CERIAS Tech Report 2004-71

A FRAMEWORK FOR CONTRACTUAL RESOURCE SHARING IN COALITIONS

by S. Sadighi Firozabadi, A.C, Squicciarini, M.Sergot, E. Bertino

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907-2086

A Framework for Contractual Resource Sharing in Coalitions

Babak Sadighi Firozabadi*
Policy Based Reasoning Group
Swedish Institute of Computer Science (SICS)
Box 1263, SE-16429 Kista, Sweden
babak@sics.se

Marek Sergot
Department of Computing
Imperial College London
180 Queen's Gate, London SW7 2BZ, UK
mjs@doc.ic.ac.uk

Anna Squicciarini† Elisa Bertino
Dipartimento di Informatica e Comunicazione
Università degli Studi di Milano
Via Comelico 39/41, 20135 Milano, Italy
{squicciarini,bertino}@dico.unimi.it

Abstract

We develop a framework for specifying and reasoning about policies for sharing resources in coalitions, focussing here on a particular, common type of contract in which coalition members agree to make available some total amount of specified resource over a given time period. The main part of the framework is a policy language with two basic elements: 'obligations' (of a member enterprise to provide a total amount of resource over a given time period) express the coalition policy, and 'entitlements' (granted by an enterprise to other coalition members) express the local policies of the coalition members. We discuss the conditions under which a local policy can be said to be in compliance with, or meet, the obligations of a coalition policy, and the conditions under which an obligation, and by extension a contract, can be said to be violated or fulfilled.

1 Introduction

There is a growing interest in facilitating collaboration between independent and heterogeneous enterprises, in the form of coalitions. One of the reasons behind creating such coalitions is to share different types of resources that are owned and independently managed by the coalition members. The set of resources are made available by participating enterprises which agree on sharing them according to a *coalition policy* expressed in terms of a *contract* between the coalition members. The coalition policy defines the agreed rights of the coalition members to use others' resources and their obligations to provide their own resources to other coalition members.

The aim of the paper is to develop a framework for specifying and reasoning about policies for sharing resources in coalitions. We focus on a particular, common type of contract in which coalition members agree to make available some total amount of specified resource over a given time period. The main part of the framework is a policy language with two basic elements: 'obligations' (of a member enterprise to provide a total amount of resource over a given time period) express the coalition policy, and 'entitlements' (granted by an enterprise to other coalition members) express the local policies of the coalition members. We discuss the conditions under which a local policy can be said to be in compliance with, or meet, the obligations of a coalition policy, and the conditions under which an obligation, and by extension a contract, can be said to be violated or fulfilled.

2 Overview

In this section we give an overview of a framework for contractual sharing of computational resources in a coalition. The idea is that coalition members, which in our case are enterprises, can share resources among each other according to a coalition policy which can be seen as a contract between the coalition members. We see one member's

*The work by Sadighi Firozabadi was partly supported by the Policy Based Management project funded by the Swedish Agency for Innovation Systems.

†The work by Squicciarini was performed during her stay at the Swedish Institute of Computer Science (SICS).

Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)
0-7695-2141-X/04 $ 20.00 © 2004 IEEE

obligation to provide a resource to another as the second member's right to access/use that resource. As a coalition member an enterprise will gain access to the resources of others and at the same time will have to release its own resources for use by the other members, according to the *rules of sharing* stated in the contract.

Resources in a coalition are shared but are still managed independently by their owners. Each member of a coalition has its own *local policy* which specifies how it intends to grant access to its resources. An enterprise may change the terms of use of its resources, in its local policy, to optimize the resource usage as time passes.

Each enterprise as a member of a coalition must publish a local policy that complies with the contract of the coalition. This local policy must be available for other coalition members to examine, in order that they may be able to plan how to make use of the shared resources available. It is a separate question whether an enterprise will in fact comply with its own local policies. An enterprise might promise to provide a total amount of resource that exceeds what it can actually deliver. An enterprise may also be a member of several coalitions, and it might produce several local policies for its resources, each compliant with the contract of one of the coalitions to which it belongs, but without being able to comply with all the coalition policies at the same time. This is similar to the way flight companies sell tickets to more people than they have available seats.

Although a member publishes a local policy specifying how it will make its resources available, it is still possible in practice that it will deny access to its resource upon a request. If a request to access a resource is not granted, although specified as available in the local policy, then the enterprise violates the corresponding obligation in the coalition policy. As a consequence of the violation, the enterprise must usually accept another obligation to be fulfilled, or in the absence of such an obligation, it will violate the entire contract. The assumption is that members always have an incentive to continue being members of coalitions, and hence they will avoid, as far as they are able, any breach of contract. In the case that a contract is violated, one can expect that some kind of punitive actions may take place, for instance, expulsion of the defaulter from the coalition. These further considerations however will be outside the framework presented in this paper.

It is important to note that all the member interactions are carried out without centralized control. Thus, a key issue of the proposed framework concerns monitoring systems. The framework provides both monitoring systems and enforcement mechanisms at different stages of the coalition life. First, a mechanism for verifying that local policies satisfy coalition policies is devised. Then, a level of monitoring is performed for controlling the actual granting of access requests. It is reasonable to assume that if a request is granted then the resource is actually allocated for the requester to use. To make this explicit, the granting of a request can be in the form of a signed certificate stating that the requesting agent is entitled to access/use the resource, which in our case is the same as allocating the resource for use by the agent. In this way, we factor out of consideration the possibility that a request is granted but the resource or the promised level of quality is subsequently not provided. A system to enforce this would be one which has a central reference monitor controlling access to the coalition resources, even though these are owned by different coalition members. Note that the resources are released by the reference monitor, only if the user can show a valid certificate issued by the resource owner stating its right to use the resource. It is up to the resource owner to keep track of the use of its resource and it cannot grant access to a resource that is already in use. We do not consider the issue of blocking granted requests in this paper, but of course one can think of a system supporting this as a way of releasing resources that are in use in order to fulfill certain obligations as a cost of violating others.

As a running example, we consider an academic alliance ($\mathcal{AA}$, in what follows) involving a number of different systems cooperating on a common research project. The entities involved are, let us say, three universities ($Milano$, $Imperial$, KTH), two research centres ($SICS$, CNR), and two business companies (*ABCent*, *Micro*) sharing resources and services. We will use this scenario throughout the paper to show how the framework can be applied.

3 Formal Framework

In this section we formalize the ideas described informally in the previous section. We start by defining what a coalition consists of and its basic elements. We develop a policy language for describing local and coalition policies. We also define how a local policy conforms to an obligation and how it conforms to a coalition policy.

Definition 3.1 (Coalition Elements) The basic elements of a coalition are:

1. $\mathcal{R}$ denotes the set of resources shared among the coalition.
2. *Users* denotes the set of users of the resources pooled in the coalition. We assume only that each user is somehow uniquely identified, for example by its public key.
3. $\mathcal{E} = \{E_1, \ldots, E_n\}$ denotes a finite set of enterprises, the members of the coalition. Each enterprise has associated with it a (possibly empty) set of users, as given by the function $users\colon \mathcal{E} \rightarrow \wp(\mathit{Users})$. $users(E)$ is the set of users who belong to enterprise E. The same user can belong to several different enterprises.

Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)
0-7695-2141-X/04 $ 20.00 © 2004 IEEE

4. Pol denotes the set of policies regulating the sharing of the resources available in the coalition. Pol conveys the coalition policy *Contract* as well as the local policy $\mathbf{LP}_E^t$ of each member enterprise $E \in \mathcal{E}$ at each time point t. For simplicity, in this paper we assume that the coalition policy *Contract* does not vary over time.
5. Each member enterprise maintains an accounting/monitoring system which keeps track of all usage of its resources and all requests for access submitted to it by other coalition members. We assume that the coalition also has access to this accounting information, either by obtaining it from member enterprises, or by maintaining its own independent monitoring and accounting system where member enterprises cannot be trusted to supply the information reliably. The minimum requirements for the accounting system are described in section 3.3.

Resources are provided to users as member of enterprises. Here, we do not assume anything about the internal structure and behaviour of the users in a given enterprise, and consider them only as agents executing actions.

3.1 Resources

A key element in characterizing a coalition is the set of resources to be shared by coalition members. To be correctly enforced, policies must be specified according to the specific features of the resources to which they refer. Resource descriptions therefore must be expressed in terms of measurable parameters. For example, the resource type *bandwidth* can be specified by the parameters *amount, duration-time, latency*. We use a generic scalar metric to specify parameters and allow them to be composed to model resource features. Resource requirements can be expressed either by exactly specifying resource capacity or by constraining the ranges of capacity. Resources may be specified by the following type of metrics, similar to those used in [4]:

— **Time Metrics** where time points are expressed as a combination of date and clock time, e.g. $(22\text{-}09\text{-}2003, 13{:}54{:}23)$. For the examples in this paper, time points will be measured to the second, though of course this is very easily changed to the needs of an actual application.

— **Scalar Metrics** given as some suitable numerical value depending on the type of resource, such as integers for the amount of disc-space (50Gb).

— **Max limit–Min limit** specifying an exclusive or inclusive upper/lower limit on the given metric.

We assume that each resource is specified by means of a set of characterizing attributes, each expressed using one of the above metrics. This way of specifying resources defines an ontology for formally checking whether a resource is properly released, meeting the parameters describing the resource itself.

Definition 3.2 (Resource comparison) Let $ResType$ be a resource type, measured according to some suitable metric m. Let R' and R'' be two instances of resource type $ResType$. $R' \sqsubseteq_m R''$ denotes that the two instances R' and R'' of resource type $ResType$ are comparable and that R' is less than or equal to R'', according to metric m.

In what follows the subscript m is often omitted where context allows.

For instance, where bandwidth resource is characterized by the attributes $\{$*amount, duration-time, latency*$\}$ we can write, e.g., $\{10Gb, 10h, 1.5ms\} \sqsubseteq_{bw} \{12Gb, 18h, 1.5ms\}$. Some amounts are not comparable. $\{10Gb, 10h, 1.5ms\}$ is not comparable with, e.g. $\{8Gb, 18h, 1.5ms\}$.

3.2 Obligations and Entitlements

The two basic elements of our policy language are *obligation* and *entitlement*. Here, we give formal definitions of these two notions. In the next section we will define how obligations can be fulfilled and violated.

Definition 3.3 (Obligation) An obligation for an enterprise P to provide a total amount R of a certain resource over the time interval I to each of a set S of enterprises is denoted as $Obl(P, S, R, I)$ where:

— $P \in \mathcal{E}$ denotes the *bearer* of the obligation, also referred to as the 'providing enterprise' or simply 'the provider';

— $S \subseteq \mathcal{E}$ is the set of enterprises to whom the obligation is owed;

— R is a specified amount of a resource in $\mathcal{R}$;

— I denotes the time period of the obligation, expressed as a time interval $I = [t_{start}, t_{end}]$.

The idea is that there is a total amount R of resource available over the time interval I. There is no restriction on how R will be claimed: it can be claimed all at once, or piecemeal. The policy language could be extended to specify also minimum and maximum amounts of R that can be claimed at any one time but we shall not do so here.

The intended reading is that obligation $Obl(P, \{E_1, \ldots, E_k\}, R, I)$ is equivalent to a collection of separate obligations $Obl(P, \{E_1\}, R, I)$, $\ldots$, $Obl(P, \{E_k\}, R, I)$. It is also possible to extend the policy language by introducing another form of obligation $CollObl(P, S, R, I)$ to represent an obligation owed by P to the set of enterprises S collectively, that is to say, to represent that enterprises S *share* access to P's

Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)
0-7695-2141-X/04 $ 20.00 © 2004 IEEE

resource R over interval I. The details for the $CollObl$ form of obligation are not difficult but are sufficiently complicated that we omit them here for brevity. (There is in any case no loss of expressivity, since if necessary we can introduce another 'artificial' enterprise E_S such that $users(E_S) = \bigcup_{E \in S} users(E)$.)

Example 1
$Obl(SICS, \{Milano, KTH\}, \{10Gb, 600h, 1.5ms\},$
$[(12\text{-}07\text{-}2003, 10{:}10{:}50), (02\text{-}12\text{-}2003, 10{:}23{:}30)])$
is an example of an obligation, stating that SICS is obliged to provide at least 10Gb of network bandwidth for a total of 600 hours over the specified time period to each one of Milano and KTH universities.

Definition 3.4 (Entitlements) Entitlements are expressions of the form $Ent_P(E, R, I)$ where:

— $P \in \mathcal{E}$ is the granter of the entitlement;
— $E \in \mathcal{E}$ is the enterprise to which the entitlement is granted;
— R is a specified amount of a resource in $\mathcal{R}$;
— I denotes the time period of the entitlement, expressed as a time interval $I = [t_{start}, t_{end}]$.

An entitlement can be regarded as a promise by an enterprise to grant access to its resource. By publishing the entitlement $Ent_P(E, R, I)$ in its local policy, the enterprise P promises that over the time interval I a total amount R of the resource will be released on request to any user belonging to the entitled enterprise E.

As in the case of obligations, it is possible to introduce another form of entitlement $SharedEnt_P(S, R, I)$ to represent that the set of enterprises S together *share* an entitlement to P's resource R throughout interval I. Again, we omit the details for brevity.

The first step for an enterprise to fulfill an obligation is to specify a local policy that is in compliance with the obligation, or, as we shall also say, one that *meets* the obligation. This step is a necessary but not sufficient condition for fulfillment of the obligation: it does not mean that the resource is actually allocated, but merely that the provider has published a plan specifying how it intends to fulfill its obligation. This plan may change as time goes on.

We now define the conditions under which a set of entitlements is *in compliance with*, or meets, an obligation. The key point is that $Obl(P, \{E_i\}, R, I)$ allows users in E_i to take up to the full amount R of the resource from P over the time period I, with no restriction on how much of the resource is requested at any time, up to the limits of the obligation. Users in E_i can take R at any time over I—all at the same time, spread evenly across the interval, or in whatever portions they choose. So in order to meet its obligation, P must publish entitlements in its local policy in such a way that all of E_i's possible uses are accommodated. Clearly, an entitlement $Ent_P(E_i, R', I')$ meets this requirement when $I' = I$ and $R \sqsubseteq R'$. But suppose that I' is a (proper) subinterval of I. Then $Ent_P(E_i, R', I')$ would not, by itself, meet the obligation since it restricts an entitled user to access R only during the sub-interval I'. A set of entitlements could meet the obligation, however, if together they cover the entire obligation interval I.

What if the entitlement interval I' extends beyond the obligation interval I? In this case, an entitled user is able to access the full amount R of obligated resource over the interval I, as long as I' does not start earlier than I (because in that case, some of the entitled resource may have been used up already before the obligation interval I commences).

Finally, we do not want to allow separate entitlements to be accumulated over the interval I in order to meet an obligation. For example, if there is an obligation to provide 100Gb of disk storage over the interval $[1, 10]$ (here and later we sometimes use integers for time points to reduce clutter), two separate entitlements of 50Gb each over $[1, 5]$ and $[6, 10]$ would not meet the obligation. Even though the sum total of entitled resource over the interval $[1, 10]$ meets the required 100Gb, two separate entitlements impose restrictions on how the entitled 100Gb can be accessed and thus do not meet the obligation. The only way that two separate entitlements over the intervals $[1, 5]$ and $[6, 10]$ could meet the obligation is if both granted at least 100Gb of disk storage each. Anything less would impose a restriction on how the 100Gb resource is accessed over $[1, 10]$.

This may seem like a very strict requirement, but the point is that if two separate entitlements of 50Gb each over time intervals $[1, 5]$ and $[6, 10]$ are to be regarded as meeting the agreed obligation, the provision that should have been specified in the contract is an obligation to provide 50Gb over time interval $[1, 5]$ and an obligation to provide 50Gb over time interval $[6, 10]$, not one single obligation for 100Gb over $[1, 10]$.

A case could be made for allowing entitlements for the same resource type which span the obligation interval to be summed up: one could say that an entitlement of, e.g., 70Gb over $[1, 10]$ combined with another entitlement of, e.g., 30Gb over $[1, 10]$ together would meet an obligation to provide 100Gb over $[1, 10]$. However, this possibility raises a number of further potential difficulties which still remain to be resolved and we therefore do not support it in this version of the framework.

The definition we use is as follows.

Definition 3.5 (l-cover) Let $I = [t_s, t_e]$ and $I_1, \ldots I_k$ ($k \geq 1$) be (closed) time intervals. $\{I_1, \ldots I_k\}$ is a *l-cover* for the interval I iff

i) $t_s \leq t'_s \leq t_e$ for every interval $[t'_s, t'_e]$ in $\{I_1, \ldots I_k\}$;
ii) for every time point $t \in I$ there is an interval I_i in $\{I_1, \ldots I_k\}$ such that $t \in I_i$.

Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)
0-7695-2141-X/04 $ 20.00 © 2004 IEEE

Condition (ii) requires that the collection of intervals $\{I_1, \ldots I_k\}$ together cover the entire interval I. Condition (i) requires that none of the intervals in $\{I_1, \ldots I_k\}$ start earlier than the interval I.

Definition 3.6 (Obligation compliance) Let $\mathbf{E}_P$ be a set of entitlements granted by enterprise P. Let $S \subseteq \mathcal{E}$ be a set of enterprises. The set of entitlements $\mathbf{E}_P$ is in compliance with an obligation $Obl(P, S, R, I)$, denoted by

$$\mathbf{E}_P \textbf{ meets } Obl(P, S, R, I)$$

iff, for every enterprise $E \in S$, there exists a set of entitlements $\{Ent_P(E, R_1, I_1), \ldots, Ent_P(E, R_k, I_k)\} \subseteq \mathbf{E}_P$ such that $\{I_1, \ldots, I_k\}$ is a l-cover for I and $R \sqsubseteq R_i$ for every $i \in 1..k$.

Example 2
Consider the following set of entitlements granted by research centre SICS:

1. $Ent_{SICS}(Milano, \{18Gb, 610h, 1.5ms\},$ $[(12\text{-}07\text{-}03, 10{:}10{:}50), (20\text{-}10\text{-}2003, 08{:}43{:}30)])$
2. $Ent_{SICS}(Milano, \{12Gb, 650h, 1.5ms\},$ $[(20\text{-}10\text{-}03, 08{:}43{:}31), (02\text{-}12\text{-}2003, 10{:}23{:}30)])$
3. $Ent_{SICS}(KTH, \{10Gb, 600h, 1.5ms\},$ $[(12\text{-}07\text{-}03, 10{:}10{:}50), (02\text{-}12\text{-}2003, 10{:}23{:}30)])$

This set of entitlements is in compliance with the obligation of Example 1. Entitlements 1 and 2 together cover the obligation interval and each provides (more than) enough resources to meet the specified requirements for Milano. Entitlement 3 covers the obligation interval and meets the specified requirements for KTH.

The following restatement of Definition 3.6 is useful.

Proposition 1 *Let $\mathbf{E}_P$ be a set of entitlements granted by enterprise P. Let $Obl(P, S, R, I)$ be an obligation. Suppose $I = [t_s, t_e]$. Then $\mathbf{E}_P$ meets $Obl(P, S, R, I)$ iff, for every enterprise $E \in S$ and every $t \in I$, there exists an entitlement $Ent_P(E, R', [t'_s, t'_e]) \in \mathbf{E}_P$ such that $t_s \leq t'_s \leq t \leq t'_e$ and $R \sqsubseteq R'$.* □

3.3 Local policies and obligation fulfillment

Definition 3.7 (Local Policy) For every enterprise $P \in \mathcal{E}$ and every time point t, the set of policies Pol conveys the local policy $\mathbf{LP}^t_P$ of P at time t.

The local policy $\mathbf{LP}^t_P$ is a (finite) set of entitlements $\{Ent_P(E_1, R_1, I_1), \ldots, Ent_P(E_n, R_n, I_n)\}$.

Note that we do not insist that the time point t must be within the validity interval of each entitlement in $\mathbf{LP}^t_P$. A local policy can thus specify future as well as past entitlements. Note also that since $\mathbf{LP}^t_P$ is a set of entitlements, the expression $\mathbf{LP}^t_P$ **meets** $Obl(P, S, R, I)$ is well defined. We can thus speak of a local policy at time t being in compliance with, or meeting, an obligation.

We assume that there is an accounting system at each enterprise P that keeps track, for each resource owned by P, the amount of that resource granted by P to each enterprise E of the coalition over any given time interval. We will also assume that the coalition itself has access to this information, either by trusting P to supply this information or by maintaining its own independent monitoring and accounting system.

Definition 3.8 (Accounting function) Let P and E be enterprises in $\mathcal{E}$, R a specified amount of a resource in $\mathcal{R}$, and t_s and t time points. $Consumed$ and $Rest$ are the accounting functions for resource usage: $Consumed(P, E, R, t_s, t)$ gives the amount of resource R that has been granted by P to E from time t_s up to but not including time t. $Rest(P, E, R, I, t)$ is the amount of P's resource R that was available at time t_s and remains available for the use of E at time t.

$$Rest(P, E, R, t_s, t) = R - Consumed(P, E, R, t_s, t{-}1)$$

In what follows the key role is that of the accounting function $Rest$. We will assume it has the following properties.

$$Rest(P, E, R, t_s, t) = \begin{cases} 0, & \text{if } t < t_s \\ R, & \text{if } t = t_s \end{cases}$$

$$0 \sqsubseteq Rest(P, E, R, t_s, t) \sqsubseteq R$$

$$Rest(P, E, R, t_s, t) \sqsubseteq Rest(P, E, R, t'_s, t) \text{ if } t_s \leq t'_s$$

$$Rest(P, E, R, t_s, t') \sqsubseteq Rest(P, E, R, t_s, t) \text{ if } t \leq t'$$

$$Rest(P, E, R_1, t_s, t) \sqsubseteq Rest(P, E, R_2, t_s, t) \text{ if } R_1 \sqsubseteq R_2$$

Having the accounting function, we are now able to define the conditions for a request to be supported by the coalition policy and the local policy of a resource provider, as follows.

Definition 3.9 (Supported request) Let $Req(u, P, R, t)$ represent a request submitted at time t from user $u \in Users$ to enterprise $P \in \mathcal{E}$ for access to an amount R of a resource owned by P.

$Req(u, P, R, t)$ is *supported* by an obligation $Obl(P, S, R', I)$ if $t \in I$, $u \in users(E)$ for some $E \in S$, and $R \sqsubseteq Rest(P, E, R', t_s, t)$ where $I = [t_s, t_e]$.

$Req(u, P, R, t)$ is a *locally supported* request if there is an entitlement $Ent_P(E, R', I)$ in the local policy $\mathbf{LP}^t_P$ of P such that $t \in I$, $u \in users(E)$, and $R \sqsubseteq Rest(P, E, R', t_s, t)$ where $I = [t_s, t_e]$.

A request $Req(u, P, R, t)$ is *trivial* if $R = 0$.

Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)
0-7695-2141-X/04 $ 20.00 © 2004 IEEE

The following result gives further support for the choice of definitions above.

Proposition 2 *Let $\mathbf{LP}^t_P$ be the local policy of an enterprise P at t. Let $Obl(P, S, R, I)$ be an obligation. Let $I = [t_s, t_e]$ and R be a resource of type $ResType$. $\mathbf{LP}^t_P$ **meets** $Obl(P, S, R, I)$ iff, whatever requests are granted by P for resource of type $ResType$, every non-trivial request supported by $Obl(P, S, R, I)$ is locally supported by $\mathbf{LP}^t_P$.*

Proof. Left-to-right: suppose $Req(u, P, R_x, t_x)$ is supported by $Obl(P, S, R, I)$. Then for some $E \in S$, we have $u \in users(E)$, $t_s \leq t_x \leq t_e$, and $R_x \sqsubseteq Rest(P, E, R, t_s, t_x)$. Since $\mathbf{LP}^t_P$ **meets** $Obl(P, S, R, I)$ there is an entitlement $Ent_P(E, R', [t'_s, t'_e])$ in $\mathbf{LP}^t_P$ such that $R \sqsubseteq R'$ and $t_s \leq t'_s \leq t_x \leq t'_e$.
We need to show $R_x \sqsubseteq Rest(P, E, R', t'_s, t_x)$. We have $R_x \sqsubseteq Rest(P, E, R, t_s, t_x) \sqsubseteq Rest(P, E, R, t'_s, t_x)$ because $t_s \leq t'_s$, and $Rest(P, E, R, t'_s, t_x) \sqsubseteq Rest(P, E, R', t'_s, t_x)$ because $R \sqsubseteq R'$.

For the converse: suppose $t_s \leq t_x \leq t_e$, $E \in S$, and $u \in users(E)$. Suppose further that the non-trivial request $Req(u, P, R, t_x)$ is the only request for resource of type $ResType$ granted by P during the interval I. Then $Rest(P, E, R, t_s, t_x) = R$ and $Req(u, P, R, t_x)$ is supported by $Obl(P, S, R, I)$. This request must also be locally supported, so there is an entitlement $Ent_P(E, R', [t'_s, t'_e])$ in $\mathbf{LP}^t_P$ such that $t'_s \leq t_x \leq t'_e$ and $R \sqsubseteq Rest(P, E, R', t'_s, t_x)$.

By Proposition 1, it just remains to show that $t_s \leq t'_s$. Suppose not: suppose $t'_s < t_s$. Suppose then that there is a granted request $Req(u, P, R', t_s - 1)$. This is outside the interval I and so does not contradict the assumption earlier. $Rest(P, E, R', t'_s, t_s) = 0$, and hence also $Rest(P, E, R', t'_s, t_x) = 0$ (because $t_s \leq t_x$). So $R \sqsubseteq 0$, and $Req(u, P, R, t_x)$ is a trivial request, which contradicts the assumption. □

¿From the above one can see why condition (i) is essential in Definition 3.5.

As informally introduced, publishing a local policy in compliance with an obligation does not imply the actual fulfillment of the obligation. *Violation* of an obligation occurs whenever a *supported* request is not granted.

Definition 3.10 (Obligation violation) An obligation $Obl(P, S, R, I)$ is violated at any time $t' \geq t$ where:

— $t \in I$ and it is not the case that the local policy $\mathbf{LP}^t_P$ **meets** $Obl(P, S, R, I)$; or

— a request $Req(u, P, R_x, t)$ supported by $Obl(P, S, R, I)$ is not granted by P before time $t+\delta$, where δ is some suitably chosen (application specific) value to allow for the time delay between the submission of a request to P and the time at which the request can be acted upon by P.

Once an obligation is violated it remains violated at all future times. Notice that an obligation $Obl(P, S, R, I)$ cannot be violated before the start of time interval I, and that we have chosen to say it becomes violated at any time during I at which the local policy fails to meet the obligation.

An obligation $Obl(P, S, R, I)$ is *fulfilled* if the local policy meets $Obl(P, S, R, I)$ at each time point in the interval I and all supported requests during I are granted. We will say that the obligation becomes fulfilled at the time the interval I ends.

Definition 3.11 (Obligation fulfillment) An obligation $Obl(P, S, R, I)$ is fulfilled at any time $t' \geq t_e$ iff $I = [t_s, t_e]$ and:

— the local policy $\mathbf{LP}^t_P$ **meets** $Obl(P, S, R, I)$ for all $t \in I$; and

— every request $Req(u, P, R_x, t)$ supported by $Obl(P, S, R, I)$ (and hence during I) is granted by P within time $t+\delta$, where δ is the (application specific) time delay between the submission of a request to P and the time at which the request can be acted upon by P.

Once an obligation is fulfilled it remains fulfilled for all future times. Notice that before the time interval I, an obligation $Obl(P, S, R, I)$ is neither violated nor fulfilled, and after time interval I it is either violated or fulfilled (but not both).

We have chosen to say that an obligation $Obl(P, S, R, I)$ becomes fulfilled at the moment the time interval I ends. We could have chosen to say that an obligation could be fulfilled earlier than that, in the case where the resource R is completely used up before I expires. This complicates the presentation unduly however so we will not bother with it here.

Notice that an obligation $Obl(P, S, R, I)$ is fulfilled if no supported request is submitted to P during the time interval I, *as long as* local policy $\mathbf{LP}^t_P$ **meets** $Obl(P, S, R, I)$ for all times $t \in I$.

Of some special interest is the case where a resource providing enterprise P can be assumed to comply with its own published local policy, in the sense that a request submitted to P is granted by P if and only if that request is locally supported by P's local policy. For that special case we have the following.

Proposition 3 *Let P be an enterprise in $\mathcal{E}$. Suppose P grants any (non-trivial) request submitted to it at time t by time $t+\delta$ iff that request is locally supported by P's local policy $\mathbf{LP}^t_P$.*

Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)
0-7695-2141-X/04 $ 20.00 © 2004 IEEE

Then an obligation $Obl(P, S, R, I)$ *is fulfilled iff* $\mathbf{LP}^t_P$ **meets** $Obl(P, S, R, I)$ *for every time* $t \in I$.

Proof. This follows immediately from Definition 3.11 and Proposition 1. □

Notice that a provider P can over-comply with an obligation in the sense that P grants in its local policy an entitlement that exceeds what it is required to provide by the coalition policy. In that case, there can be locally supported requests that are not supported by any obligation: P might then fail to grant a locally supported request without violating any of its obligations under the coalition policy.

3.4 Contracts

Now we consider a special type of *contract*, which will convey the coalition policy. We call it a contract because it is a policy that all members of a coalition have to agree on. A contract specifies the members' obligations towards one another. Contracts are constructed from two basic building blocks: the *obligation sequence* and the *contract block*.

Definition 3.12 (Obligation sequence) An *obligation sequence* $Obl(P, S, R_1, I_1; \ldots; R_k, I_k)$ ($k \geq 1$) is a set of obligations $Obl(P, S, R_1, I_1)$, …, $Obl(P, S, R_k, I_k)$ such that $R_1, \ldots, R_k$ are all instances of the same resource type and $I_1, \ldots, I_k$ is an ordered sequence of contiguous time intervals.

We will also write obligation sequences in the form $Obl(P, S;\ Seq)$ where Seq stands for the sequence of resource-interval pairs $R_1, I_1; \ldots; R_k, I_k$.

$Obl(P, S, R_1, I_1; \ldots; R_k, I_k)$ is *violated* at time t if any of its constituent obligations $Obl(P, S, R_i, I_i)$ is violated at time t.

$Obl(P, S, R_1, I_1; \ldots; R_k, I_k)$ is *fulfilled* at time t if all its constituent obligations $Obl(P, S, R_i, I_i)$ are fulfilled at time t. Clearly an obligation sequence $Obl(P, S, R_1, I_1; \ldots; R_k, I_k)$ is fulfilled when its last constituent obligation $Obl(P, S, R_k, I_k)$ is fulfilled.

A request $Req(u, P, R_x, t)$ is *supported* by $Obl(P, S, R_1, I_1; \ldots; R_k, I_k)$ if it is supported by any of the constituent obligations $Obl(P, S, R_i, I_i)$.

We do not define the 'meets' relation between arbitrary sets of entitlements and an obligation sequence, but we do want to define when the local policy $\mathbf{LP}^t_P$ of enterprise P meets (is in compliance with) an obligation sequence. Informally, $\mathbf{LP}^t_P$ meets $Obl(P, S, R_1, I_1; \ldots; R_k, I_k)$ if the obligation sequence is not violated before t and $\mathbf{LP}^t_P$ meets any constituent obligation $Obl(P, S, R_i, I_i)$ whose time interval I_i has not yet expired at time t. The following is an equivalent and more concise formulation.

Definition 3.13 (Obligation sequence compliance) A local policy $\mathbf{LP}^t_P$ is in compliance with ('meets') an obligation sequence $Obl(P, S, R_1, I_1; \ldots; R_k, I_k)$, denoted

$$\mathbf{LP}^t_P \textbf{ meets } Obl(P, S, R_1, I_1; \ldots; R_k, I_k)$$

if every constituent obligation $Obl(P, S, R_i, I_i)$ is either fulfilled at t or $\mathbf{LP}^t_P$ **meets** $Obl(P, S, R_i, I_i)$.

This means that obligations fulfilled in the past can be ignored by the local policy, which is clearly desirable. It also means that once a constituent obligation is violated the local policy can no longer be modified to meet the obligation sequence. In particular, if the local policy fails to meet an obligation sequence that has started, the local policy can never be modified later to meet the obligation sequence retrospectively.

The following property confirms that the definitions of obligation sequence compliance mirror the corresponding definitions for obligation compliance.

Proposition 4 *Let* $Obl(P, S;\ Seq)$ *be an obligation sequence with* $Seq = R_1, I_1; \ldots; R_k, I_k$. *Let* $I_k = [t_s, t_e]$. *Let* $\mathbf{LP}^t_P$ *be the local policy of enterprise* P *at time* t.

$Obl(P, S;\ Seq)$ *is violated at any time* $t' \geq t$ *iff:*

— $t \in I_1 \cup \cdots \cup I_k$ *and it not the case that* $\mathbf{LP}^t_P$ **meets** $Obl(P, S;\ Seq)$*; or*

— *a request* $Req(u, P, R_x, t)$ *supported by* $Obl(P, S;\ Seq)$ *is not granted by* P *before time* $t{+}\delta$*, where* δ *is the (application specific) time delay to allow a submitted request to be acted upon by* P.

$Obl(P, S;\ Seq)$ *is fulfilled at any time* $t' \geq t_e$ *iff:*

— $\mathbf{LP}^t_P$ **meets** $Obl(P, S;\ Seq)$ *for all* $t \in I_1 \cup \cdots \cup I_k$*; and*

— *every request* $Req(u, P, R_x, t)$ *supported by* $Obl(P, S;\ Seq)$ *is granted by* P *within* $t{+}\delta$. □

The second building block for contracts is the *contract block*.

Definition 3.14 (Contract Block) A contract block CB_P of a coalition policy is an expression of the form:

$$Obl(P, S_1;\ Seq_1) \parallel \ \ldots\ \parallel Obl(P, S_k;\ Seq_k)$$

where each component $Obl(P, S_i;\ Seq_i)$ ($i \in 1..k$) is an obligation sequence. P is the *bearer* of CB_P.

The contract block CB_P is *fulfilled* at time t if any of CB_P's components $Obl(P, S_i;\ Seq_i)$ is fulfilled at time t.

The contract block CB_P is *violated* at time t if all of CB_P's components $Obl(P, S_i;\ Seq_i)$ are violated at time t.

A request $Req(u, P, R_x, t)$ is *supported* by the contract block CB_P if it is supported by any of CB_P's components $Obl(P, S_i;\ Seq_i)$.

Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)
0-7695-2141-X/04 $ 20.00 © 2004 IEEE

A local policy $\mathbf{LP}_P^t$ is in compliance with ('meets') the contract block CB_P, denoted $\mathbf{LP}_P^t$ **meets** CB_P, if $\mathbf{LP}_P^t$ **meets** $Obl(P, S_i;\ Seq_i)$ for any component $Obl(P, S_i;\ Seq_i)$ of CB_P.

The effect of the contract block is to give the bearer P a complete *free choice* about which of the component obligation sequences it wishes to fulfill. Although in principle these obligation sequences may differ in terms of the amount of a resource, the type of the resource, the set of enterprises to whom the resource is to be granted, and/or the time intervals over which the obligations hold, in practice only certain special forms are useful.

In particular, the contract block can be used to capture the common pattern of contractual obligations in which violation of one obligation can result in new obligations that then must be fulfilled. Often these new obligations will impose more stringent requirements, providing both an incentive to fulfil the original, easier obligation and a means of compensating for any violations that occur. By arranging the components of a contract block appropriately, violation of one component leaves only the choice of more stringent obligations if the contract block is to be fulfilled.

Example 3
Suppose SICS agrees to provide KTH with 100Gb of disk storage over the time interval $[1, 10]$ *with 50Gb to be provided over the interval* $[1, 5]$ *and a further 50Gb to be provided over* $[6, 10]$. *(Here we employ integers for time points rather than the day-time notation simply to reduce clutter.) If, however, SICS fails (for whatever reason) to provide* KTH *with 50Gb over the interval* $[1, 5]$*, but manages to provide at least 30Gb over that period, then it must provide* KTH *with 90Gb over the interval* $[6, 10]$ *to compensate.*

This coalition policy can be represented by a contract block with the following two components:

1. $Obl(SICS, \{KTH\};\ 50Gb, [1, 5];\ 50Gb, [6, 10])$
2. $Obl(SICS, \{KTH\};\ 30Gb, [1, 5];\ 90Gb, [6, 10])$

Suppose first that SICS violates its obligation to provide at least 30Gb of storage at some time t *during* $[1, 5]$*, either by failing to include an appropriate entitlement in its local policy at time* t *or by denying a supported request at time* t *from a user at* KTH*. In that case, the obligations in component 1 are also violated at time* t*, and the whole contract block is violated.*

Suppose that SICS fulfills its obligation to provide 50Gb of storage over $[1, 5]$*. It therefore also fulfills its obligations for* $[1, 5]$ *in component 2. It can fulfill the contract block as a whole by providing 50Gb over* $[6, 10]$*. It can also fulfill it by providing more than 50Gb over* $[6, 10]$ *but that would be surplus to requirements.*

Suppose now that SICS violates its obligation to provide at least 50Gb of storage (component 1) at some time t *during* $[1, 5]$ *though it does not violate the obligation to provide 30Gb (component 2), for example, by including an entitlement for* KTH *to 40Gb in its local policy at time* t*. Since the first obligation of component 1 is now violated, the only way that SICS can fulfill the contract block as a whole is by fulfilling component 2. Effectively SICS now has an obligation to provide 90Gb of storage over* $[6, 10]$*, since that is the only available means of fulfilling the contract block. Notice that once component 1 is violated it remains violated. Having violated component 1 with its local policy at time* t*, SICS cannot undo the violation, for example, by altering its local policy later.*

It should be clear that the example can be extended with further layers in similar fashion. Suppose that if SICS fails to provide 30Gb over the period $[1, 5]$ *but manages to provide 10Gb, then it must provide 100Gb over the interval* $[6, 10]$*. This is represented by adding a third component to the contract block:*

3. $Obl(SICS, \{KTH\};\ 10Gb, [1, 5];\ 100Gb, [6, 10])$

The reader may care to check how new obligations are effectively triggered by various kinds of violations in this extended example.

It is worth observing that the obligations in a contract block such as the one in the previous example are similar but not exactly the same as obligations of the 'contrary to duty' type [9]. In a 'contrary to duty' structure there is a primary obligation, and a secondary obligation which comes into force if the primary obligation is violated. To that extent the contract block is similar. The difference is that in a contract block there is no 'primary' obligation that gets special status: as long as at least one of the components of the contract block is fulfilled, there is no violation of the contract block as a whole.

We are now ready to define a contract. A contract is simply a set of contract blocks, with one important additional proviso. With the definitions constructed above, we must ensure that there are never two (or more) separate and independent obligations requiring one enterprise P to provide several instances of the same resource to the same enterprise E at the same time. In such circumstances (to be stated more precisely below), P can fulfill several obligations at once with just one single entitlement. Suppose, for example, there are two separate obligations $Obl(P, \{E\}, R_1, I)$ and $Obl(P, \{E\}, R_2, I)$. One might assume that E is thereby granted a total allowance of $R_1 + R_2$ over interval I. The treatment of obligation compliance developed above, however, gives a different effect: that of a single obligation $Obl(P, \{E\}, R, I)$ where R is the larger of R_1 and R_2. When formulating contracts care must be taken to ensure that this is the intended effect. The alternative would be to *name* all obligations and associate entitlements explicitly with named obligations. Fortunately, the

Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)
0-7695-2141-X/04 $ 20.00 © 2004 IEEE

conditions under which problems can arise, and which need to be eliminated, are comparatively obscure.

Definition 3.15 (Contract) A contract is a (finite) set of contract blocks $\{CB_1, \ldots, CB_n\}$ $(n \geq 1)$ such that, for any pair of obligations $Obl(P, S, R, [t_s, t_e])$ and $Obl(P, S', R', [t'_s, t'_e])$ in different contract blocks with R and R' both instances of the same resource type and $S \cap S' \neq \emptyset$, we have $t_s \neq t'_s$.

Given any set of contract blocks, the required conditions for a contract are easily and mechanically checked.

Definition 3.16 (Contract compliance) A contract *Contract* is violated at time t if any contract block in *Contract* is violated at time t .

Contract is fulfilled at time t if every contract block in *Contract* is fulfilled at time t .

A local policy $\mathbf{LP}^t_P$ complies with *Contract* iff $\mathbf{LP}^t_P$ **meets** CB for every contract block CB in *Contract* for which P is the bearer of CB.

4 Future work

The work presented in this paper is ongoing research, originally started with [5, 2], for designing and implementing a formal and sound framework for contractually regulated coalitions.

We are currently working on integrating the idea of *witness* and associated protocols, introduced in [2], as a trusted third party for monitoring the resource sharing between coalition members. We are also extending the framework with the delegation mechanism and the privilege calculus, given in [6], for decentralised management of privileges in coalitions. We will extend the privilege calculus for support of entitlements and obligations as presented in this paper.

For future work we are following three complementary directions. The first direction is to extend the policy language. The combination of obligation sequence and contract block presented here is still rather restrictive; we have not yet explored how restrictive it will prove to be in practice. We also plan to provide means of specifying polices about policies, that is to say, policies specifying what other (coalition) policies can be created and under what circumstances. In this paper, we have allowed local policies to vary over time but the coalition policy *Contract* is fixed. Finally, the focus in this paper is on obligations and entitlements to a specified amount of consumable resource over a specified period of time. We also need to speak of obligations and entitlements to non-consumable resources where there is no sense of measure, such as access to a file, which is either granted or not. There is no difficulty in extending the definitions to support non-consumable resources as well; indeed, the corresponding definitions are very much simpler than those for consumable resources. We have concentrated on consumable resources in this paper precisely because they are more involved.

The second direction is to investigate the relationships between our framework and various methods for optimization and dynamic scheduling of resource allocations. We believe that there are interdependencies between coalition and local policies and the way planning of resource usage can be done. This is of course related to the idea that an enterprise can be a member of several coalitions offering the same resources. Hence, we need mechanisms for deciding which obligations to fulfill in order to minimize sanctions resulting from non-compliance whilst at the same time optimizing resource usage.

Thirdly, we plan to develop mechanisms and architecture to monitor contractual performance and enforce contractual agreements, enabling secure detection of contractual violations and triggering recovery mechanisms that apply to those violations.

5 Related work

Bettini and colleagues [3] formalize a rule-based policy framework that includes 'provisions' and 'obligations', and investigates a reasoning mechanism within this framework. They distinguish between actions (*provisions*) that have to be performed before a decision is taken and actions (*obligations*) that will be taken after the decision. These actions are represented as two disjoint sets of predicates. The system implementing the policy rules must deduce what actions (if any) may be performed to gain access, and what promises (if any) must be made after gaining the access. The system also monitors the progress of obligation fulfillment and, in case of failure, takes compensatory actions. Provisions are structured and have an associated weight, allowing the selection of the weakest obligation thus considering semantic relations between them. The main distinction between this work and ours is that Bettini's focuses on enhancement of policy rules with provisions and obligations to be accounted by systems implementing these rules, whereas we mainly focus on modeling policies for distributed communities sharing resources. Furthermore, Bettini's system is not integrated with any temporal reasoning technique, though the authors have indicated that this is a topic to be explored.

Xuhui Ao and Naftaly Minsky [1] have developed a fully implemented regulatory mechanism for coalitions, based on a very general view of the governance of coalitions. Similarly to our work, the coalition is governed by a global policy P_C, and each coalition member E_i is governed by a local policy, which must conform to P_c. Their definition of both coalition and local policy is provided using the LGI language [8], a general message-exchange mechanism that allows an open group of distributed agents to engage in a

Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)
0-7695-2141-X/04 $ 20.00 © 2004 IEEE

mode of interaction governed by a policy. Our concept of coalition policy is different. We see it as a contract, and provide formal definitions of coalition and local policy and relationships between them. Furthermore in [1] local policies are obtained using a top-down approach, as refinements of coalition policies, whereas we consider local policy as a plan for enterprises to allocate resources under their own control while complying with coalition policy as expressed in the contract. Moreover in [1] no punitive mechanism is considered, since in that framework every obligation is necessarily fulfilled unless it is repealed.

In [7] there is proposed a contract monitoring system intended to provide automated checking of business to business contracts. It introduces a novel modelling approach to obligations, unifying the treatment of both permissions and obligations by reifying both and describing permit and burden passing in a way analogous to the established treatment of capabilities.

[4] deals with the problem of negotiating access to resources that exist within different administrative domains, presenting protocols for managing the process of negotiating access to resources in a distributed system. The approach taken is to define a general resource management model within which reservation, acquisition, task submission, and binding of tasks to resources can be expressed in a uniform fashion.

6 Conclusion

We have presented a framework for specifying and reasoning about policies for sharing resources in coalitions, focussing on the common case where coalition members agree to make available some total amount of consumable resource over a specified period of time. The simpler case of access to non-consumable resources, which can either be granted or not, can be treated in similar (simpler) fashion.

The framework provides a policy language with two basic elements: 'obligations', which are used to express the coalition policy (contract), and 'entitlements' which are used to express the local policies of the coalition members. Local policies may vary from time to time; for simplicity in this paper we have assumed that the coalition policy is fixed.

Each member of a coalition must publish its local policy so that other members can plan how to make use of the resources provided. A local policy must comply with, or 'meet', the obligations specified by the coalition policy. A member enterprise violates an obligation either by failing to publish a local policy that meets the obligation or by denying the resource in contravention of its own local policy. An accounting mechanism keeps track of which resources are granted over time. We have investigated the conditions under which violation of an obligation in the coalition policy (contract) coincides with contravention of a local policy.

We have presented two basic building blocks for constructing coalition policies (contracts) from 'obligations'. An 'obligation sequence' collects a set of related obligations over an ordered sequence of contiguous time intervals. A 'contract block' specifies a set of alternative obligations with a free choice about which of them must be fulfilled. By combining these structures in various ways, one can represent common patterns of contractual provisions, such as the common case where violation of one obligation results in new, usually more stringent, obligations intended to compensate for the violation.

Our current work includes development of the mechanisms and infrastructure required to monitor contractual performance and enforce contractual performance.

References

[1] X. Ao and N. H. Minsky. Flexible regulations of distributed coalitions. In *Proceedings of the 8th European Symposium on Research in Computer Security (ESORICS)*, volume 2808 of *LNCS*, Gjøvik, Norway, October 2003. Springer Verlag.

[2] E. Bertino, E. Ferrari, and A. Squicciarini. A decentralized framework for controlled sharing of resources in virtual communities. In *Proceedings of the 17th IFIP Conference On Database and Applications Security, Estes Park, CO*, August 2003.

[3] C. Bettini, S.Jajodia, X. S. Wang, and D. Wijesekera. Provisions and obligations in policy management and security applications. In *Proceedings of the 28th International Conference on Very Large Data Bases (VLDB)*, Hong Kong, China, August 2002.

[4] K. Czajkowski, I. Foster, C. Kesselman, V. Sander, and S. Tuecke. Snap: A protocol for negotiating service level agreements and coordinating resource management in distributed systems. In *Proceedings of the 8th Workshop on Job Scheduling Strategies for Parallel Processing*, volume 2537 of *LNCS*, pages 153–183, Edinburgh, Scotland, July 2002.

[5] B. S. Firozabadi and M. J. Sergot. Contractual access control. In *Proceedings of Security Protocols, the 10th International Workshop*, volume 2845 of *LNCS*, pages 96–102, Cambridge,UK, April 2002. Springer Verlag.

[6] B. S. Firozabadi, M. J. Sergot, and O. Bandmann. Using Authority Certificates to Create Management Structures. In *Proceeding of Security Protocols, the 9th International Workshop*, volume 2467 of *LNCS*, pages 134–145, Cambridge, UK, April 2001. Springer Verlag.

[7] P. Linington and S. Neal. Using policies in the checking of business to business contracts. In *Proceedings of the 4th IEEE Workshop on Policies for Distributed Systems and Networks*, pages 207–218, Como, Italy, June 2003.

[8] N. H. Minsky and V. Ungureanu. Law-governed interaction: a coordination and control mechanism for heterogeneous distributed systems. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 9(3):273–305, july 2002.

[9] H. Prakken and M. Sergot. Contrary-to-duty obligations. *Studia Logica*, 57(1):91–115, 1996.

Proceedings of the Fifth IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'04)
0-7695-2141-X/04 $ 20.00 © 2004 IEEE