \wedge \forall s \in Services(o) \rightarrow Provider(s) \subseteq Roles(o) \quad (6)$$

This restriction assures that the services of an OU will be provided only inside it, but they can be accessed by agents from other OUs (e.g. using the *Invoke* relation).

As pointed out in this section, the *Invoke* relation allows services to invoke other services to reach their goals. In order to execute this operation, it must be assured that the provider of the invoker service must be client of the invoked service. Formally:

$$\forall s_1, s_2 \in S, s_2 \in Invokes(s_1) : \forall r_1 \in Provider(s_1) \rightarrow \exists r_2 \in Client(s_2) \wedge r_1 = r_2 \quad (7)$$

A key issue for the system designer is to assure that the services located in an organizational unit must help to reach its goals. Formally, it is described as:

$$\forall o \in OU, \forall s \in Services(o) \wedge \forall g_1 \in Pursues(s) : \exists g_2 \in Goals(o) \rightarrow g_2 \in GT(g_1) \quad (8)$$

It is is possible that a specific goal of a service could not be reached by any of the tasks that compose it, so then this service must invoke another which should include at least a task that achieves this desired goal. Formally, it is expressed as:

$$\forall g \in Obtains(s_1) \rightarrow (\exists t \in Task(s_1) \wedge g \in Achieves(t)) \vee (\exists s_2 \in S \wedge g \in Obtains(s_2) \wedge s_2 \in Invokes(s_1)) \quad (9)$$

4.1.3 Environment Dimension

The Environment Dimension describes the artifacts, i.e. entities that populate the environment of a MAS. This dimension uses the concept of artifact (Ricci et al., 2007), an element introduced by the Agents & Artifacts (A&A) conceptual framework. These elements are employed by agents in order to reach their goals, since artifacts have no associated goals. Additionally, the A&A framework presents the concept of workspace, used to define the topology of the environment of a MAS.

Definition 4. *The Environment Dimension of a Virtual Organization is defined as* $ED = \langle WS, AR, EnvFunc \rangle$ *where:*

- *WS is the set of workspaces that build the environment of a MAS, where* $ws \in WS$ *is defined as* $ws = \langle Loc \rangle$ *and Loc is referred to the location of the workspace inside the environment.*
- *AR is the set of artifacts, where an artifact* $ar \in AR$ *is defined as* $ar = \langle PR, OP, LO, St \rangle$*, where:*
  - *PR are the observable properties of an artifact that agents can check without executing any operation on it.*
  - *OP is the set of operations that agents can execute when interacting with the artifact.*
  - *LO refers to the link operations, which allows the composition and distribution of artifacts.*
  - *St is the internal state of an artifact.*
- $EnvFunc = \langle Located, Composition \rangle$ *is the set of functions that act on the environmental elements, where:*
  - $Located : AR \rightarrow 2^{WS}$ *describes the set of workspaces where an artifact is located.*
  - $Composition : WS \rightarrow 2^{WS}$ *allows defining intersection and nesting relations between workspaces that build the environment.*

Properties of the Relations. The *Composition* relation is reflexive and symmetrical, since a workspace can intersect with itself. Additionally, it is necessary for an artifact to be contained at least in a workspace. Formally:

$$\forall ar \in Ar : \exists ws \in WS \rightarrow ws \subseteq Located(ar) \quad (10)$$

4.1.4 Normative Dimension

The Normative Dimension describes normative restrictions on the behavior of the entities of the system, including sanctions and rewards, based on the work in (Criado et al., 2010).

Definition 5. *The Normative Dimension of a Virtual Organization is defined as* $ND = \langle N, >_n \rangle$ *where:*

- *N is the set of norms of the system.*
- $>_n$ *is an order relationship between norms, defining the priority between them. This relation establishes a total relation order between the norms governing the system, avoiding the priority confusion when a norm is executed.*

Formally, a norm is defined as:

Definition 6. *A norm* $n \in N$ *is defined as* $n = \langle D, CO, AC, EX, SA, RE \rangle$ *where:*

- $D = \{O, F\}$ *is the deontic operator, i.e. obligations (O) and prohibitions (F) that impose restrictions in the behavior of the agents.*
- *CO is a logical formula that represents the action that must be carried out in case of obligations, or has to be avoided in case of prohibitions.*
- *AC, EX are well-formed formulas that determine the conditions of norm activation and expiration, respectively.*

- *SA, RE ∈ S are expressions that describe the actions (sanctions, SA; and rewards, RE) that will be carried out in case of violation or fulfilment of norms, respectively.*

Properties of the Relations. The priority function $>_n$ is asymmetrical and transitive, defining an univocal relation between the norms governing the system.

The topology of the system will also define new order relationships between norms. If an OU called ou_2 is contained in an OU named ou_1, its norms must have higher priority than the norms of ou_1. Formally,

$$\forall ou_1, ou_2 : n_1 \in Norms(ou_1) \wedge n_2 \in Norms(ou_2) \\ \wedge Contains(ou_1, ou_2) \rightarrow n_2 >_n n_1 \quad (11)$$

4.2 Organizational Entity

The Organizational Entity of a Virtual Organization is the set of active elements of the organization. These elements can change through time. They are considered as the dynamic elements of the system.

Definition 7. *The Organizational Entity of a Virtual Organization is defined as* $OE = \langle A, GR, AN, AS \rangle$ *where:*

- *A is the set of agents that populate the VO.*
- *GR is the set of groups that are currently in the system. A group is an instantiation of an organizational unit.*
- $AN \subseteq N$ *is the set of active norms of the system, i.e. all those norms whose activation condition is true but its expiration condition has not been reached yet* $(AC \wedge \neg EX)$.
- $AS \subseteq S$ *is the set of services that the agents of the organization are currently providing.*

The agents (A) populating the system are playing roles, they are located into groups (GR) and provide services (S), as described in the next subsection. An OU defines an organizational pattern for the agents that are inside it, but this does not define the concrete agents that must populate it. On the contrary, a group is a concrete instantiation of an OU, defining a set of agents that populate it. Thus, an OU can be instantiated by different groups.

4.3 Organizational Dynamics

The Organizational Dynamics presents the relations between the elements of the Organizational Structure and the Organizational Entity.

Definition 8. *The Organizational Dynamics of a Virtual Organization is defined as* $\phi = \langle plays, inUnit, provides, perceives, isUnit \rangle$ *where:*

- $plays : A \rightarrow 2^R$ *is a function that relates an agent with the set of roles that he is playing inside the organization.*
- $inUnit : A \rightarrow 2^{GR}$ *is the function that describes the groups where an agent is located.*
- $provides : A \rightarrow 2^S$ *represents the set of services that an agent provides.*
- $perceives : A \rightarrow 2^{WS}$ *represents the set of workspaces that an agent is able to perceive.*
- $isUnit : GR \rightarrow OU$ *defines the type of organizational unit instantiated by a group.*

Properties of the Relations. The *plays*, *inUnit*, *provides* and *perceives* relations allow agents to play different roles, be located in different groups, provide different services and perceive different workspaces in the organization, respectively. The *isUnit* relationship allows knowing the type of organizational unit that a concrete group is instantiating.

The situation where an agent plays a role inside a unit and a scenario where an agent is inside an organizational unit playing a role can be checked in equations 12 and 13. It must be noted that the *Roles* function is recursive, as explained in section 4.1.1.

$$\forall r \in plays(a) \rightarrow \exists o \in OU \wedge \exists g \in inUnit(a) : \\ isUnit(g) = o \wedge r \in Roles(o) \quad (12)$$

$$\forall g \in inUnit(a) \rightarrow \exists o \in OU \wedge isUnit(g) = o \wedge \\ \exists r \in Roles(o) : r \in plays(a) \quad (13)$$

The first equation establishes that an agent can only play the roles provided by the groups where he is located. These roles are the ones provided by the organizational units instantiated by these groups. The second equation defines that an agent must play at least a role from each group where he is located.

In addition, using the *provides* relationship from ϕ, it is possible to define the set of active services (AS) from OE. Formally,

$$AS = \bigcup_{a \in A} s \in provides(a) \quad (14)$$

4.4 Multiagent Systems based on Virtual Organizations

In the previous sections, the different dimensions and entities that compose the state of a Virtual Organization at a given time were defined in a formal way. Nevertheless, a VO changes through time, passing from one state of the organization to another. Thus, it is necessary to define all the possible states of the organization as well as the allowed transitions between these states. For this issue, we based our work in the proposal from (da Rocha Costa and Dimuro, 2008).

To model the states of a Virtual Organization and their transitions, let VO be the universe of all the possible organizations O. A multiagent system based on virtual organizations is a structure $MAS = (VO, D)$ where, for every time $t \in T, D^t \subseteq VO \times VO$ defines transitions between different states of the system. In every state of the organization $O \in VO$, in a given time $t \in T$, there is a set of possible next states of the organization, denoted by $D^t(O) \subseteq VO$. Thus, for every $t \in T$, it holds that $O^{t+1} \in D^t(O^t)$, so an organization will only change to another state when it is allowed to reach from the initial state.

Since the organization is composed by three elements (OS, OE and ϕ), before executing a change of state it is necessary to check that these elements are able to change from the initial state to the possible destination state. Formally:

$$(OS^{t+1}, OE^{t+1}, \phi^{t+1}) \in D^t((OS^t, OE^t, \phi^t)) \leftrightarrow OS^{t+1} \in D^t_{OS}(OS^t) \wedge OE^{t+1} \in D^t_{OE}(OE^t) \wedge \phi^{t+1} \in D^t_{\phi}(\phi^t) \quad (15)$$

However, in order to swap from one state to another, it is not necessary to produce a change in all three elements that compose the Virtual Organization. A change ranges from a very small variation in one or few of the elements building the organization to a big amount of changes in a large amount of entities from the VO. Formally, it is possible that:

$$OS^{t+1} = OS^t \vee OE^{t+1} = OE^t \vee \phi^{t+1} = \phi^t \quad (16)$$

5 DISCUSSION

In section 3 an analysis of the most relevant formalization proposals was presented. The *Virtual Organization Formalization* (VOF) takes inspiration from features taken from some of the analyzed proposals. In this section, we depict a comparison between VOF and these background proposals.

Firstly, the organizational temporal evolution proposed by VOF is mainly based on PopOrg, which models the dynamics of the *population* (similar to our *OE*) and the *organization* (similar to our *OS*).

Regarding the structure of an organization, OperA offers relations between roles that are similar to those included in VOF. The supervision of VOF is similar to the combination of the power and authorization relations of OperA, expressing that an agent is able to delegate its objectives to a subordinated agent, like the power relation does (the authorization relation expresses the power relation, but as a temporal situation). Also, in OperA, the objective that a subordinated agent can take from a superior agent is determined by the type of existing relation between roles, which establishes their hierarchy. However, VOF defines this hierarchy using the *RoleHier* relation.

In MOISE^{Inst}, the structural levels of the organization are split into: (i) *individual level*, built by the organizational roles, and presents hierarchy relations between roles (similar to our *RoleHier* relation); (ii) *social level*, which is built from link relationships between roles, classified as *acq* (*acquaintance*), i.e. having a representation of other agents, *com* (similar to our *inf* relation), in which agents are able to communicate between them, and *aut* expressing authority over other agents, thus combining *col* and *sup* relations from VOF; and (iii) *collective level*, which defines *groups* of agents, establishing the *compatibility* between roles and their *cardinalities*. VOF adds the *comp* relation, in order to express whether an agent can take a given role if he is taking another role.

LAO is the only analyzed proposal which explicitly models the topology of the system by means of dependency chains (that are interaction possibilities) between agents. The topology can be a *hierarchy*, if there is a chain of command, or a *network*, if every agent is responsible for an organizational goal and has a delegation relationship to another agent. VOF models the topology of the system using *Contains* relations between OUs. These relations allow defining three types of organizations: *hierarchy*, similar to the structure defined by LAO; *team*, when all agents collaborate with each other; and *plain*, which assumes information relationships between roles.

Regarding the Functional Dimension, although PopOrg and POMF model concepts that are similar to services (by means of exchange processes or workflows, respectively), they are better described in VOF. PopOrg focuses on the actions developed by the process and the agents that are carrying them out, while POMF is focused on describing the tasks that compose a given workflow. VOF goes beyond, (as it follows a Service Oriented Approach) and formalizes a service by means of the roles that it can provide and consume, the goals that can be achieved with this service, the invoke relationships between services, and the tasks that compose each service (as well as the goals that these tasks help to reach).

The environment used in VOF is based on the Agents & Artifacts conceptual framework, which was included in the SODA metamodel, but it has not been included in any other formal approach yet.

VOF model norms in a very similar way to the proposal of MOISE^{Inst}, although they use different languages to describe norms. VOF is able to relate a norm to a set of OUs, using the *Norms* relationship from *DimRelations*, limiting its effect only to this set.

The Organizational Entity from VOF can be also

compared with other proposals. For example, a specified group is defined in MOISE^{Inst} as a 'group specification', while VOF defines it as an Organizational Unit. On the other hand, a group instantiation is named 'group' in both MOISE^{Inst} and VOF. In addition, the *OE* from VOF defines the set of norms and services that are currently active in the organization.

Finally, VOF clearly divides the static elements of the system (i.e. elements that will produce a structural change if they are modified) and the more dynamic elements of the MAS, represented in the *OE*. Our specification gives agents the possibility to belong to a specific group and provide or use a service.

6 CONCLUSIONS AND FUTURE WORK

This work presents a formal specification for Virtual Organizations, named VOF (*Virtual Organization Formalization*), which is composed by: (i) the Organizational Specification, which details the static components of the system and divides them by means of the organizational dimensions; (ii) the Organizational Entity, which defines the active elements of the system; and (iii) the Organizational Dynamics, which details the relationships between elements from the Organizational Specification and the Organizational Entity.

Additionally, we have analyzed a set of different formalizations, focusing on typical organizational concepts taken from the Organizational Dimensions. After this analysis, we noticed that the analyzed formalizations do not take into account all concepts from Organizational Dimensions. Therefore, our proposal is aimed to cover all these concepts and to provide a formalization as much complete as possible.

As future work, this formalization will help us when dealing with concepts related to adaptation in Organization Centered Multiagent Systems, being easier for us to identify the entities of the system which would change through time. VOF will be integrated into the reasoning process of BDI agents, in order to develop agents that are able to know whether an organization is working in a correct way, or he needs to execute an adaptation process. Moreover, using this formalization we will be able to check the correctness of a defined OCMAS.

ACKNOWLEDGEMENTS

This work is supported by TIN2009-13839-C03-01 and PROMETEO/2008/051 projects of the Spanish government and CONSOLIDER-INGENIO 2010 under grant CSD2007-00022.

REFERENCES

Clarke, E. (1999). Model checking. In *Foundations of Software Technology and Theoretical Computer Science*, page 54. Springer.

Criado, N., Argente, E., and Botti, V. (2010). Rational Strategies for Autonomous Norm Adoption. In *Proc. COIN@AAMAS*, pages 9–16.

Criado, N., Argente, E., Julian, V., and Botti, V. (2009). Designing virtual organizations. In *Proc. PAAMS*, volume 55 of *Advances in Soft Computing*, pages 440–449.

da Rocha Costa, A. and Dimuro, G. (2008). Semantical concepts for a formal structural dynamics of situated multiagent systems. *Proc. COIN III*, pages 139–154.

Dignum, V. (2003). *A model for organizational interaction: based on agents, founded in logic*. PhD thesis, Utrecht University.

Dignum, V. and Dignum, F. (2007). A logic for agent organizations. In *Proc. FAMAS*, pages 83–100.

Emerson, E. (1991). Temporal and modal logic, Handbook of theoretical computer science (vol. B): formal models and semantics.

Foster, I., Kesselman, C., and Tuecke, S. (2001). The anatomy of the grid: Enabling scalable virtual organizations. *Int. J. High Perform. Comput. Appl.*, 15(3):200.

Gâteau, B., Boissier, O., Khadraoui, D., and Dubois, E. (2005). Moiseinst: An organizational model for specifying rights and duties of autonomous agents. In *Proc. EUMAS*, pages 484–485.

Gonzalez-Palacios, J. and Luck, M. (2007). Towards compliance of agents in open multi-agent systems. In *Software Engineering for Multi-Agent Systems V*, Lecture Notes in Computer Science. Springer.

Grossi, D., Dignum, F., Dastani, M., and Royakkers, L. (2005). Fundations of organizational structures in multiagent systems. In *Proc. AAMAS*, pages 690–697.

Haesevoets, R., Weyns, D., Holvoet, T., and Joosen, W. (2009). A formal model for self-adaptive and self-healing organizations. In *ICSE09 Workshop*. IEEE.

Jonker, C., Sharpanskykh, A., Treur, J., and Yolum, P. (2007). A framework for formal modeling and analysis of organizations. *Appl. Intell.*, 27(1):49–66.

Picard, G., Hübner, J., Boissier, O., and Gleizes, M. (2009). Reorganisation and self-organisation in multi-agent systems. In *Proc. OrgMod'09*.

Popova, V. and Sharpanskykh, A. (2006). Process-Oriented Organization Modeling and Analysis Based on Constraints. *Enterprise Information Systems - Contains Special Issue: Modelling, Simulation, Verification and Validation of Enterprise Information Systems*.

Ricci, A., Viroli, M., and Omicini, A. (2007). Give agents their artifacts: the A&A approach for engineering working environments in MAS. In *Proc. AAMAS*, page 150.