# Access Control Model for Inter-organizational Grid Virtual Organizations

B. Nasser[1], R. Laborde, A. Benzekri, F. Barrère, M. Kamel

*Université Paul Sabatier - IRIT/SIERA -*
*118 Rte de Narbonne F31062 Toulouse Cedex04 FRANCE*
*Telephone: +33 (0) 5 61 55 60 86 - Telecopy: +33 (0) 5 61 52 14 58*
{nasser, laborde, benzekri, barrere, mkamel} @irit.fr

**Abstract.** The grid has emerged as a platform that enables to put in place an inter-organizational shared space known as Virtual Organization. The Virtual Organization (VO) encompasses users and resources supplied by the different partners for achieving the VO's creation goal. Though many works offer solutions to manage a VO, the dynamic, on the fly creation of virtual organizations is still a challenge. Dynamic creation of VOs is related to the automated generation of access control policy to trace its boundaries, specify the different partners' rights within it and assure its management during its life time. In this paper, we propose OrBAC (Organization Based Access Control model) to model the Virtual Organization which serves as a corner stone in the VO creation automated process. OrBAC framework specifies the users' access permissions/interdiction to the VO resources, where its administration model AdOrBAC flexibly models the multi-stakeholder administration in the Grid.

## 1 Introduction

In today's demanding business environment and the rapid technological advancement, organizations tend to compete by investing in a strategic product development capability that grows in strength and competence with the entire organization. This necessitates minimizing the costs associated with training and developing dedicated skills in several domains concurrently. Massive data volumes treatment, long complex computations, unique or critical resources exploitation are examples of applications that necessitate special expertises and resources to deal with.

Enterprises are inclined to form supply chains of specialized partners to treat the different tasks via cooperation agreements and contracts. Grids supply an infrastructure to deploy internet-based collaborations [3,4,17]. The grid, from the user's point of view, is a virtual powerful device based on coordinated resources supplied by different partners. However, from administrative point of view, it is a dynamic sharing space called Virtual Organization (VO) that includes users, resources and sharing relationships [1,3,4,5].

Building a VO, in an open environment as the Internet, necessitates enclosing its boundaries. These boundaries encircle legitimate users and resources introduced by the different partners. In addition, it entails sharing relationships which specifies the way and the contexts of usage as: Physicians may *use* Computing-Device and *store* data within *work time*. As a way to achieve that, access control policies may be employed to restrain resources access (computing device, storage space) to legitimate users (physicians) in certain contexts (work time).

In addition to the large number of users and resources, the complexity of access control in Virtual Organizations rises from the fact that multiple access stakeholders are involved [5,8]. Each partner needs to keep control on his local assets (add, remove, modify permissions) without compromising discretion or delegating administration to other domains [5]. A client domain needs to manage his users without referring back to the resource provider. So finally, it results with a shared space with multiple access stakeholders, dynamic users and dynamic resources.

[1] Author to contact for more information

Current VO management solutions offer mechanisms to enforce access control policies. These solutions propose the use of access control lists [17] at the different resources sites or employ identity certificates with an out-of-band access control policy [6,7,8]. This vagueness with dealing with policy (out of band) can cause serious security breaches when new users or resources can be added or removed dynamically. Policy management becomes essential to prevent security breaches and administrative responsibilities should be clear and well specified.

In this paper, we propose a methodology to automate VOs creation starting first with the discovery of potential providers, acquiring access rights to certain resources and finally enforcing the access control policy on runtime upon resources allocation. We focus on an access control management model that constitutes the corner stone on the way to dynamic VO creation and management [1]. The criteria of selection identifies are: the need to separate policy from the dynamic infrastructure and the separation of administrative tasks among the different partners.

Based on these criteria, an access control model should be chosen to specify the users-resources relationships. Meanwhile its administration model specifies partners' administrative relationships and thus VO access control policy creation and management. Comparing different access control models as DAC (Discretionary Access Control), MAC(Mandatory Access Control), RBAC(Role Based Access Control), CBAC(Coalition Based Access Control) [9,10,11,18], we argue that OrBAC (Organization based Access Control) [12] is the most appropriate in our context. ORBAC abstracts users, actions and objects with role, activity and view where its administration model AdOrBAC [14] flexibly models the different administration responsibilities within an organization. This paper starts by introducing the grid environment and its virtual organization notion. In section 3 we cite some of the grid environment characteristics relevant to the access control. To well place our work we show the VO life cycle in section 4 to discuss the choice of an access control model in section 5. Sections 6 introduce OrBac access control model and AdOrBAC its administration model. In section 7 we show how the grid environment can be modeled using Or-BAC to have a clear vision of access control policy in a VO. Finally we discuss some modeling issues and conclude with future works.

## 2 The Grid

The term "Grid" was adopted in resemblance to the electric grid to designate a powerful system where a pool of distributed devices confederates to allow an authorized user to plug for on-demand energy. However in computer context, energy is supplied in the form of computer hardware and software for problem solving as computing, data storage or applications sharing. The grid has emerged as a platform that has as goal enabling coordinated resource sharing and problem solving in dynamic multi-domain virtual organization [5].

The different organizations in a grid environment federate into an alliance for a specific goal (ex. project cooperation, resources rental or application provider). They put in place what we call a Virtual Organization (VO) which constitutes the shared space between the different partners. An organization may need to participate in multiple Virtual Organizations according to its needs and thus have multiple shared spaces with multiple communities.

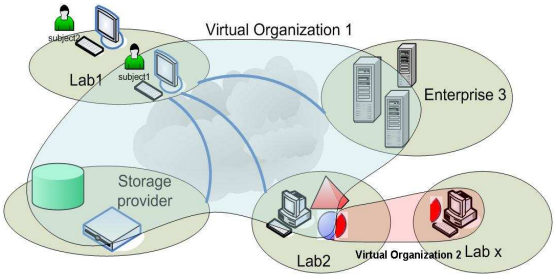


**Fig. 1.** Multiple partners participate in different VOs

Consider the example in Figure 1, a physicist at "Lab1" uses a simulation application at laboratory "Lab2" and then analyzes the data using computation cycles hired from "Enterprise3" and finally stores the whole results at a storage provider.

Virtual Organizations may have different attributes as long/short life-time and scopes that vary from intra-enterprise till hundreds of partners. These attributes depend on the goal of a VO and its problem nature. According to its goal, the Virtual Organization creation requires specification of sharing relationships for precise levels of control over how shared resources are put in place and used. This means that the access control policy should precise users' rights at the tapped resources. The granted rights may depend on the context (as time, availability or performance), the user identity and the allocated resource.

## 3 Environment characteristics

Having a close look, we point out some issues characterizing the grid environment:

**Users**: The grid is often build between multiple domains where administrative authorities may add, remove or even change rights of VO users. This is typically the case when an enterprise allocates some employees for a certain project for a certain time, to transfer, after, some of these to another project. Managing the users and their associated rights to access resources at different sites is not a simple task in such a large scale environment. Knowing all the potential users of a VO a priori is not a valid assumption.

**Resources:** VO resources have a volatile nature as the users. A resource may be added or removed or replaced in an environment where faults are frequent. To overcome these problems, resources are often virtualized [16] and then resource description becomes more important than resource naming. Clients need then to request resources by property ex. capability, quality of service or configuration. Resources properties may be assembled within sets serving as Views for this resource. Views serve better in managing resources by considering their utilization. A device may have different views as for example: a device may be seen as a computing device and a storage device at the same time.

**Reutilization:** the use cases of resources cannot be limited whether from the client point of view (data extraction jobs may convert a computing cluster into a high performance data server) or from the provider point of view to respond to various kinds of problems (participation in different VOs) or maximize revenue (pay per usage). Once again this joins the above idea of resources virtualization. Resources should not be referenced by name but by properties or a view which is more appropriate to the grid usage.

**Multi domain aspect:** The grid is founded on the idea of dispersed users and resources within different administrative domains. Centralizing the administration of a VO implies delegating the management of all the users and resources to a single authority. However, a partner in a VO may be a rival in another, so delegating him the control and management of local resources is often not realistic. Each domain needs to manage its resources locally keeping in mind the supplied services' availability. For example, a storage provider may need to move the data on other storage devices locally for maintenance reasons without referring back to the client. An enterprise needs to add locally an employee to a certain workgroup to use the VO resources without returning back to the other partners. Excluding a central authority, the VO requires scalable multi-administration by the involved partners. By multi-administration we don't mean a domain with multiple administrators having the same rights, but rather an environment with different administrators having each partial administration rights within the whole structure. Though the grid considers inter-domain aspects in the Virtual Organization, however it doesn't specify how to put in place such a space.

**Delegation:** The grid problem definition aims at relieving the user from the burden of resource coordination. To enable coordinated resource sharing, delegation is necessary. Delegation is employed when at runtime a task on a device needs accessing another device on behalf of the user (accessing data, subtasks execution…). The mission becomes complicated when the task is sensible and the third device lies at a third partner's administrative domain. Controlling delegation and the delegated rights is necessary not to have access control security breaches.

**Contexts:** users may be authorized to access resources in certain contexts where outside these contexts access is forbidden. Take for example resources rental for a definite time or even the resources rental within different time periods (ex. from 8h till 15h). These contexts are not limited to temporal constraints but may be spatial (user location) or any other network quality of service constraints.

## 4 Virtual Organization lifecycle

To better show the context where this work is situated we discuss the major steps of the VO creation process (fig.2). Inspired from the Virtual Enterprises (VE) lifecycle presented in [19].

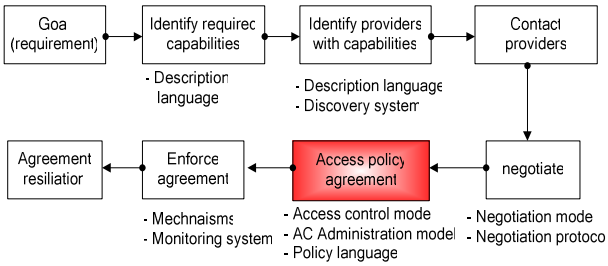


**Fig. 2.** the VO creation process along with the needs under each step

A goal or a requirement is the motivation for launching the process. A goal may be massive data storage, exploiting certain applications or computing resources utilization. To achieve this goal, required capabilities should be identified. These capabilities are the criteria on which the customer bases the choice of a service provider. Considering for example the goal to be data storage, the required capabilities may be a certain storage space and performance statistics to be supplied by the provider. This necessitates a description language to express a wide variety of parameters. This language may be used by the provider also to describe his services. These descriptions may be published on Internet, where clients may use a discovery system to find and contact the potential providers [16]. We will not detail further this subject in this paper to focus on our main interest the access control policy.

In such a distributed system customers need to negotiate access to services based on their requirements and the provider capabilities (ex. quality, price, content, type of allocated servers). Thus, the negotiation includes defining users and shared resources as well as specifying the sharing relationships. The process finally closes with the enforcement of the agreement that assures the runtime management of the virtual organization till the end of its lifetime. As existing solutions for access control management within virtual organization we find the Community Authorizations Service (CAS) and Virtual Organization Management System (VOMS).

CAS [6] allows enforcing access control policies within the VO using X.509 extensions. The extensions include low level user's permission to grid resources as read/write file permissions. This solution was proposed as a mechanism to enforce access control in the Globus toolkit. On the other hand, the Virtual Organization Management System (VOMS) [7], constitutes a server that supplies grid users with X.509 certificates extended with role and group information. The user contacts the server and obtains an X.509 certificate to be used while contacting the resource. The destination resource needs to know the access policies concerning the role/group to make authorization decisions.

GRASP project [8] offers an architecture that uses X.509 certificates to create secure groups (Closed Collaboration Teams) within the lifetime of the Virtual Organization. The VO itself is static, with out-of-band static policy management. The works, shown above, in the access control domain fall within the "Enforce agreement" block of our life cycle (figure 2). These mechanisms refer to certain access control policy however the works don't go farther to specify this policy and its different actors. The boundaries of a VO are defined by the sharing relationships which are access control policies specifying permissions/prohibition within the VO.

The dynamicity feature of the entities within the virtual organization makes specifying the sharing relationships burdensome. The sharing relationships should be independent of the underlying physical entities by referring to the users function [11][13] and the resources properties [12]. The role of each partner in producing this policy becomes a part of the distributed access control management [13][14].

## 5 Access control model for VO

The model to be used should take into consideration the Grid environment constraints explained in the above sections. In addition to the contextual policy, the model should represent the different users and resources in such a way that these dynamic entities don't affect the policy validity. This can be done by abstracting the different entities where the management model should attribute to each side the right to manage his users and/or resources (Fig 3).

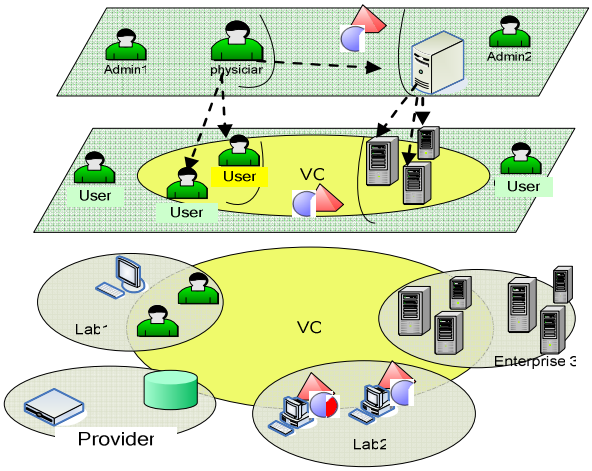


**Fig. 3**. VO access control relations

As traditional access models we find the DAC (Discretionary Access Control) [10] with its simple form the access matrix. It gives users rights to do actions on certain objects. However this model neither offers the needed abstraction nor models the different administration tasks. On the other hand, MAC models (Mandatory Access Control) propose centralized management which is not convenient for a distributed multi-administrated environment [10]. Both of these models don't treat contexts.

RBAC [11][13] introduces the "role" notion which represents a function in an organization (ex. engineer, administrator). A subject is attributed a role which is associated with a set of permissions (or privileges). Using roles to abstract the subjects responds to the grid needs, however at the "object" side RBAC doesn't offer a similar abstraction. Even though, RBAC doesn't prevent such abstraction, but this abstraction will no more be within the same framework. In addition, RBAC doesn't express contextual permissions or interdictions [12] which are important in the grid environment for example: role "engineer" has the right to use application "appl. 1" for 3 hours.

DRBAC (distributed role based access control) [9] is a derivative of RBAC model proposed for scalable decentralized access control that spans multiple administrative domains. Even though the distributed management is one of the grid requirements, DRBAC still suffers from RBAC drawback concerning contexts and resources abstraction.

CBAC (Coalition based access control model) [18] is envisaged to treat access control in coalitions where each must be able to share specific resources while ensuring that these resources are safe from inappropriate access. It captures inter-organization relationships and contexts however the administration model is not yet complete to model multiple administrators that each is responsible for a certain part of the system.

# 6 Or-BAC

In the grid environment each partner has resources that are put in common to be shared by the community. What we need is an access control model that indicates: who can do what in which context. Or-BAC [12] [14] access control model is proposed for modeling a security policy that is not restricted to static permissions and include contextual rules related to permissions, prohibitions and obligations. It aims at introducing an abstraction level that separates access control policy from its implementation.

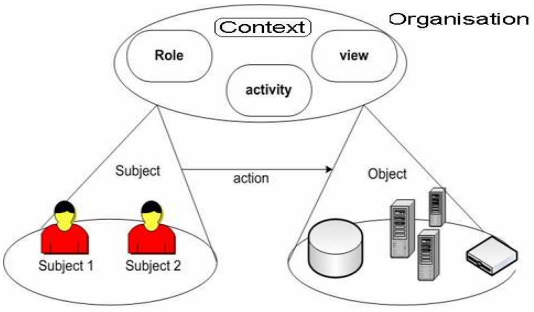


**Fig. 4.** Or-BAC access control model

Or-BAC model offers a flexible mean to model such an environment employing abstract notions as "role, activities and views" to abstract the traditional "subject action object" (figure 4). We recall here the principal notions in this model:

- Organization: it is an organized group of active entities, i.e. subjects, playing some role or other. Our virtual organization may be modeled as an organization in Or-BAC model since it is the result of a group of entities each playing a role (ex. Provider, consumer, user, resource...) within the whole structure.

- Role: It corresponds to certain privileges in an organization. The entity Role is attributed to a subject within an organization. Abstracting users using roles facilitates user management in a large scale dynamic environment. It is sufficient to assign/revoke a role to a new/departing user. Policy expressed in terms of roles stay valid even if users change.

- View: a view corresponds to a set of objects that satisfy a common property (storage device, databases, files etc). This notion may abstract the VO resources according to their usage. This is more adapted to the on demand resource allocation where the user may not be interested by the device itself but by certain characteristics as the storage capacity, computing power…

- Activity: It is used to abstract actions as "read", "write". An activity may correspond to a combination of actions. Activities notion may be extended further towards workflows that may be easier to deal with at the business level.

- Context: according to certain conditions with certain temporal or spatial parameters (ex. Secure connection, daytime, time), the role will be authorized to launch activities. These extra conditions are Contexts.

Or-BAC with these notions responds to the grid requirements. We may consider a Virtual Organization to be an organization in Or-BAC. Entities in the virtual organization would be roles instead of users, and views instead of objects. The large scale problem of the grid (users, resources) is overcome by the offered abstraction. On the other hand, the multi-administration is also taken into consideration as we will see in the following sections.

### 6.1 Or-BAC relationships

There are eight basic sets of entities: Org (organization: an organized group of subjects, playing some role within the group), S (a set of subjects), A (a set of actions), O (a set of objects), R (a set of roles), $\mathcal{A}$ (a set of activities), V (a set of

views), C (a set of contexts). Or-BAC considers that org $\subseteq$ S, S $\subseteq$ O. Any entity may have attributes, for instance if S is a subject, then name(S), address(S) represents the name and the address of subject S. Or-BAC also defines relations between these sets:

- Empower is a relation over domains *Org x S x R*. *Empower (org, s, r)* means that *org* empowers subject *s* in role *r*.
- Use is a relation over domains Org x O x V. *Use (org, o, v)* means that *org* uses object *o* in view *v*. This helps to give different definitions for the same view in different organizations.
- Consider is a relation over domains Org x A x $\mathcal{A}$. *Consider (org, $\alpha$, a)* means that org considers that actions $\alpha$ falls within the activity *a*.
- Define is a relation over domains Org x S x A x O x C. *Define (org, s, $\alpha$, o, c)* means that within organization *org* context *c* holds between subject *s*, action $\alpha$ and object *o*.
- Policy definition: access control policy is defined by the relation permission. Permission is a relation over domains Org x R x $\mathcal{A}$ x V x C, *Permission (org, r, a, v, c)* means that organization *org* grants role *r* permission to perform activity *a* on view *v* within context *c*.

We consider that the grid Virtual Organization is an organization Org in the Or-BAC context. To construct this space, add users, attribute users to roles and resources to views and specify access permissions, a management model is needed. It should control the activities: management of organizations, management of roles, activities, views and contexts, assignment of users to roles, assignment of permissions to roles, assignment of users to permissions.

### 6.2 Or-BAC administration model (AdOrBAC)

AdOrBAC is the administration model of Or-BAC. It defines special views as URA (user role assignment), PRA (permission role assignment)... Objects belonging to these views have special semantics, namely they will be respectively interpreted as an assignment of user to role and permission to role. Intuitively inserting an object in these views enables an authorized entity to respectively assign a user to a role and a permission to a role. Conversely, deleting an object from these views will enable a user to perform a revocation. Distributing the administration responsibilities among the grid partners consists of defining which roles are permitted to access the views URA, PRA … or to more specific views when the role has not complete access to one of these views (Fig 5).

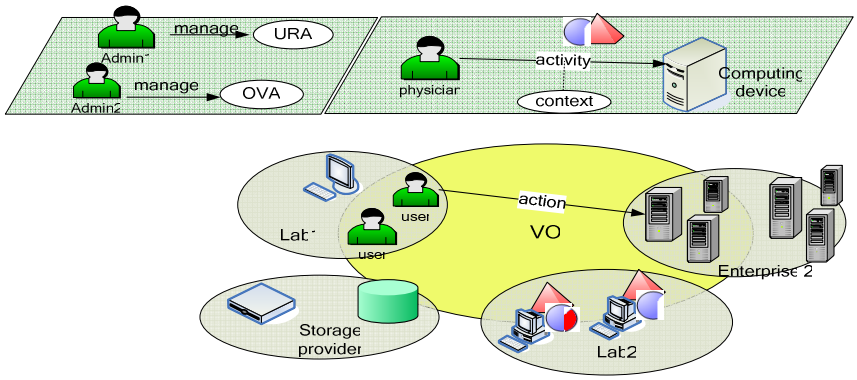


**Fig. 5.** OrBAC and AdOrBAC model

**URA in AdOr-BAC.** It is used to determine who is allowed to assign a user to a role and on which conditions. Assigning a user to a role equals adding a new object in a given view called URA (fig 6). Three attributes are associated with this object:

- *Subject* to designate the subject which is related to the assignment.
- *Role* that corresponds to the role to which the subject will be assigned
- *Org* to represent the organization to which the subject is assigned.

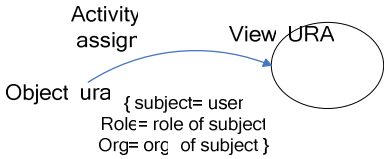


**Fig. 6.** user role assignment

So to add a user to a special role "physicist" in the physics department "*phy-dept"* at organization "Lab", we have to create a view "URA- physicist - phy –dept" defined as follows:

*∀ ura, Use (Lab, ura, URA- physicist - phy -dept) <=> Use (Lab, ura, URA) ^ (role (ura) = physicist) ^ ( org (ura) = phy-dept)*

This way we defined a view for the physicists in the phy-dept. There is a relationship between adding an object to this view and the relation "empower":

*∀ Org, ∀ ura, Use (org, ura, URA ) -> Empower (org (ura), subject (ura), role (ura))*

*"Assign"* is the activity to be given to the administrator to do this assignment job.

*Permission (Lab, administrator, assign, URA- physicist -phy-dept, default)*

This method allows attributing roles administration to a certain administrator whom we grant the permission to assign/revoke users. We may attribute a member of a certain domain the right to add "ura" objects in a view defined of his domain users we may then satisfy the requirement that each domain keeps control on its users.

**PRA in AdOr-BAC.** Permission role assignment follows the same logic as URA concerning adding an object to a view; however the object has 5 attributes:

Issuer= issuing organization
Grantee, privilege, target = role, activity and view concerned by the permission
Context = designate the context in which the rule can be applied.

There is a link between adding an object to this view and the relation permission:

*∀ pra, ∀ org, ∀ context, Use (org, pra, PRA) -> Permission (issuer(pra), grantee(pra), privilege(pra), target(pra), context(pra))*

**View Object Assignment.** In analogy to URA [14], we would like to define a view including certain collection of objects and attribute the manage activity to an administrator (ex. View-admin). Associating a user with the role *View-admin*

*Empower (org, user, View-admin)*

Attributing permission *manage* to the role *"View-admin":*

*Permission (org, View-admin, manage, VOA-org2, default)*

Where *VOA-org2* is the view of a group of resources having an attribute *org* that is equal to org2, it can be defined as follows:

*∀ voa, Use(org, voa, VOA-org2) <=> Use(org, voa, VOA)^ (org(voa)=org2)*

So finally attributing a resource or object to a view in the organization is equivalent to adding an object voa with attributes (object to be added, view to which it should be added, in org as organization) to the VOA-org2 view. The view-admin is in charge of that.

## 7 Virtual Organization Modeling

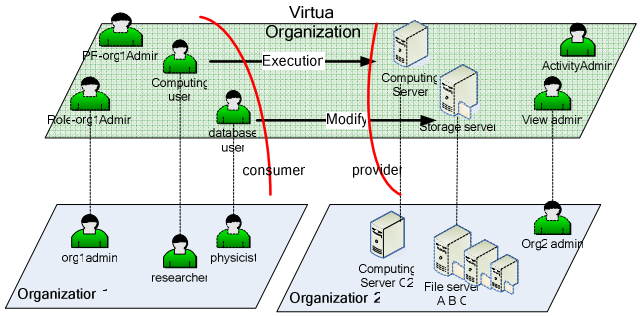


**Fig. 7.** Or-BAC model in Grid environment

Consider the scenario in figure 7 where two organizations need to cooperate forming a Virtual Organization. Consider the simplified case where organization org2 supplies the resources (provider) and org1 contains the users (consumer). Putting in place the VO needs to define the participating entities from both sides including roles, views, activities and the administration authorities.

Modeling this environment using Or-BAC, we start by the management of the virtual organization itself. Considering that VO is an organization in Or-BAC framework, then such an entity may have attributes. As discriminating attributes there are the involved participants, a name discriminating a certain VO within the different VOs that can be set up among the same partners, and may be a certain expiry date after which the cooperation is terminated.

*Name (VO) =cooperation 1*
*Partners (VO) = org1, org2*
*Time= timelimit*

**At the administration plan:** we should define the relevant roles in the Virtual Organization as:
*Relevant-role (VO, Role-org1Admin); Relevant-role (VO, PR-org1Admin); Relevant-role (VO, ViewAdmin); Relevant-role (VO, ActivityAdmin).*

In the same way, the relevant views and the Relevant activities names should be defined for the VO.
To attribute to a certain "*org1admin*" the role "*Role-org1Admin*" that has the responsibility to *assign/revoke* roles to the users of organization org1:

*Empower (VO, org1admin, Role-org1Admin)*
*Permission (VO, roleadmin, manage, URA- org1,default)*

Where URA-org1 is defined as:
*∀ Ura, Use (VO, ura, URA-org1) <=> Use (VO, ura, URA) ^ org (ura) = org1*

To attribute to a certain "*org1admin*" the role "*PR-org1Admin*" that has the responsibility to assign/revoke permissions to roles of organization org1:

*Empower (VO, org1admin, PR-org1Admin)*
*Permission (VO, PR-org1Admin, manage, PRA-org1, default)*

Where PRA-org1 is defined as:
*∀ pra, Use (VO, pra, PRA-org1) <=> Use (VO, pra, PRA) ^grantee(pra)∈ {r/ ∃ ura∈ URA-org1, role(ura)=r}^ privilege (pra) ∈ activities(VO)*

To attribute to a certain "org2admin" the role *"ViewAdmin"* that has the responsibility to assign/revoke views to objects of organization org2:
*Empower (VO, org2admin, View-org2Admin)*
*Permission (VO, View-org2Admin, manage, VOA-org2, default)*

Where *VOA-org2* is defined as:
*∀ voa, Use (VO, voa, VOA-org2) <=> Use (VO, voa, VOA) ^ org (voa) = org2 ^ view(voa) ∈ views(VO)*

To attribute to a certain *"org2admin"* the role *"ActivityAdmin"* that has the responsibility to assign/revoke activities to actions comprehensible by organization org2 and its different resources:

*Empower (VO, org2admin, ActivityAdmin)*
*Permission (VO, ActivityAdmin, manage, AaA-org2, default)*

Where *AaA-org2* is defined as:
*∀ aaa, Use (VO, aaa, AaA-org2) <=> Use (VO, aaa, AaA) ^ org (aaa) = org2*

**Assigning concrete level to the abstract level:** according to figure 5 authorized roles/users Role-org1Admin has the permission to empower a user with a role as in the following rules:
*Empower(VO, researcher, computinguser)*
*Empower(VO, physicist, databaseuser)*

ViewAdmin has the permission to use an object in a view as in the following rules:

*Use(VO,storageserver,FileserverA&FileserverB&FileserverC)*
*Use(VO,computing server, computingserverC2)*

ActivityAdmin has the right to consider an action within an activity as in the following rules:
*Consider (VO, execute, Execution)*
*Consider (VO, write, Update)*
*Consider (VO, read&write, Modify)*

PR-org1Admin has the right to permit a role to do an activity on a view as following:
*Permission(VO, database user, Modify, storage server, workTime)*
*Permission(VO, computing user, Execution, computing server, day&night)*

This way using Or-BAC we modeled our grid environment abstracting the users and resources within a single framework. This framework allows different partners to participate in specifying the access control policy keeping the autonomy and flexibility in managing their local assets. This formalism should be employed within a cooperation and negotiation framework to put in place a multi administrated VO.

**Delegation:** Modeling delegation is not shown yet. We will pass in general to see how is the demarche; however we will not enter in details being limited in this paper. Delegation in the grid context is to give some device working on behalf of a user some rights to enable it access other resources. This set of rights is normally a subset of the set of rights of the user himself (Fig 8).

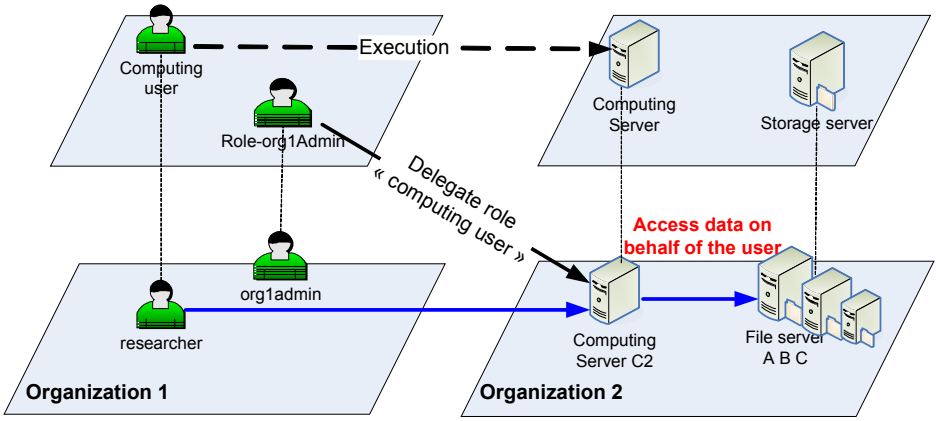


**Fig. 8.** Delegation modelling

Delegation may be supported by the model when the internal policy of an organization authorizes giving the administrator role the right to assign roles for subjects in other domains for a special task. So to model delegation there should be a view containing the delegated entities and the right to give them roles or sub roles in order to complete the assigned task. This way we can control delegation which depends on the policy of the client as well as the provider. The client may sometimes not trust the object and thus doesn't delegate it. The same can be said about the provider who may not accept that his resources go farther to propagate anonymous malicious activities for example (Distributed Denial of Service attacks).

## 8 Modeling issues

It is necessary to refer back to the set of constraints defined above in section 3 and relate them to the model terms. Abstraction of entities at the VO level allows having two distinct levels: valid policy and dynamic entities. On the other hand, every partner has his settled internal access control system. It is not desired to modify this system (users, roles, privileges…) each time the partner adheres to a different virtual organization. Though the activities of user in the virtual organization may reveal the hierarchy and the different responsibilities between the involved users, however this should not compromise the internal organization structure. The different privileges given to users at the virtual organization level should be subject to the intra-organization management. The user role assignment policy is an internal issue according to which the organization attributes a role to a user.

The same discussion can be employed to argue that the OVA policy also should be internal to the provider domain. However this internal policy should take into consideration the commitments of the provider to supply a specific service with particular characteristics. Following this logic we attribute the URA to the organization of the user and the VOA to the organization under which the objects are administered. We base on a pre-virtual organization negotiation to give certain roles (ex. Role admin, View admin) certain privileges on certain resources within certain contexts. These contexts may depend on specific partner parameters (resource quality of service, rental time, and specific periods of time…).

## 9 Conclusion and future works

The grid has a goal enabling coordinated resource sharing and problem solving in dynamic multi-domain virtual organization. For this reason it sets up a virtual organization which constitutes the shared common space between the different partners. However the grid doesn't indicate how to construct such a space in a dynamic, multi-administrated

environment. On the way to dynamically create virtual organizations we had to find a model that organizes the different responsibilities in a VO. It should take into consideration the environment constraints as multi-organizational aspect, large scale, dynamic resource allocation… We chose Or-BAC for this mission which shows flexibility to model these situations. Or-BAC models a multi-administered environment using the "role view activity" abstraction which abides to the other constraints (large scale and dynamic resources). With this abstraction Or-BAC policy rules are independent of the physical underlying infrastructure. What is still need to be detailed in our model is the delegation aspect which is necessary for the domain of ubiquitous computing. We will treat this issue independently [15].

The next step after modeling is to implement the automated creation of such environment by discovery, negotiation, and generation of an SLA-like policy [8] which may be powered with QoS parameters along with the access control ones. Finally this policy needs enforcement using appropriate mechanisms within the grid layers [5].

## References


1. B. Nasser, A. Benzekri, R. Laborde, F. Grasset, F. Barrère. "Access Control Model for Grid Virtual Organizations", to appear in ICEIS conference, 2005.

2. Grid Support for Ubiquitous Computing Research Group Global Grid Forum. http://ubigrid.lancs.ac.uk/ubicomp_rg_ charter.html

3. Gilles Fedak, Cecile Germain, Vincent Neri, and Franck Cappello, 2001. XtremWeb: A Generic Global Computing System. CCGRID2001, workshop on Global Computing on Personal Devices, May 2001, IEEE Press.

4. Arcot Rajasekar, Michael Wan, Reagan Moore, Wayne Schroeder, George Kremenek, Arun Jagatheesan, Charles Cowart, Bing Zhu, Sheau-Yen Chen, Roman Olschanowsky. Storage Resource Broker – Managing Distributed Data in a Grid.
Available online: http://www.npaci.edu/DICE/Pubs/CSI-paper-sent.doc

5. Ian Foster, Carl Kesselman, Steven Tuecke, 2001. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. In, International J. Supercomputer Applications,15(3),2001.

6. Cannon S., Chan S., Olson D., Tull C., Welch V., Pearlman L., 2003. Using CAS to manage Role based VO sub-groups. In CHEP 2003, La Jolla, California, available online:
http://www.globus.org/security/CAS/Papers/CAS-group-CHEP03.pdf

7. Alfieri R., Cecchini R., Ciaschini V., dell'Agnello L., Frohner A., Gianoli A., L″orentey K., and Spataro F., 2003. "VOMS, an authorization system for virtual organizations", DataGrid Project, available online: http://grid-auth.infn.it/docs/VOMS-Santiago.pdf

8. Djordjevic I., Dimitrakos T., Phillips C. An architecture for dynamic security perimeters of virtual collaborative networks. Proc. 9th IEEE/IFIP Network Operations and Management Symposium, (NOMS 2004), April 2004. IEEE-CS.

9. Eric Freudenthal, Tracy Pesin, Lawrence Port, Edward Keenan, and Vijay Karamcheti. dRBAC: Distributed Role-based Access Control for Dynamic Coalition Environments

10. Pierangela Samarati, Sabrina De Capitani di Vimercati. Access Control: Policies, Models, and Mechanisms.

11. Ravi Sandhu, Edward Coyne, Hal Feinstein, Charles Youman, 1996. Role-Based Access Control Models. IEEE Computer, vol. 29, n° 2, pp.38-47, février, 1996.

12. Anas Abou El Kalam, Rania El Baida, Philippe Balbiani, Salem Benferhat, Frederic Cuppens, Yves Deswartes, Alexandre Miege, Claire Saurel, Gilles Trouessin , "Organization Based Access Control" , available online:
http://www.rennes.enst-bretagne.fr/~fcuppens/articles/Or-BAC.pdf

13. Sandhu R., Munawer Q., 1999. The ARBAC99 Model for Administration of Roles. In Proceeding of the 15th Annual Computer Security Applications Conference (ACSAC'99), Phoenix, Arizona, 6-10 December 1999, IEEE Computer Society, pp. 229-241.

14. Frederic Cuppens, Alexandre Miege. Ad-ORBAC: An Administration Model forOr-BAC. Available online: http://www.rennes.enst-bretagne.fr/~fcuppens/articles/csse04.pdf

15. Welch, V., Foster, I., Kesselman, C., Mulmo, O., Pearlman, L., Tuecke, S., Gawor, J., Meder, S. and Siebenlist, F. (2004). X.509 proxy certificate for dynamic delegation, Proceedings of the 3rd Annual PKI R&D Workshop.

16. Karl Czajkowski, Ian Foster, Carl Kesselman, Volker Sander, Steven Tuecke, 2002. “SNAP: A Protocol for Negotiation of Service Level Agreements and Coordinated Resource Management in Distributed Systems”. Draft submission to JSSPP’02 April 30, 2002. Available online: http://www-unix.mcs.anl.gov/~schopf/ggf-sched/GGF5/sched-GRAAP.3.pdf

17. Ian Foster, Carl Kesselman, 1997. Globus: A Metacomputing Infrastructure Toolkit. Intl J. Supercomputer Applications, 11(2):115-128.

18. E. Cohen,R. Thomas, W. Winsborough, D. Shands. Models for coalition based access control (CBAC).

19. Nitin Nayak, Tian Chao, Jenny Li, Joris Mihaeli, Raja Das, Annap Derebail, Jeff Soo Hoo, “Role of Technology in Enabling Dynamic Virtual Enterprises”. Available online: http://cersi.luiss.it/oesseo2001/papers/13.pdf