Access Control in Virtual Communities
Prohibition, Permission, Authorization and Delegation of Power in the Grid

Guido Boella
Dipartimento di Informatica
Università di Torino
Italy
e-mail: guido@di.unito.it

Leendert van der Torre
CWI
Amsterdam
The Netherlands
e-mail: torre@cwi.nl

Abstract

We are interested in the design of access control policies for virtual communities of agents based on the grid infrastructure. In a virtual community agents can play both the role of resource consumers and the role of resource providers, and they remain in control of their resources. We argue that this requirement imposes a distinction between the authorization to access a resource given by the virtual community and the permission to do so issued by the resource providers. Our model is based on a logical multiagent framework that distinguishes the three roles of resource consumption, provision, and of authorization.

1 Introduction

Access control is "the process of mediating every request to resources and data maintained by a system and determining whether the request should be granted or denied" [26]. Access control policies are central in *in virtual communities* based on the grid or on peer to peer systems. Pearlman *et al.* [21] define a virtual community as a large, multi-institutional group of individuals who use a set of rules, a policy, to specify how to share their resources, such as disk space, bandwidth, data, online services, *etc.* Policies in virtual communities become more complex than in distributed systems due to, e.g., the following reasons.

- Every agent in the community can play both the role of a resource consumer as well as that of a resource provider: agents do not only use resources, but they also put at disposal to the other participants of the community the resources they own. Resource providers retain the control of their resources and they specify in local policies the conditions of use of their resources.
- In centralized systems the central manager permits agents to access the resources which it owns and controls, according to the policies defined by itself. In contrast, in virtual communities, access control cannot be directly implemented by a central manager who owns all the resources.
- Resource providers should implement local access control according to the community's security policies. However, they should also not be overburdened by the task of continuously updating the policies as they change and new members join the community. Hence, a resource provider *delegates* to a central agent (called *Community Authorization Service* or CAS by [21]) who is not in control of its resources the task of deciding which requests must be granted and which denied.
- Agents who participate to the community are heterogeneous and change frequently, so they cannot be assumed to be always cooperative and to stick to the system policies, as concerns both requesting access to resources and providing access to their resources.

Moreover, new questions are raised, like whether the task of *authorizing* requests performed by the CAS is identical to the task performed by a resource provider when it *permits* access, and what is the relation between these two agents.

In this paper we address the problem how distributed access control policies composed by prohibitions, permissions and authorizations can be designed. To disentangle the different concepts involved in policies for virtual communities, our design is based on a logical multiagent model: so we keep advantage both from the clearness of a logic model and from the recent developments in agent theory. Two subproblems of such a model are:

1. How should permissions and authorizations be distinguished and how are they related?
2. How can a resource provider delegate to the CAS the power of authorizing resource consumers and why can the power to issue permissions not be delegated?

The problem of designing access policies for virtual communities has been recently raised, e.g., by Pearlman *et al.* [21] and Sadighi Firozabadi and Sergot [24]. Pearlman *et al.* [21] argue that the solution is "to allow resource owners to grant access to blocks of resources to a community as a whole, and let the community itself manage fine-grained access control within that framework". The centralized management of resources owned by the single resource providers is performed by a *Community Authorization Service*: "A community runs a CAS server to keep track of its membership and fine-grained access control policies. A user wishing to access community resources contacts the CAS server, which delegates rights to the user based on the request and the user's role within the community. These rights are in the form of capabilities which users can present at a resource to gain access on behalf of the community".

In Figure 1 we depict the process of accessing a resource in a virtual community, as described by Pearlman *et al.* [21]. When a resource provider a_3 wants to join a community (step 1), it informs the CAS a_2, which replies with the requirements on how its resource must be shared with the community (2). When a resource consumer a_1 wants to access the resource of agent a_3, it must not only authenticate itself with agent a_2 providing its credentials (3), but it must also get a proof that its request conforms to the community's access policy. This proof is expressed by a *capability* (e.g., a X.509 certificate) provided by agent a_2 to a_1 (4), which identifies the agent and states that it is *authorized* to access the resource. Now, agent a_1 can make the actual request to a_3, forwarding it the capability (5). After checking the truthfulness of the capability, agent a_3 replies to a_1 (6).

However, the authorization by agent a_2 contained in the capability is not enough for agent a_1's request being granted. In a virtual community, agent a_3 maintains the control of its resource: the request is granted only if it is also permitted by the local policy of agent a_3.

In this paper, we argue that the authorization issued by the CAS is conceptually different from the permission granted by the resource provider and that the resource provider delegates to the CAS the power to issue authorizations rather then to issue permissions, since the latter power requires being in control of a resource:

A resource provider *controls* its resources: it receives requests and decides whether it permits access. Only the resource provider who controls these resources can forbid or permit access, in the sense that it lets agents access them; in fact, without its willingness to do so the resources are not available even to authorized users. The power to permit access cannot be delegated. E.g., a resource can be accessed only if the web server answers to a request. Moreover, nothing prevents the resource provider to grant access to agents which are not authorized.

Authorities (like the CAS) have the power to authorize access but not the power to permit access to these resources since they do not control them. They can exercise the power to authorize according to the virtual community security policies since they have the resources to implement the system policies, have more detailed knowledge about the current policies, about who are the current participants in the communities and which are their roles.

The notion of authorization to access a resource and the notion of permission should be kept distinct to have a correct model of the situation and to prevent dangerous misunderstandings in designing access policies

We analyze this distinction in a multiagent framework which applies several ideas developed in theoretical and legal studies. A cue that these notions have different properties is found also in the ordinary use of the terms authorization and permission. E.g., for the Cambridge Advanced Learner's Dictionary [22] permitting is "to allow something", "to make it possible for someone to do something, or to not prevent something from happening", while authorizing means "to give someone *official* permission to do something". Moreover, dictionaries of law like [13] argue that authorizations and permissions are related but different concepts, and that authorizations do not create new permissions.

Our model is based on some assumptions. First, as explained in detail in Section 2, we use for the individual agents existing agent technology. To incorporate prohibitions and permissions we attribute mental attitudes to the normative system, as proposed by [2], an assumption which facilitates the design as well as the evaluation of the design. Second, we believe that a model based on agents and defined in terms of well founded notions inspired to the regulation of human society, as prohibitions, permissions and authorizations, can be more easily understood by human users and security managers; moreover, it can be applied also to virtual organizations composed by both artificial agents and humans. Third, all the roles - consumers, providers and authorities - are modelled as heterogeneous autonomous intelligent agents: to understand the behavior of the system they belong to it is necessary to reason on their beliefs and their motivations: desires and goals, and prohibitions and permissions they are subject to.

This paper is organized as follows. In Section 2 we introduce our multiagent design for distributed access control, and in Section 3 we introduce the formal agent model and and the definition of norms, authorizations and delegation. Moreover, in Section 4, we discuss a scenario where the formal model is applied, and in Section 5 we discuss the theoretical foundations of the design.

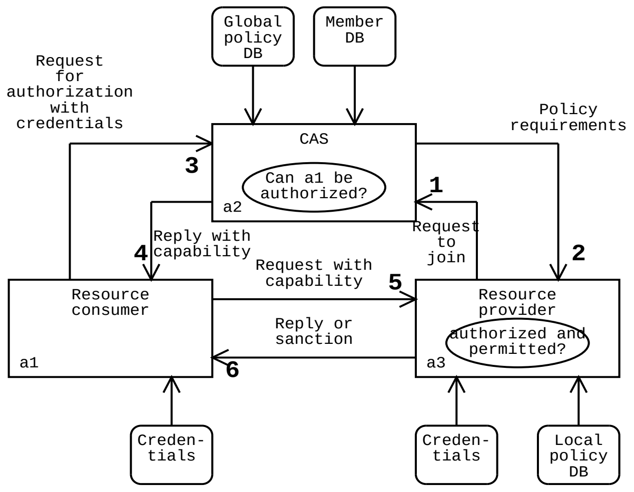

Figure 1. The players in a virtual community.

2 A multiagent design for access control

In this section we first introduce the three dimensions of our design: the agents and roles involved in the model, the design of the individual agents, and the primitives we use to incorporate normative concepts. Thereafter, we define prohibitions, permissions, authorizations and delegations.

2.1 Agents and roles

In virtual communities there is no separation of resource providers from resource consumers, and they can play the role of authorities too. Our design is therefore based a single set of agents, which can each play one or more of the following roles.

First, in the role of resource consumers, agents can access resources to achieve their goals. They are subject to norms regulating security, prohibitions and permissions, and also endowed with authorizations to access resources.

Second, in the role of resource providers, the agents can provide access to the resources they own. We call this the *normative role*, since they can issue norms, i.e., prohibitions and permissions about the access of a resource, and enforce their respect by means of sanctions, and delegate the power to authorize resource consumers.

Third, in the role of what we call authorities, the agents can declare resource consumers authorized when they are requested to do so. They know that their declarations are considered as authorizations by the resource providers since they have been delegated the power to authorize resource consumers on behalf of resource providers.

We represent agents by a set $\mathcal{A} = \{a_1, a_2, \ldots\}$, and agent a_i in a consuming role by $c(a_i)$, in a producing role by $p(a_i)$, and as an authority by $u(a_i)$.

2.2 Design of individual agents

In the design of individual agents, we incorporate the standard beliefs-desire-intention or BDI design as it has been developed in agent theory and standardized, e.g., in FIPA. Note that this standardization not only concerns the agents themselves, but also the communication and interaction protocols between them. Further developments in this area can directly be incorporated in our design for distributed access control.

In agent models, mental attitudes are assigned to agents: their beliefs, abilities and motivational attitudes. In the BDI model, agents' behavior is governed by their specific balance between beliefs, desires and intentions. In our model, we assign these mental attitudes to each role of the agent.

The agent perspective is not only useful to design distributed access control policies, but also to evaluate the design by simulation. To formalize decision making in a multiagent setting, in [5] and [6] we introduce a qualitative game theory which can be used to model the decision process of agents subject to permissions and authorizations as defined in this paper.

Agents deliberate to take actions; in this paper we are interested in particular in the access of resources. Typical access actions are rwx of unix, CRUD of databases and, more recently, access to web resources and services. We therefore define several *resource actions* for manipulating resources. Let RS be a set of resources. The set of resource actions of agent a_i for a resource r are RA_i=w,u,c,x,... such as $w(r)$ (write r), $u(r)$ (update r), $c(r)$ (create r), $x(r)$ (execute r), *etc.*

2.3 Primitives for normative notions

To define prohibitions, permissions, authorizations and delegation of institutional power, we introduce primitives for violations, sanctions, institutional facts and *counts as* relations.

The first concept is *violation*. The normative role can decide whether something is considered a violation or not. Thus, a violation is modelled as an atomic action.

The second concept is *sanction*. Since it is not possible to assume that all agents are cooperative and respect the norms, sanctions provide motivations to fulfill the norms. A sanction is an action negatively affecting an agent, i.e., the agent desires the absence of the sanction. If an agent does not respect a prohibition, we do not identify between behavior that is considered a violation and sanctions which are applied in case of violations. Consequently, there are also actions that represent sanctions.

The third concept is a *counts as* relation. We use it to define other related concepts such as having the power, being empowered, and institutional facts. The counts as relation is a binary relation between actions and institutional facts. An example of this relation is the fact that a signature by the head of the department on a purchase order counts as the institutional commitment of the department to pay for that order: the head of the department has the institutional power to buy on behalf of the department. If an action x of agent a_1 counts as an institutional fact q for the agent a_2 in a normative role, i.e., $p(a_2)$, $counts\text{-}as_2(x, q)$, then agent a_2 believes that q is a consequence of agent a_1's doing x. We describe this situation saying that agent a_1 has the power to create the institutional fact q by doing x.

2.4 Permissions, authorizations and delegation

The basic idea of our design is that we use only the concepts introduced thus far - actions, beliefs, desires and goals - because in that case we can reuse existing agent technologies to build distributed access control policies. But how can these concepts be used to define permissions, authorizations, and delegations? Do we not need additional primitive concepts like obligation and power? Roughly, the basic ideas is as follows. Prohibitions and permissions are defined in terms of goals attributed to the normative role, together with some additional constraints; and authorizations are beliefs attributed to the normative role, together with some additional constraints due to the fact that there are three agents (or roles) involved.

Prohibition to do an action is the obligation not to do the action. Prohibitions are defined as goals of normative roles. This is paraphrased as: Your wish (goal, desire) is my command (prohibition). The unfulfillment of the goal is considered as a violation and is sanctioned.

Permission makes direct reference to the definition of prohibition. A behavior is permitted if it is not considered by a normative role as a violation and thus it is not sanctioned. The main role of permissions is to provide exceptions to prohibitions in a given context [11]. For example, a permission to kill in self-defence makes sense only in the context of a prohibition to kill. Thus a permission is based on a goal of the normative role not to consider a given behavior as a violation.

Authorization is a belief of a normative role which appears as a condition in a permission. I am authorized only if You believe so.

Declaration of authorization is an action of an authority which states that an agent can be considered authorized according to its own policy. Authorities can have whatever reason for declaring authorized someone: e.g., they can be forbidden not to do so, or they can have the goal that only some agents are authorized (if the authority is the manager of the system).

Delegation is the change of authority. The authorities play the role of dispatcher of declarations. The declaration turns into a belief of the normative role that an agent is authorized. A normative role delegates the authority when it joins the community.

Before we can make prohibitions and permissions more precise, we have to consider the notion of *control*, because an agent depends on the agent in control of the resource [28]. E.g., in virtual communities agents are dependent for their membership to the system: if they do not stick to policies they are denied citizenship. Moreover, agents depend on resource providers for the possibility and the quality of access to the resources: e.g., in P2P file sharing systems a consumer has a reduced bandwidth if it does not share files too. A resource provider is an agent who is in control of a resource if it receives requests and decides whether to let access take place or to deny it. In Sadighi Firozabadi *et al.* [25]'s terminology, this is a preventative normative system. In some situations, however, accesses cannot be prevented, e.g., for efficiency reasons identities cannot be always checked and only statistical monitoring and audit is performed: only a detective normative system is possible. Hence, we base the notion of control on a more general definition: an agent is in control of a resource if it receives requests and can sanction the agents which make forbidden requests. As a particular case, we have the preventative model discussed above: the agent in control sanctions the violation by avoiding that the access to a requested resource achieves its effect (e.g., an HTTP get request receives as a response an error message).

Agent $p(a_2)$ is *in control of a given resource* r for what concerns a resource action $f_1(r)$ executed by agent $c(a_1)$ on r, $control_2(f_1(r))$, iff:

- Agent $c(a_1)$ who (tries to) commits access violations about $f_1(r)$ can be sanctioned by agent $p(a_2)$.

In [8] we identify the possibility to sanction an agent as the essential precondition for the ability to create norms: prohibitions and permissions. The definition of prohibition is given in terms of the mental attitudes of the agent and of the normative role concerning violations and sanctions. A prohibition $F_{(1,2)}(x, s \mid c)$ is read as 'agent $c(a_1)$ is forbidden in system $p(a_2)$ to see to it that x in context c, otherwise it is sanctioned with s'. An agent $c(a_1)$ is forbidden by a resource provider $p(a_2)$ to do x, $F_{(1,2)}(x, s \mid c)$, iff:

1. If condition c holds, agent $p(a_2)$ wants the absence of x and that agent $c(a_1)$ adopts $\neg x$ as its decision.
2. If $x \wedge c$ then agent $p(a_2)$ has the goal that x is considered as a violation.
3. If agent $p(a_2)$ considers x as a violation then it has as a goal that it sanctions agent $c(a_1)$ with s.
4. Agent $c(a_1)$ has the desire not to be sanctioned.

Item 4 specifies that the sanction must be a motivation for respecting the prohibition.

An agent $c(a_1)$ is permitted by agent $p(a_2)$ to do x in context c, $P_{(1,2)}(x \mid c)$ iff:

- If $c \wedge x$ then agent $p(a_2)$ has the goal not to consider x as a violation.

There is an asymmetry between prohibitions and permissions, since the latter ones play the role of exceptions to the former ones: the content of the prohibition is the negation of a goal of the normative role, in contrast the content of a permission is not its goal. In the case of the permission to kill in self defence, the normative system clearly does not have the goal that someone is killed even in self defence circumstances.

Differently from many other works on prohibitions and permissions we do not represent deontic concepts by means of primitive operators. Our choice depends not only on theoretical reasons (as discussed in [2]), but especially on the necessity to consider the motivations underlying the decisions of agents. In a heterogeneous system like the grid infrastructure, as also [24] argue, it is not possible to assume that the mere existence of a prohibition is a sufficient motivation for its respect. Moreover, the agents have to deal with obligations conflicting with their motivations and obligations conflicting with each other. So it is necessary to have an explicit model of their decision process under sanction based prohibitions.

Authorizations are the means used by agents to regulate the access of consumers to resources which they do not control. But there is no way to make authorized users access a resource without a permission by the resource provider: hence, authorizations are distinct from and presuppose permissions. An authorization is useless unless the resource provider permits authorized agents to access the resource r it controls: authorizations change what is prohibited to an agent and legitimate an action but without introducing or removing any norm.

Consider the following example. An agent a_3 joins some virtual community; it will both use the resources provided by the community, say downloading shared files, and provide its resource r to the other members of the community, say some of its disk space to store files: agent a_3 plays both the role of a resource consumer, $c(a_3)$, and that of a resource provider, $p(a_3)$. Since agent $p(a_3)$ controls its disk space (it is the only one who can decide that storing or retrieving files take place), it regulated the access to the disk by means of some local policy: prohibitions and permissions. E.g., it prohibited to read files during the day, it prohibited to store files exceeding 2.5Mb and it permitted to store compressed files whatever their size.

When agent a_3 joins the community, it agrees that also some of the members of the community use its disk space resource r. In order to do so, in principle, agent $p(a_3)$ could modify the norms regulating the access to its resource: e.g., by maintaining the prohibitions to read file during the day and to store large files and the permission to store zipped files, and by adding the permission about which members of the community can store and retrieve files on its disk space.

However, this solution imposes on agent $p(a_3)$ an heavy burden. In fact, even if the problem of authenticating which are the current users of the community can be dealt with by some trusted third party who gives them e-certificates, there are still some other complexities: it remains the problem of which members of the community are the ones which the community currently wants that they can access the resource and under which conditions they can do so. Moreover, the community access policies may change with time so that agent $p(a_3)$ should be kept informed and should modify the norms regulating access to the resource it owns. The complexity of modifications could also introduce unwanted errors in the access policy of agent $p(a_3)$.

What is needed is a solution which transfers the burden of managing the community policies to other agents, playing the role of authorities, which have the knowledge and resources to perform this task. However, it is impossible to say that an authority $u(a_2)$ changes the prohibitions and permissions posed by agent $p(a_3)$: norms are defined in terms of goals of agent $p(a_3)$, and authorities cannot change agent $p(a_3)$'s goals. Moreover, $u(a_2)$ is not in control of the resource so it cannot impose sanctions to motivate the respect of prohibitions. Finally, agent $p(a_3)$ wants to preserve its autonomy, so that it does not accept that someone else can change the norms regulating access to its resource.

The solution is that agent $p(a_3)$ creates only a permission saying that authorized agents can access the resource r. But the decision to authorize agents to access the resource

r is delegated to the authority $u(a_2)$ which has up to date knowledge on the system policies and members. Delegating the decision to authorize is easier than delegating permissions: the authorization is not a goal of the agent $p(a_3)$ but just a belief which can be induced by the authority by issuing e-certificates and capabilities to the agents which are authorized. Moreover, it does not require that the delegated agent is in control of the resource.

When the set of agents which can be authorized changes as a consequence of new community policies, agent $p(a_3)$ does not have to change the norms regulating access: new authorizations are created when the authority $u(a_2)$ issues new capabilities (or, in our abstract terminology, $u(a_2)$ declares them authorized). The capabilities are recognized by agent $p(a_3)$ as the proof that the permission to access r applies to a consumer $c(a_1)$ requesting access to r.

Then we have three agents involved in an authorization with different roles:

1. The agent who is authorized is a resource consumer.
2. There is an agent, the "authority", who has the power to authorize: this agent decides which resource consumers should be authorized and declares them authorized.
3. The resource provider who considers declarations by authorities as authorizations of the resource consumers and thus permits access to the resource it controls.

There is an authorization by resource provider $p(a_3)$ to resource consumer $c(a_1)$ to access resource $f_1(r)$ when:

- Agent $p(a_3)$ controls access to resource r by $f_1(r)$, otherwise any permission to access $f_1(r)$ would be useless.
- There is a conditional permission to $f_1(r)$ which has among its conditions a fact representing that it is authorized, $u_3(f_1(r))$:

$$P_{(1,3)}(x \mid u_3(f_1(r)))$$

- Agent $c(a_1)$ believes that agent $p(a_3)$ believes it is authorized: $u_3(f_1(r))$, so that the conditional permission is enabled.

The condition $u_3(f_1(r))$ is made true by an action of an authority: authority $u(a_2)$ "declares" $c(a_1)$ authorized (for example by signing a sheet, or by providing it with an e-certificate or a capability). This declaration action denoted by $g_2(f_1(r))$ has a meaning only if it has an effect on the beliefs of the resource provider: this link is provided by the delegation of the institutional power to authorize agents.

Authority $u(a_2)$ is delegated the institutional power to authorize agent $c(a_1)$ to access a resource r with resource action f_1 on behalf of agent $p(a_3)$ by means of declaration $g_2(f_1(r))$, $Del_{(2,3)}(g_2(f_1(r)), u_3(f_1(r)))$, iff:

- A declaration $g_2(f_1(r))$ by authority $u(a_2)$ counts as $u_3(f_1(r))$ for the agent $p(a_3)$:

$$counts\text{-}as_3(g_2(f_1(r)), u_3(f_1(r)))$$

I.e., if agent $p(a_3)$ believes that $u(a_2)$'s action to declare agent $c(a_1)$ authorized $g_2(f_1(r))$ has as a consequence the authorization $u_3(f_1(r))$ by agent $p(a_3)$ to agent $c(a_1)$.

In order for agent $c(a_1)$ to reason about what is authorized and permitted to do, it must consider the beliefs, desires and goals of the other agents: in this way it can discover whether an action of agent $u(a_2)$ is considered as an authorization by agent $p(a_3)$; only if $p(a_3)$ considers $c(a_1)$ authorized, $p(a_3)$ leaves agent $c(a_1)$ access the resource without considering this action a violation and thus punishing agent $c(a_1)$.

Works on delegation logics ([1], [18]) address the problem of how not only resource providers can delegate authorities but also delegated authorities can delegate their power to authorize to other agents. For space reason we cannot enter into this issue, but note that the delegation of power from an authority to another one can be modelled again in terms of the counts as relation.

This discussion shows that nothing requires that agent $u(a_2)$, who is delegated the authority to authorize other agents, is itself permitted nor authorized nor delegated to authorize itself to access the given resource. The separation of institutional power from the permission to exercise it, identified by Makinson [19], is important for virtual communities; as Bandmann *et al.*[1] argue, an organization could "outsource some administrative task" such as assigning access rights to some agent without allowing it to have those access rights.

Moreover, authorizations only help the resource providers in understanding when the community's policy allows access. However, the authorizations do not constrain the freedom of the resource providers to deny access to agents which are authorized or to grant access to unauthorized agents. The autonomy of the resource providers can be limited by community policies directed towards their behavior: in particular, the community can impose prohibitions or permissions on what the resource provider should or should not consider as violations. As described in this section, this is possible only since resource providers depend on the community in that it can sanction them (e.g., by cancelling their membership to the community). In [5] we address this issue of global policies constraining local policies.

3 A formal model

In this section we introduce the agent model and formalize the notions of prohibition, permission, authorization and delegation.

3.1 Individual agent design

As mentioned before, for the individual agent design we can use any BDI agent design. We choose the rule based BOID architecture [10]; though in our theory, and in contrast to the BOID architecture, prohibitions are not taken as primitive concept. Beliefs, desires and goals are represented by conditional rules. Actions, called *decision variables*, can have conditional and indirect effects with a non-monotonic character.

First of all, the roles played by agents:

Definition 1 (Agents) *Let $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ be a set of n agents. An agent $a_i \in \mathcal{A}$ can play three roles:*

1. *Resource consumer, denoted as $c(a_i)$.*
2. *Resource provider, denoted as $p(a_i)$.*
3. *Authority, denoted as $u(a_i)$.*

Next *decisions*. We assume that the base language contains boolean variables and logical connectives. The variables are either *decision variables* of an agent, which represent the agent's actions and whose truth value is directly determined by it, or *parameters*, which describe the state of the world and whose truth value can only be determined indirectly. Our terminology is borrowed from Lang *et al.* [17].

Definition 2 (Decisions) *Let $A_i = \{m, m', m'', \ldots\}$, the decision variables of $a_i \in \mathcal{A}$, and $P = \{p, p', p'', \ldots\}$, the parameters, be $n+1$ disjoint sets of propositional variables. A literal is a variable or its negation. $d_i \subseteq A_i$ is a decision of agent a_i.*

The consequences of decisions are defined by the agent's epistemic state, i.e. its beliefs about the world: how a new state is constructed out of previous ones given a decision is expressed by a set of *belief rules*, denoted by B_i. Belief rules can conflict and agents can deal with such conflicts in different ways. The epistemic state therefore also contains an ordering on belief rules, denoted by $\geq_i^B$, to resolve such conflicts.

Definition 3 (Epistemic states) *Let a rule built from a set of literals be an ordered sequence of literals $l_1, \ldots, l_r, l$ written as $l_1 \wedge \ldots \wedge l_r \rightarrow l$ where $r \geq 0$. If $r = 0$, then we also write $\top \rightarrow l$.*

The epistemic state *of agent a_i, $1 \leq i \leq n$, is*

$$\sigma_i = \langle B_i, \geq_i^B \rangle$$

B_i is a set of rules; $\geq_i^B$ is a transitive and reflexive relation on the powerset of B_i containing at least the subset relation.

Example 1 *Let $s = \{p\}$ be the current state, $d_1 = \{a\}$, $B_1 = \{a \rightarrow q, a \wedge p \rightarrow \neg q\}$ and $\geq_1^B = \{a \wedge p \rightarrow \neg q\} > \{a \rightarrow q\}$: q is a consequence of action a, unless p is true: the second rule is an exception to the first one. The new state resulting from the decision d_1 in state s given the belief rules B_1 is $\{p, \neg q\}$: the applicable rules are $\{a \rightarrow q, a \wedge p \rightarrow \neg q\}$, but since they are conflicting only the rule $a \wedge p \rightarrow \neg q$ with higher priority in the ordering $\geq_1^B$ is applied.*

The agent's motivational state contains two sets of rules for each agent. *Desire* (D_i) and *goal (G_i) rules* express the attitudes of the agent a_i towards a given state, depending on the context. Given the same set of rules, distinct agents reason and act differently: when facing a conflict between their motivations, different agents prefer to fulfill different goals and desires. We express these agent characteristics by a priority relation on the rules which encode, as detailed in Broersen *et al.* [10], how the agent resolves its conflicts.

Definition 4 (Motivational states) *The* motivational state *M_i of agent a_i $1 \leq i \leq n$ is a tuple $\langle D_i, G_i, \geq_i \rangle$, where D_i, G_i are sets of rules, $\geq_i$ is a transitive and reflexive relation on the powerset of $D_i \cup G_i$ containing at least the subset relation.*

For simplicity, we assume *transparency* of epistemic and motivational states: each agent knows the other agents' mental states.

The decision process of an agent a_i tries to minimize (according to the ordering $\geq_i$ on goal and desire rules) the goal and desire rules in G_i and D_i which remain unsatisfied given a certain decision d_i.

Definition 5 (Unfulfilled motivational states) *Let $U(R, s)$ be the unfulfilled rules of state s,*
$U(R, s) = \{l_1 \wedge \ldots \wedge l_n \rightarrow l \in R \mid \{l_1, \ldots, l_n\} \subseteq s$ and $l \notin s\}$

The unfulfilled mental state description *of agent a_i is*

$$U_i = \langle U_i^D = U(D_i, s), U_i^G = U(G_i, s) \rangle$$

Example 2 *Given the motivational state of agent a_1 $\langle D_1 = \{\top \rightarrow z\}, G_1 = \{\top \rightarrow x, y \rightarrow w, z \rightarrow u\}, \geq_1 \rangle$ the unfulfilled motivational state description of agent a_1 in state $s = \{x, y\}$ is*

$U_1 = \langle U_1^D = \{\top \rightarrow z\},\ U_1^G = \{y \rightarrow w\} \rangle$

In calculating which are the effects of a decision d_i given an initial state s, the agent uses the belief rules B_i and the ordering on them $\geq_i^B$ to resolve the possible conflicts. Moreover, agent a_i must predict the decisions of the agents acting after itself by recursively modelling ([15]) them using the information on their belief, goal and desire rules captured by their motivational states. The interested reader can find the details of the qualitative decision model in [5] and [6].

3.2 Norms

Prohibitions and permissions are defined in terms of goals and desires of the bearer of the norm and of the normative role. To represent violations for each propositional variable we add a *violation variable*.

Definition 6 (Violation variables) *The violation variables of agent $p(a_j)$ are a subset of the decision variables of $p(a_j)$ written as V_j=$\{V^i_j(x) \mid x$ a literal built out of a propositional variable in $P \cup A_i$ $\}$: x is a violation by agent $c(a_i)$.*

Definition 7 (Conditional prohibition with sanction) *Agent $c(a_i)$ is prohibited by agent $p(a_j)$ to decide to do x (a literal built out of a variable in $P \cup A_i$) with sanction s (a propositional variable) under condition q (a proposition), $F_{(i,j)}(x, s \mid q)$, iff:*

1. *$q \rightarrow \neg x \in G_j$: if agent $p(a_j)$ believes that q it has as a goal that agent $c(a_i)$ adopts $\neg x$ as its decision.*

2. *$q \wedge x \rightarrow V^i_j(x) \in G_j$: if agent $p(a_j)$ believes that $q \wedge x$ then it has the goal $V^i_j(x)$: to recognize x as a violation done by agent $c(a_i)$.*

3. *$V^i_j(x) \rightarrow s \in G_j$: if agent $p(a_j)$ decides $V^i_j(x)$ then it has as a goal that it sanctions agent $c(a_i)$.*

4. *$\top \rightarrow \neg s \in D_i$: agent $c(a_i)$ has the desire not to be sanctioned.*

A permission to do x is an exception to a prohibition to do x if agent $p(a_j)$ has the goal that x does not count as a violation under some condition.

Definition 8 (Conditional permission) *Agent $c(a_i)$ is permitted by agent $p(a_j)$ to decide to do x (a literal built out of a propositional variable in $P \cup A_i$) under condition q (a proposition), $P_{(i,j)}(x \mid q)$, iff*

- *$q \wedge x \rightarrow \neg V^i_j(x) \in G_j$: if agent $p(a_j)$ believes $q \wedge x$ then it wants that x is not considered a violation done by agent $c(a_i)$.*

The permission overrides the prohibition if the goal that something does not count as a violation ($q \wedge x \rightarrow \neg V^i_j(x)$) has higher priority in the ordering on goal and desire rules $\geq_j$ with respect to the goal of a corresponding prohibition that x is considered as a violation ($q \wedge x \rightarrow V^i_j(x)$):

$\geq_j \supseteq \{q \wedge x \rightarrow \neg V^i_j(x)\} > \{q \wedge x \rightarrow V^i_j(x)\}$

We do not consider here the problem of how normative role's characteristics can be generated; e.g., see [7] for a discussion of the problem of the legal sources of norms.

3.3 Resource, authorization and delegation

We introduce now the notion of resource, of control of a resource, of authorization and delegation of the institutional power to authorize access to a resource.

An agent who manipulates a resource by means of some action is called a *resource consumer*:

Definition 9 (Resources) *Let RS be a set of resources. Let $RA_j = \{f_j(r) \mid r \in RS\}$ be a set of resource actions of agent $c(a_j)$ on $r \in RS$.*

As discussed in Section 2.4, the possibility to punish violations by means of some sanction s is among the preconditions for creating a prohibition; for this reason, the notion of controlling a resource, which is a precondition for issuing norms concerning access control, is defined in terms of a condition which appears also in the definition of prohibition. We add this definition anyway since Definition 7 of prohibition is given in a more general form referring to literals and not only to resource actions.

Definition 10 (Control of resource) *Agent $p(a_j)$ controls a resource action f_i of agent $c(a_i)$ on resource $r \in RS$, $control_j(f_i(r))$, iff:*

- *Agent $p(a_j)$ can negatively influence agent $c(a_i)$ when it executes $f_i(r)$ by means of some decision variable or parameter which it can control $s \in A_j \cup P$ such that $\top \rightarrow \neg s \in D_i$: agent $c(a_i)$ desires not to be sanctioned.*

An agent who controls a resource is called *resource provider*.

As a particular case, $s = \neg p$ can be a literal built out of a parameter representing the failure of accessing a resource: e.g., reading a file has the desired effect of knowing the content of the file, and blocking the reading action results in the impossibility of knowing the information contained in the file. $c(a_i)$ believes that $p(a_j)$ with $m \in A_j$ prevents to achieve the effect p of $f_i(r)$ which $c(a_i)$ desires; $f_i(r) \rightarrow p \in B_i$, $\top \rightarrow p \in D_i$ and m has the effect $\neg p$: $m \rightarrow \neg p \in B_i$ and $\geq^B_i \supseteq \{m \rightarrow \neg p\} > \{f_i(r) \rightarrow p\}$.

Besides issuing norms, an agent which controls a resource can consider other agents authorized to access the resource it controls:

Definition 11 (Authorizations) *Let the parameters P contain a set of so-called authorization variables:*

$$H_j = \{u_j(f_i(r)) \mid a_i \in \mathcal{A} \text{ and } f_i(r) \in RA_i \text{ and } control_j(f_i(r))\}$$

They are institutional facts representing that the resource provider $p(a_j)$ considers agent $c(a_i)$ authorized to access r with action f_i.

Instead, declaring an agent authorized does not have the requirement to control a resource:

Definition 12 (Declarations) *Let the decision variables of agent $u(a_k)$ contain a set of so-called declaration variables*

$T_k = \{g_k(f_i(r)) \mid a_i \in \mathcal{A}$ *and* $f_i(r) \in RA_i\}$

$g_k(f_i(r))$ means that agent $u(a_k)$ declares agent $c(a_i)$ authorized to access r with action f_i.

The point of declaring agents authorized is that a declaration generates an actual authorization if it counts as as an authorization for the normative role controlling the resource.

Definition 13 (Counts as relation) *A decision variable $x \in A_k$ of agent $u(a_k)$, counts-as q, q a literal built out of a parameter, for agent $p(a_j)$, $counts\text{-}as_j(x, q)$, iff:*

- *$x \rightarrow q \in B_j$: agent $p(a_j)$ believes that x has q as a consequence.*

Definition 14 (Delegation of authorization) *Agent $u(a_k)$ is delegated by agent $p(a_j)$ the institutional power to authorize agent $c(a_i)$ to do $f_i(r) \in RA_i$ by means of declaration $g_k(f_i(r)) \in T_k$ ($u_j(f_i(r)) \in H_j$), $Del_{(k,j)}(g_k(f_i(r)), u_j(f_i(r)))$, iff:*

- *$counts\text{-}as_j(g_k(f_i(r)), u_j(f_i(r)))$*

By transparency of mental states, if $g_k(f_i(r)) \rightarrow u_j(f_i(r)) \in B_j$ then all agents believe that $p(a_j)$ believes that $g_k(f_i(r))$ counts as an authorization.

An agent who has been delegated the institutional power to authorize access is called an *authority*. It is not requested to control any resource.

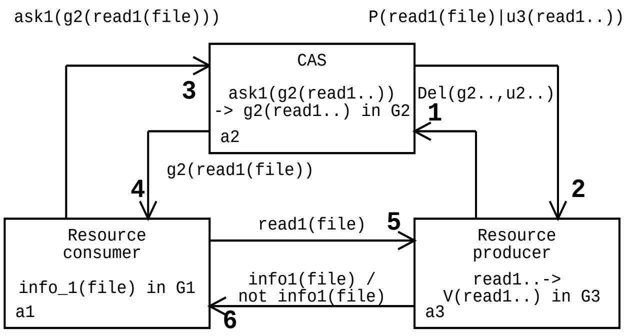

Figure 2. Requesting access.

4 Applicative scenarios

In this section we formalize some scenarios in our model. First of all, consider a simple situation where only a resource provider $p(a_3)$ and a resource consumer $c(a_1)$ are involved. Agent $p(a_3)$ provides access to a resource $file$, say a file, in particular, by action $read_1$, say reading the file $file$, which agent $c(a_1)$ can do on $file$. We model $read_1(file) \in A_1$ as a decision variable which makes agent $c(a_1)$ know the content of the file (represented by the parameter $info_1(file) \in P$); agent $c(a_1)$ represents this connection in its beliefs: $read_1(file) \rightarrow info_1(file) \in B_1$. However, the result of reading the file is not automatic, since agent $p(a_3)$ can prevent the success of the access by means of a decision variable $block_3 \in A_3$; agent $c(a_1)$ knows this fact, since $block_3 \rightarrow \neg info_1(file) \in B_1$; this belief rule works as an exception to the previous one since it has priority over it according to the ordering $c(a_1)$ uses to resolve conflicts among beliefs:

$\geq_1^B \subseteq \{block_3 \rightarrow \neg info_1(file)\} > \{read_1(file) \rightarrow info_1(file)\}$.

The fact that $p(a_3)$ can block the attempt of accessing the resource means that it is in control of the resource: in fact, the decision variable $block_3$ can make false $info_1(file)$ which is a desire of the consumer when it wants to access the resource: $\top \rightarrow info_1(file) \in D_1$. Hence, $control_3(read_1(file))$ is true by Definition 10.

For this reason, $\neg info_1(file)$ can be used as a sanction (as in Definition 7, it is a literal built out of a parameter and $p(a_3)$ indirectly controls it by doing $block_3$) when $p(a_3)$ poses prohibitions on its resource: the local access policy of $p(a_3)$ is to prohibit $c(a_1)$ to access the resource during the day ($\neg night$), in order to reduce the load on its server:

$F_{(1,3)}(read_1(file), \neg info_1(file) \mid \neg night)$.

This prohibition is defined in terms of goals of $p(a_3)$, as described by Definition 7:

$$\begin{aligned} G_3 \supseteq \{&\neg night \rightarrow \neg read_1(file), \\ &read_1(file) \wedge \neg night \rightarrow V_3^1(read_1(file)), \\ &V_3^1(read_1(file)) \rightarrow \neg info_1(file)\} \end{aligned}$$

These goals explain both the behavior of $p(a_3)$ and of $c(a_1)$: when $p(a_3)$ faces an attempt to access the resource $file$ during the day, $read_1(file) \wedge \neg night$, it has the goal to consider the attempt as a violation $V_3^1(read_1(file))$ ($read_1(file) \wedge \neg night \rightarrow V_3^1(read_1(file)) \in G_3$); thus, it has the goal to sanction it ($V_3^1(read_1(file)) \rightarrow \neg info_1(file)$) by executing the action $block_3$. Its decision is $d_3 = \{V_3^1(read_1(file)), block_3\}$. On the other hand, agent $c(a_1)$ is motivated by these goals not to try to attempt to access the resource by doing $read_1(file)$ during the day: by predicting the reaction d_3 of $p(a_3)$, it understands that its action does not result in the desired outcome $info_1(file)$ (due to $block_3 \rightarrow \neg info_1(file) \in B_1$); if, instead,

$read_1(file)$ is executed during the night, then the conditional goal of $p(a_3)$ to consider the attempt as a violation is not relevant: $read_1(file) \wedge \neg night \rightarrow V_3^1(read_1(file))$ is not applicable in situation $read_1(file) \wedge night$. $p(a_3)$ does not have any motivation to execute the blocking action to sanction $c(a_1)$ and, then, the access takes place with the desired effect (since $read_1(file) \rightarrow info_1(file) \in B_1$ and $\neg block_3$).

We now reconsider this scenario in the context of a virtual community. As shown in Figure 2 we have three agents: agent a_3, who plays the role of a resource provider $p(a_3)$ controlling the resource $file$, the resource consumer $c(a_1)$, and the CAS of the virtual community represented by agent a_2; it is playing the role of an authority $u(a_2)$.

When agent a_3 joined the community (step 1) it maintained the prohibition to access $file$ with action $read_1$ during the day ($F_{(1,3)}(read_1(file), \neg info_1(file) \mid \neg night)$), but it agreed to share the resource with the other members of the community even during the day. To implement the agreement, it could issue a permission to do $read_1(file)$; unfortunately, $p(a_3)$ does not know which are the current members (in this case whether agent a_1 is a member) nor which is the access policy of the community for what concerns the resource $file$ which $p(a_3)$ is sharing.

In contrast, agent $u(a_2)$, as a CAS, has up to date information about which are the members of the community and which of them can access the resource according to the community policy. Here, we ignore the problem of which are the current members and we focus again on agent $c(a_1)$, to check whether it should be allowed to do $read_1(file)$ or not. Assume that it is the case that the community wants that agent $c(a_1)$ access $read_1(file)$.

What is missing is a connection between agent $p(a_3)$'s access policy and agent $u(a_2)$'s. The first step to do is that agent $p(a_3)$ decides to consider agent $u(a_2)$'s decisions about $c(a_1)$ as its own decisions: when it joins the community, agent $p(a_3)$ acquires the belief rule that what agent $u(a_2)$ says (or declares, in our terminology: $g_2(read_1(file)) \in T_2$) about $c(a_1)$'s access to $file$ is considered by $p(a_3)$ as an authorization by itself: $u(a_2)$'s action $g_2(read_1(file))$ counts as $u_3(read_1(file)) \in H_3$ for $p(a_3)$. The belief rule

$g_2(read_1(file)) \rightarrow u_3(read_1(file)) \in B_3$

means that $p(a_3)$ delegated to agent $u(a_2)$ the institutional power to authorize $c(a_1)$ to do $read_1(file)$:

$Del_{(2,3)}(g_2(read_1(file)), u_3(read_1(file)))$

The institutional fact $u_3(read_1(file))$ - the authorization by agent $p(a_3)$ to access the resource - is useless if taken alone. To have a meaning it must have an effect on the goals of agent $p(a_3)$ about what $c(a_1)$ is prohibited, or not, to do. We argued that the authorization by itself does not create new permissions (and, hence, no new goal of agent $p(a_3)$). Rather, the authorization must be a condition of some goal of agent $p(a_3)$, a goal that prevents $p(a_3)$ to consider the access during the day ($read_1(file) \wedge \neg night$) as a violation (as, instead, it would be prescribed by the goal $read_1(file) \wedge \neg night \rightarrow V_3^1(read_1(file))$):

$read_1(file) \wedge \neg night \wedge u_3(read_1(file)) \rightarrow \neg V_3^1(read_1(file)) \in G_3$

where the second rule has priority over the first one:

$\geq_3 \supseteq \{read_1(file) \wedge \neg night \wedge u_3(read_1(file)) \rightarrow \neg V_3^1(read_1(file))\} > \{read_1(file) \wedge \neg night \rightarrow V_3^1(read_1(file))\}$

The second rule is an exception to the rule considering access as a violation. By Definition 8, agent $c(a_1)$ is permitted to access the resource during the day if authorized:

$P_{(1,3)}(read_1(file) \mid \neg night \wedge u_3(read_1(file)))$

$p(a_3)$ permits $read_1(file)$ only to agents authorized by $u(a_2)$, which, as a CAS, knows the current composition of the community and its global policy. Depending on the truth of $u_3(read_1(file))$ agent $c(a_1)$ is considered as a violator or not in doing $read_1(file)$. But the authorization does not change the local access policy of $p(a_3)$, i.e., the set of prohibitions and permissions, posed by agent $p(a_3)$ at the moment of joining the community; the authorization only changes the sphere of what is prohibited or permitted under certain circumstances.

When agent $c(a_1)$ wants to do $read_1(file)$ (5), it knows that it is forbidden to do so during the day, unless it is authorized by $p(a_3)$. It knows that in order to be considered authorized $u_3(read_1(file))$ by $p(a_3)$, it has to present a capability from agent $u(a_2)$ containing its declaration of authorization $g_2(read_1(file))$ (5). So, when it authenticates itself with the CAS (3) , it can also request a declaration by deciding for $ask_1(g_2(read_1(file))) \in A_1$.

Agent $c(a_1)$ has to take a decision and compares the different alternatives for achieving its goal $read_1(file)$: requesting the resource by doing $read_1(file)$ alone(step 5) or first asking for a declaration (3) and then requesting to access the resource (5). If it decides for the first solution it knows that it violates the prohibition $F_{(1,3)}(read_1(file), \neg info_1(file) \mid \neg nigth)$ and that it will be sanctioned with $\neg info_1(file)$ by $p(a_3)$ (6). How can agent $c(a_1)$ foresee the behavior of agent $p(a_3)$? It knows the mental state of agent $p(a_3)$; first of all, $p(a_3)$ has among its goals G_3 the goal that $read_1(file)$ is not executed and that the execution of such action is considered as a violation ($V_3^1(read_1(file))$) and thus sanctioned ($\neg info_1(file)$, from the definition of $F_{(1,3)}(read_1(file), \neg info_1(file) \mid \neg night)$).

If $c(a_1)$ decides to do $read_1(r)$ alone, it would trigger the goal of considering it as a violation and, as a consequence, the violation would trigger the goal to sanction it. However, since $P_{(1,3)}(read_1(file) \mid \neg night \wedge u_3(read_1(file)))$, the goals of $p(a_3)$ contain also an exception to the goal of considering $read_1(file)$ a violation, in case $read_1(file)$ is executed by an authorized agent $u_3(read_1(file))$.

From the agent characteristic of agent $p(a_3)$, $c(a_1)$ knows that this second goal is preferred to the first one: $\geq_3 \supseteq \{read_1(file) \wedge \neg night \wedge u_3(read_1(file)) \rightarrow \neg V_3^1(read_1(file))\} > \{read_1(file) \wedge \neg night \rightarrow V_3^1(read_1(file))\}$

So, in case it is considered authorized, the goal of considering $read_1(file)$ a violation is cancelled by the preferred goal not to consider authorized agents as violators (6).

How does $c(a_1)$ know that it is considered authorized by $p(a_3)$? Since $p(a_3)$ delegated $u(a_2)$ the power to declare authorizations, $c(a_1)$ knows that, according to $p(a_3)$'s beliefs B_3, a declaration $g_2(read_1(file))$ by $u(a_2)$ is considered as an authorization $u_3(read_1(file))$: $g_2(read_1(file)) \rightarrow u_3(read_1(file)) \in B_3$.[1] To get a declaration to be included in its capability, agent $c(a_1)$ can request one to agent $u(a_2)$ by doing action $ask_1(g_2(read_1(file)))$ (3). Also in this case agent $c(a_1)$ must reason about the motivations that lead agent $u(a_2)$ to reply to its request (4). For example, the CAS can simply have the goal to respond to the requests of agent $c(a_1)$:
$ask_1(g_2(read_1(file))) \rightarrow g_2(read_1(file)) \in G_2$

From this goal agent $c(a_1)$ can predict that its request to be declared authorized will be satisfied, so that its decision $d_1 = \{ask_1(g_2(read_1(file))), read_1(file)\}$ will lead to the desired consequence $info_1(file)$ without any sanction.

This is not the only possible motivation for agent $u(a_2)$ to declare agent $c(a_1)$ authorized. As Firozabadi and Sergot [24] argue it is possible that in heterogeneous virtual communities even the administrators can sometimes be considered not fully trusted. In a different scenario, it is possible that agent $p(a_3)$, when it joined the community, decided not only to consider the declarations by agent $u(a_2)$ as authorizations, but also to forbid the CAS not to declare authorized the agent $c(a_1)$ when it requests the capability to do $read_1(file)$:
$F_{(2,3)}(\neg g_2(read_1(file)), s' \mid ask_1(g_2(read_1(file))))$ where s' is some sanction affecting agent $u(a_2)$.

As in case of agent $c(a_1)$'s decision, agent $u(a_2)$, when it takes its decision, has to recursively model the decision of $p(a_3)$, in turn.

[1]For the sake of simplicity, in this paper we do not consider the problem of transmitting the capability containing the proof of the declaration and we assume that agent $p(a_3)$ is immediately acquainted with that. Details about communication issues and observations can be found in [5].

5 Foundations

In this section we contextualize our contribution with respect both to the problem of access control in virtual communities and to the logical foundations of the notion of institutional fact given in law studies and deontic logic.

The role played by the CAS in virtual communities can be formalized in terms of what we called the authority role. However, Pearlman *et al.* [21] use the term 'right' both for the authorizations provided by the CAS and the permissions granted by the resource providers: "the user effectively gets the intersection of the set of rights granted to the community by the resource provider and the set of rights defined by the capability granted to the user by the community."

This use of the terms right, authorization and permission as synonyms is frequent in policies for managing access control in distributed systems (e.g., [20], [18], [26]). In this paper we show why and how these concepts should be kept distinct in the context of virtual communities.

Even before the development of the grid architecture, the necessity to integrate autonomous resource providers has emerged in the field of federated databases. De Capitani and Samarati [12] have developed a two step authorization process for accessing data in a federation of databases: when a client wants to access data, it must be authorized first by the federation of databases and then by the single local databases. De Capitani and Samarati [12]'s model allows to specify different authorization strategies. A federation of databases, however, presents some differences with respect to a grid architecture. In particular, all the accesses to the data are requested to the federation and not to the local providers. So, the federation does not only play the role of the CAS in a grid architecture, but it is also in control of the federated data: it can prevent a client from accessing data which are accessible at the level of the local databases. Hence, in our terminology, both the federation and the local databases *permit* access to a resource, rather then merely authorizing it. Our model can deal also with such scenario.

A recently emerged requirement advanced, e.g., in [9] is that it is essential to allow agents which want to access a resource being informed about which policies are constraining a given resource. Our model can contribute to this issue since we enhance the relationship existing between an authorization given by the CAS and the permission given by the resource provider. Including authorizations explicitly in the conditions of a permission enables the resource provider to inform a resource consumer about the fact that it is not only necessary to stick to the resource provider's policy, but also to which policy of the community.

In this paper we assumed that the owner or stakeholder of the resource is also the agent who is in control of it. However, this is not the general case; sometimes, the agent or component (called the reference monitor) that grants or denies access to a resource can be distinct from the resource owner, and it acts in accordance to the access rules defined by the owner. In [6], we address the problem of systems where the agent, called *defender*, who implements the policies, is different from the agent who specifies the local access control policies on the resource it owns.

The necessity of a fine grained analysis of the concepts of permission and authorization in the security policy field is witnessed also by Sadighi Firozabadi and Sergot [24] who argue that a mere permission given by the resource provider to access a resource which it controls must be distinguished from the *entitlement* to access the resource: an agent is entitled and not merely permitted when the policies regulating the virtual community prohibit the resource provider not to permit the agent to access the resource.

Another distinction comes from the formalization of legal reasoning by means of deontic logic. Jones and Sergot [16] distinguish permissions from authorizations interpreted in the sense of having been delegated the *institutional power* to do something: "when we say that the Head of Department is authorized to purchase equipment, we mean first and foremost that he has been granted by the institution the power to enter valid purchase agreements". Instead, "sometimes when we say that an agent is authorized to do such-and-such we mean no more than that he has been granted permission to do it".

Law studies argue that a further distinction must be drawn also in this last sense of the term authorization as mere permission. The [13]'s dictionary of law argues that adding or removing an authorization does not change the normative status of an agent while a new permission does; i.e., authorizations do not change the norms (prohibitions and permissions), an agent is subject to; rather, authorizations lift the legal obstacles and limitations, thus legitimating an action of the agent: they change the sphere of what is prohibited or permitted to the agent without adding or removing norms. This is possible since norms have a conditional character so that what is currently considered as a violation or not depends on which are the norms that have their conditions satisfied in the current situation. The fact that authorizations do not modify the existing norms, but change what is prohibited and permitted to an agent anyway, means that authorizations enable the conditions of some permissions. Hence, the institutional power to authorize can be delegated to other agents who do not directly control the resources, since creating an authorization does not require to change prohibitions and permissions.

Some scholars argue, instead, that the power to create permissions can be delegated. [25], for example, propose a framework where this power can be delegated as any other power to create institutional facts. We show in this paper that once prohibitions and permissions are not considered as primitive logical entities, the preconditions for their creation emerge. When we define them in terms of goals of the normative role, it emerges that controlling a resource is necessary for issuing a norm.

In Section 1, we highlighted that according to Cambridge Advanced Learner's Dictionary [22] *officiality* seems to be the first dimension distinguishing permissions from authorizations: the official character of authorizations depends on the fact that they are *institutional facts*, and this character distinguishes them from permissions; an authorization is an institutional fact which appears among the conditions of some permission issued by a normative role: if the normative role believes this fact, the permission is enabled so that what is permitted in the current situation is changed by the authorization. The creation of institutional facts is a commonplace feature of legal systems and norm-governed organizations. According to [16], "it is that particular agents are empowered to create certain types of states by mean of the performance of specific types of acts. Typically, the states created will have a normative character".

Authorizations are created by authorities who have been delegated the institutional power to do so by some institution (in our model the institution is some normative role). An authority has been *delegated the power to create an institutional fact* if the institution recognizes the authority's action as *counting as* something else (as in Searle [27]'s construction of social reality). E.g., the fact that an authority declares an agent authorized counts as an authorization by a normative role.

For Jones and Sergot [16], the counts as relation expresses the fact that a state of affairs or an action of an agent "is a sufficient condition to guarantee that the institution creates some (usually normative) state of affairs". As [16] suggest this relation can be considered as "constraints of (operative in) [an] institution". [16] express them as conditionals embedded in a modal operator. For Governatori *et al.* [14], these conditionals must have a nonmonotonic character, since it is possible that under some circumstances a certain state of affairs does not counts as something else.

In Section 3.3 we modelled the counts as relation in terms of beliefs of the normative role. Since the beliefs of agents are modelled as nonmonotonic conditionals, these are suited to represent also counts as conditionals. For space reasons in this paper we will not discuss the logical properties of beliefs and of the counts as relation. By reducing counts as to beliefs we capture also the original Searle [27]'s idea that a institutional power inherently works only if *collectively recognized* by the agents composing the institution.

In summary, an authorization to access is meaningful only if there is a permission which has among its conditions the authorization. Note that a similar dependence relation between normative concepts exists between permissions and prohibitions: permissions require the existence of prohibitions they are an exception to.

6 Summary and concluding remarks

In this paper we introduce a design for distributed access control, based on existing agent technologies. Our approach is based on a distinction between three roles for each agent: as a resource provider, a resource consumer and authority. Using a minimal set of primitives, we introduce sanction based prohibitions, permissions, authorizations and delegations. Our design is capable of formalizing a wide range of scenarios. Organizational hierarchies can be introduced, and the extension to full role based access control is straightforward.

On the theoretical side, our design supports the thesis that the notion of authorization must be kept distinct from the notion of permission: an authorization is an institutional fact which, to be useful, requires the existence of some permission. Permissions, in contrast, derive their meaning from the existence of prohibitions they are exception of. Prohibitions and permissions can be issued only by agents who are in control of resources, authorization can be delegated. This distinction is necessary since, when a new agent joins a virtual community, it has to provide its resources in a way that is coherent with the policies of the community. The integration of the norms regulating the access to its resources with those of the community can be performed by other agents by means of the delegation of the power to authorize. In this way, changes to the community policies can be implemented by the authorities which have been delegated the institutional power to authorize rather then by the single resource providers who joined the community. In fact, an authority can change authorizations and as a consequence the sphere of what is forbidden or permitted for a resource consumer is changed without the need to change the prohibitions of permissions issued by the resource provider.

There are several issues for further research. First, Bandmann *et al.* [1] argue that delegation of authorization should be regulated since not all agents can be authorized by a delegated authority. Second, our framework is applied for modelling hierarchies of policies in order to represent the fact that the norms issued by local resource providers can be constrained by obligations and permissions posed at the global level; for example, a resource provider can be forbidden to permit access to its resources to certain resource consumers, or it can be obliged to permit access. Our framework has been extended to deal with this problem by means of obligations and permissions concerning what the local resource providers have to consider as a violation [3, 5]. Finally, in [4] we explore how to formalize our model using the standard BDI$_{CTL}$ logic [23] for agent verification.

References

[1] O. L. Bandmann, B. Sadighi Firozabadi, and M. Dam. Constrained delegation. In *IEEE Symposium on Security and Privacy*, 2002.

[2] G. Boella and L. van der Torre. Attributing mental attitudes to normative systems. In *Procs. of AAMAS'03*, Melbourne, 2003. ACM Press.

[3] G. Boella and L. van der Torre. Decentralized control: Obligations and permissions in virtual communities of agents. In *Procs. of ISMIS Conference*. Springer Verlag, 2003.

[4] G. Boella and L. van der Torre. Game specification in the trias politica. In *Procs. of BNAIC'03*, 2003.

[5] G. Boella and L. van der Torre. Local policies for the control of virtual communities. In *Procs. of IEEE/WIC Web Intelligence Conference*. IEEE Press, 2003.

[6] G. Boella and L. van der Torre. Norm governed multiagent systems: The delegation of control to autonomous agents. In *Procs. of IEEE/WIC IAT Conference*. IEEE Press, 2003.

[7] G. Boella and L. van der Torre. Permissions and obligations in hierarchical normative systems. In *Procs. of ICAIL 03*, pages 109–118, Edinburgh, 2003. AMC Press.

[8] G. Boella and L. van der Torre. Rational norm creation: attributing mental attitudes to normative systems, part 2. In *Procs. of ICAIL 03*, pages 81–82, Edinburgh, 2003. AMC Press.

[9] P. Bonatti and P. Samarati. Logics for authorizations and security. In J. Chomicki, R. van der Meyden, and G. Saake, editors, *Logics for Emerging Applications of Databases*. Springer Verlag, 2003.

[10] J. Broersen, M. Dastani, J. Hulstijn, and L. van der Torre. Goal generation in the BOID architecture. *Cognitive Science Quarterly*, 2(3-4):428–447, 2002.

[11] E. Bulygin. Permissive norms and normative systems. In A. Martino and F. S. Natali, editors, *Automated Analysis of Legal Texts*, pages 211–218. Publishing Company, Amsterdam, 1986.

[12] S. De Capitani di Vimercati and P. Samarati. Authorization specification and enforcement in federated database systems. *Journal of Computer Security*, 5(2), 1997.

[13] F. del Giudice. *Nuovo dizionario giuridico*. Simone Editore, 2001.

[14] J. Gelati, G.Governatori, N. Rotolo, and G.Sartor. Declarative power, representation, and mandate. a formal analysis. In *Procs. of JURIX 02*, 2002.

[15] P. J. Gmytrasiewicz and E. H. Durfee. Formalization of recursive modeling. In *Proc. of first ICMAS-95*, 1995.

[16] A. Jones and M. Sergot. A formal characterisation of institutionalised power. *Journal of IGPL*, 3:427–443, 1996.

[17] J. Lang, L. van der Torre, and E. Weydert. Utilitarian desires. *Autonomous agents and Multi-agent systems*, pages 329–363, 2002.

[18] N. Li, B. Grosof, and J. Feigenbaum. Delegation logic: A logic-based approach to distributed authorization. *TISSEC*, 6(1):128–171, 2003.

[19] D. Makinson. On the formal representation of rights relations. *Journal of philosophical Logic*, 15:403–425, 1986.

[20] J. Moffett and M. S. Sloman. Policy hierarchies for distributed systems management. *IEEE Journal of Selected Areas in Communications*, 11(9):1404–1414, 1993.

[21] L. Pearlman, V. Welch, I. Foster, C. Kesselman, and S. Tuecke. A community authorization service for group collaboration. In *Procs. of Policies for Distributed Systems and Networks International Workshop, POLICY 2002*. 2002.

[22] C. U. Press. *Advanced Cambridge Learners' dictionary*. Cambridge University Press, 2002.

[23] A. S. Rao and M. P. Georgeff. Decision procedures for BDI logics. *Journal of Logic and Computation*, 8(3):293–343, 1998.

[24] B. Sadighi Firozabadi and M. Sergot. Contractual access control. In *Procs. of 10th International Workshop of Security Protocols*, Cambridge (UK), 2002.

[25] B. Sadighi Firozabadi, M. Sergot, and O. Bandmann. Using authority certificates to create management structures. *Lecture Notes in Computer Science*, 2467:134–145, 2002.

[26] P. Samarati and S. De Capitani di Vimercati. Access control: Policies, models, and mechanisms. In R. Focardi and R. Gorrieri, editors, *Foundations of Security Analysis and Design LNCS 2171*. Springer Verlag, Berlin, 2001.

[27] J. Searle. *The Construction of Social Reality*. The Free Press, New York, 1995.

[28] J. S. Sichman, R. Conte, C. Castelfranchi, and Y. Demazeau. A social reasoning mechanism based on dependence networks. In A. G. Cohen, editor, *Proc. of 11th ECAI*, pages 188–192. Wiley, Chichester, 1991.