Chapter 3
Management of Virtual Organizations

Nicola Capuano, Angelo Gaeta, Matteo Gaeta, Francesco Orciuoli, David Brossard, and Alex Gusmini

Abstract In the Virtual Organization (VO) Management area the main challenge has been to develop policies and models for governance and lifecycle management of a business-to-business (B2B) collaboration. This work included research and development in the areas of federated identity management and semantics in addition to VO, business registries and B2B collaboration managements. The main results produced in the VO Management area include capabilities, patterns and software solutions to simplify governance and lifecycle management of B2B collaborations (VOs), and to manage applications distributed over several federated network hosts (e.g. Cloud Computing platforms).

3.1 Introduction

The activities of the VO Management area have led to the identification of Technical Requirements, Common Capabilities, Design Patterns and Software components to address the issues of governance and lifecycle management of a VO, including aspects of security and semantics in a VO.

The main challenges addressed by this area are the creation and management of a secure federated business environment among autonomous administrative domains, the separation of concerns between provision and management of application services and operational management of the VO infrastructure (e.g. separating the coordination of application execution from Resource monitoring), and the automatic discovery of available resources or services which meet a given set of functional requirements inside a VO or among different VOs.

The three key capabilities developed in this area are: (i) VO Set-up [14], this is a capability that facilitates business partner identification, and the creation and life-cycle management of a circle of trust among business partners. A competitive differentiator is that trust is aligned to consumer/provider relationships; hence

N. Capuano (✉)
Centro di Ricerca in Matematica Pura ed Applicata (CRMPA) c/o DIIMA, via Ponte Don Melillo, 84084 Fisciano (SA), Italy
e-mail: capuano@crmpa.unisa.it

T. Dimitrakos et al. (eds.), *Service Oriented Infrastructures and Cloud Service Platforms for the Enterprise*,
DOI 10.1007/978-3-642-04086-3_3,

supporting the evolution of circle of trust to a trust network that reflects supply relationships; (ii) Application Virtualization [15], this is a composite capability that enables managing the deployment, distribution and configuration of capabilities and resources required for offering a service that is distributed over multiple hosts/cloud platforms. It offers a unifying layer for managing identity, secure service integration, SLA fulfilment and performance monitoring across multiple platforms; (iii) Automated Resource Discovery [16], a capability that improves the process of resource and service discovery in a VO by adopting semantic models and technologies.

The former two of these capabilities have been validated in a project case study demonstrating a network-centric distributed platform for scalable, collaborative online gaming [2]. The concept of a Virtual Hosting Environment that underpins this Business Experiment is an innovation that is transferable across vertical market sectors and appears to offer a generic solution for distributing services and resources in multiple Cloud Computing platforms depending on SLA requirements and offering value add by strengthening security, identity management, performance monitoring and accounting. The latter of these capabilities (automated resource discovery) has been validated in another case study focusing or sharing anti-fraud data fro roaming users within an international Group of mobile operators [5].

The rest of the chapter is devoted to introduce the main challenges of the Virtual Organization Management area, a selection of the most relevant common technical requirements, a set of common capabilities, design patterns and software components, a sample scenario showing how components interact together and how they can be collectively adopted to address a common business issue, and lastly the lessons learnt during our analysis of the case studies and some good practices identified.

The chapter is concluded with some considerations on the business adoption of the developed components.

3.2 The Main Challenges

Within the VO thematic area we have firstly tried to fix terminology and concepts. During the analysis we have taken common VO concepts from ECOLEAD [7] and TrustCoM [25] into account, in recognition of the fact that substantial basic research has already been done in the area and also that basic research on VO models and foundations is outside of the scope of the project.

Nonetheless, we think it is worth mentioning the approach for VO creation that we have selected according to the analysis of the business experiments.[1] In [6], some approaches investigated in R&D to create a VO are presented and described.

[1] Many case studies analyzed rely upon the service concept and WS-* family of specifications.

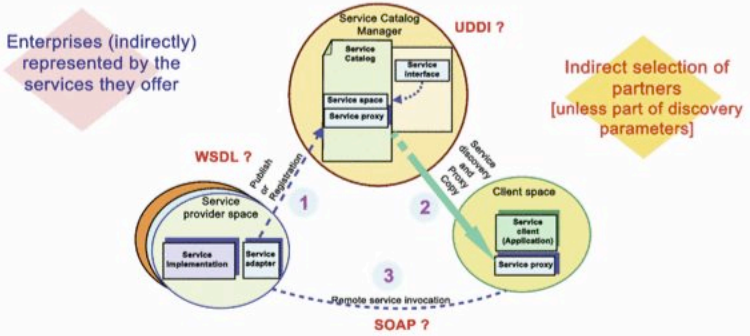

Fig. 3.1 Service Federation approach for VO

Among those, for our purpose, we are close to the so-called *Service Market based* or *Service federation* approach.

According to this approach the potential collaborative behaviour of each company is "materialized" by a set of services, and members of the VO are considered as Service Providers. The approach assumes the existence of one entity that keeps a catalogue of services where service provider companies publish their service offerings. To interact with each other, companies use standard protocols and technologies for service description, communication and data formats. Indeed, this is the approach in which the SOA, and particularly Grid based SOA like the Open Grid Services Architecture (OGSA) [9], represents a major trend in developing systems based on services.

Figure 3.1, from [6], depicts the situation.

With respect to the VO creation and management, the main problems addressed by the VOM area relate to mechanisms for federating services/resources belonging to different Service/Host Providers and facilitating the access and management to the federated services/hosts, on mechanisms allowing application and services deployment, and on mechanisms to manage the policies inside the federated group taking into account the policies of each Service/Host Provider that owns the service/resource.

Security for VO aims to improve the above mentioned mechanisms by exploiting models, standards, and specifications for secure federation. It will also take into account policies for access control and authorization mechanisms. These studies have been done in cooperation with the Security Area of the BEinGRID project.

Semantics for VO, finally, is focused on semantic annotation of services/resources and automatic services/resources discovery.

3.3 Technical Requirements

In the following sub-sections, the most relevant common technical requirements elicited from the case studies analysed are presented and described. For each one of the requirements, technical novelty[2] and business impact[3] are presented.

Before describing the selected requirements, it is worth summarising the lessons learnt during the elicitation activity. What clearly appears is that business communities are more interested in simplifying the management of heterogeneous resources in a federated business environment than in the dynamicity of the life-cycle of VOs. In particular, the business experiments analysed present common requirements related to accessing and managing, in a simple and secure way, heterogeneous distributed resources shared among the organisations participating in a collaboration. The experiments also present issues relating to resource discovery and application/service deployment.

Another key problem that emerges from the analysis is the dematerialisation of the ICT infrastructure underpinning VOs: application and ICT resource providers want to reduce or outsource the overhead of managing the distributed Service Oriented Infrastructure that underpins their Business-to-Business collaborations. It is worth mentioning that we identified business experiments—within the BEinGRID project—that present approaches with high innovation potential to address these issues.[4]

3.3.1 Secure Federation

This requirement is about the creation of a secure federated business environment among autonomous administrative domains.

The main challenges which secure federations encounter relate to the trust establishment and secure credentials distribution across multiple domains. In particular, the challenges identified relate to common federated identity issuing mechanisms, common cross organisational trust establishment mechanisms (which need to be independent of the partner-specific authentication/authorisation inside the trust realm), recognisable set of credentials, configurable solutions to support federation-related interactions.

The challenges behind this requirement are currently not addressed by traditional solutions, although research effort has been undertaken in R&D projects like TrustCoM [25], NextGrid [19], and BREIN [1]. The business value of this requirement is very relevant for business scenarios in which actors need to establish Business-to-Business trusted relationships.

---

[2]Technical novelty indicates how challenging and otherwise unavailable a solution addressing the requirement can be.

[3]Business impact indicates if there is a concrete business case behind the requirement.

[4]It is the case, for example, of the implementation of a Virtual Hosting Environment for on-line gaming application provision. See http://www.beingrid.eu/be9.html.

3.3.2 Separation of Infrastructure Management Capabilities from Application Specific Ones

This requirement is related to the separation of concerns between the provision and management of application services (e.g. coordinating application execution, SLA monitoring) and the management of the VO infrastructure (e.g. Resource monitoring, Accounting modules, Service registry).

This requirement covers a problem faced by Application Service Providers (ASPs) that are currently responsible for managing the infrastructure and the federated hosting/execution environments. In the context of the VO life-cycle, this requirement essentially covers the set-up of the infrastructure of a VO. During the VO operational phase, common capabilities that address this requirement allow VO members to focus on managing the application level while outsourcing the administration of the underlying Service Oriented Infrastructure.

The challenges behind this requirement need an innovative solution that allows the exposure of applications in a simple, secure and manageable way without being involved in the management of the enabling infrastructure.

In terms of business impact, the requirement allows for mitigation of risks, increased flexibility (separation of responsibilities), and potential cost reduction or strategic advantage as it enables the outsourcing of infrastructure, and also handles overflow capacity and disaster recovery.

3.3.3 Automatic Resource and Service Discovery

This requirement addresses the need to discover inside a VO or among different VOs available resources/services which meet a given set of functional and/or non-functional requirements.

The challenge is to identify those services and service providers which can meet the requirements and which can reliably provide the required service, and subsequently to make a selection based upon considerations such as performance, reliability, trust, cost and quality of service.

3.4 Common Capabilities

In this section, the most relevant common capabilities are presented and described. The capabilities address recurring problems of the case studies analysed. For each capability, we describe the problem addressed, present a high-level design pattern and a sample implementation of the capability.

It is worth mentioning that during our analysis, the goal has been to abstract as much as possible the specific solutions implemented in the case studies analyzed in order to identify common capabilities that can also be reused in other contexts.

Table 3.1 summarizes the relationship between the selected technical requirements, the selected common capabilities and the software components.

Table 3.1 Selected technical requirements, capabilities and components

Common technical requirements	Common capabilities	Software components
Secure federation	VO Set-up	VO Set-up
Separation of infrastructure management capabilities from application specific ones	Creation of instances in service oriented distributed infrastructures Application Virtualization	Application Virtualization
Automatic Resource/ Service Discovery	Automatic Resource/ Service Discovery	Automatic Resource Discovery

3.4.1 VO Set Up

This capability addresses some recurring problems during the VO lifecycle, mainly in the identification and formation phases, such as partner identification, creation and management of a circle of trust among partners.

This capability is useful in typical cross-enterprise collaborative scenarios where participants (users, services, resources) have to be identified. A demand for including new participants can appear during the collaboration lifetime, and the existing participants may be dropped.

At the same time, the security of the collaboration needs to be maintained: members of a collaboration must be able to identify one another, identify messages as coming from other members of the federation, and identify the truth of claims made by other parties in the federation.

3.4.1.1 High Level Design

Figure 3.2 presents a pattern to solve the problem addressed by this capability.

The VO Set Up acts as a façade and interacts with two components: a Registry component allowing to identify potential partners of a VO and with a Federation component that is responsible for starting the creation of a circle of trust among participants.

3.4.1.2 Sample Implementation

A sample implementation of this capability is provided by the BEinGRID VO Set Up component [14].

The VO Set Up is a web service providing functionalities to support the VO lifecycle phases, and in particular the Identification and Formation phases where members of the VO have to be identified and a circle of trust among them has to

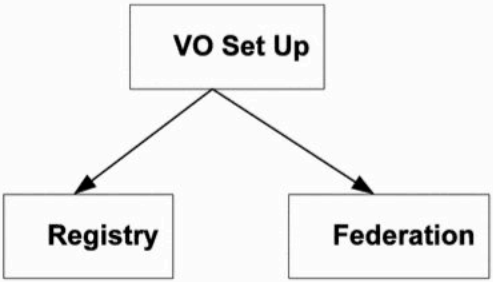

Fig. 3.2 VO Set Up

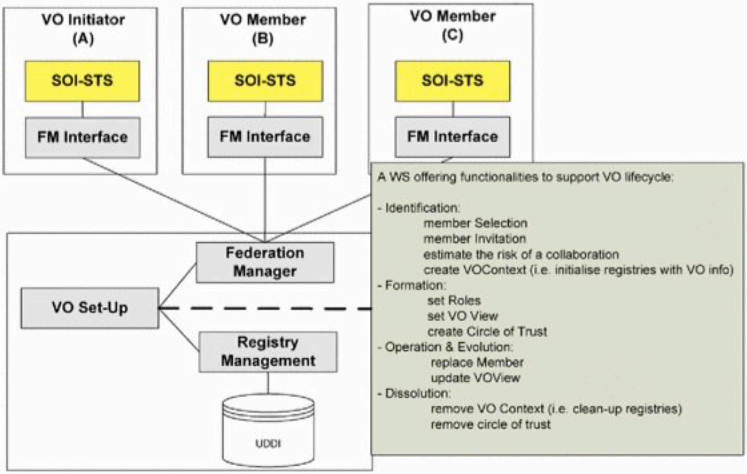

Fig. 3.3 High level architecture of the VO Set Up component

be created. The component allows the management of VO-related registries and the management of secure federation lifecycle.

A high level architecture of this component is shown in the next picture where, for completeness, are also shown the functionalities of the component divided per VO lifecycle phases. The picture shows also a possible deployment of the building blocks of the component. It is worth noting that to allow the secure federation lifecycle management, the VO Set Up interacts (via the federation manager building block) with the Security Token Service component [17] developed in BEinGRID. In the picture, the FM (Fed Manager) interface is a programmatic interface allowing to decouple the VO Set Up component from the specific SOI-STS implementation.

Each partner of a VO needs a Security Token Service (SOI-STS, which acts as an identity broker as well as a circle-of-trust enabler) and, on one partner site, the VO Set Up and its building blocks should be deployed.

This component combines VO registries and federation management in a single solution.

VO registries are built on top of UDDI standard [20] and allow the publication, discovery, and update of VO members and services. The secure federation model implemented is borrowed from the TrustCoM results [25]. The model is credential and policy-based and allows for establishment of asymmetric and binary trust relationships. The TrustCoM model has been improved and implemented, and integration with UDDI has been achieved to enable an enhanced identification phase.

The VO Set Up implements also a basic model to evaluate the risk associated with a collaboration. In its current implementation, the risk is estimated by evaluating a weighted mean of "reliability" values associated to each provider in a collaboration. The "reliability" is a metadata (implemented using the tModel structure of the UDDI standard) associated to each provider in a collaboration. The value associated to the reliability is given via feedbacks by other entities collaborating with the Provider in the past.

It is worth mentioning that the benefits of adopting the VO Set Up component is that it acts 'as a glue' among different capabilities that are required in the VO identification & formation phase. Without the adoption of this component, providers willing to trigger or participate in a VO would need to deploy and manage different components such as, for example, business registry, a service instance registry, and an Identity Management solution.

The VO Set Up component has been evaluated in the context of a concrete case study: the Virtual Hosting Environment for Distributed Online Gaming [2]. The validation of the component inside a concrete experiment has allowed us to prove the usefulness of its functionalities for VO identification and formation, the usefulness of UDDI (and the tModels defined to customise UDDI information model) as registry for VO members and VO service instances, and the process to create the circle of trust. Moreover, the experiment allowed to verify that VO registries and the federation manager could be centrally managed and configured in a coherent way via the VO Set Up component. More information on the VO Set Up evaluation can be found in [10].

3.4.2 Creation of Instances in Service Oriented Distributed Infrastructures

This capability addresses the recurring problems of service identification & creation for running and managing applications on a distributed set of resources or endpoints belonging to a Service-Oriented Infrastructure (SOI).

A common case foresees an Application Service Provider (ASP) that has to provide application capabilities to a client on the basis of an agreed contract. The ASP

is a member of a VO and is aware that it can provide the application capabilities but it does not know where the application capabilities are actually deployed. It is also unaware of the status of the heterogeneous resources of the VO. For this purpose, the ASP delegates the selection of suitable hosting environments and the instantiation of the concrete services offering the required application capabilities.

3.4.2.1 High Level Design

The next picture presents a pattern to solve this problem. The client (e.g. the Application Service Provider) asks the Matchmaker for the selection of the most suitable environment. The Matchmaker performs matchmaking on the basis of application requirements and profiles of the endpoints and returns a list of suitable hosting environments. Next, the client asks for the creation of manageable service instances by invoking a high level factory, namely the Virtual Hosting Environment (VHE) factory.

The VHE factory delegates the creation of a manageable service instance to a concrete factory which, in turn, creates instances of the Management Services and of the Application Service. The endpoint references of the created instances are returned to the client.

This pattern allows the on-the-fly discovery of endpoints on which application requirements can be guaranteed and creates service instances on those endpoints. It also allows the abstraction from the specific application creation details of a particular environment. Another advantage is to decouple the application-specific logic from the management ones. To add a new family of services, the VHE factory interface has to be modified.

This pattern basically combines to the GoF Façade, Abstract Factory and Observer [11] design patterns. This pattern is also similar to the Broker Service Pattern presented in [21].

It is appropriate to apply this pattern when:

- The requestor does not know what are the suitable hosting environments where it is possible to create the instances;
- The environment within which the application operates is very dynamic, and resources are likely to register and de-register often;
- There is the need to be independent from the specific details of the creation of the family of services' instances.

3.4.2.2 Sample Implementation

Following the pattern aforementioned, the capability described in this section has been implemented in the Application Virtualization component [15].

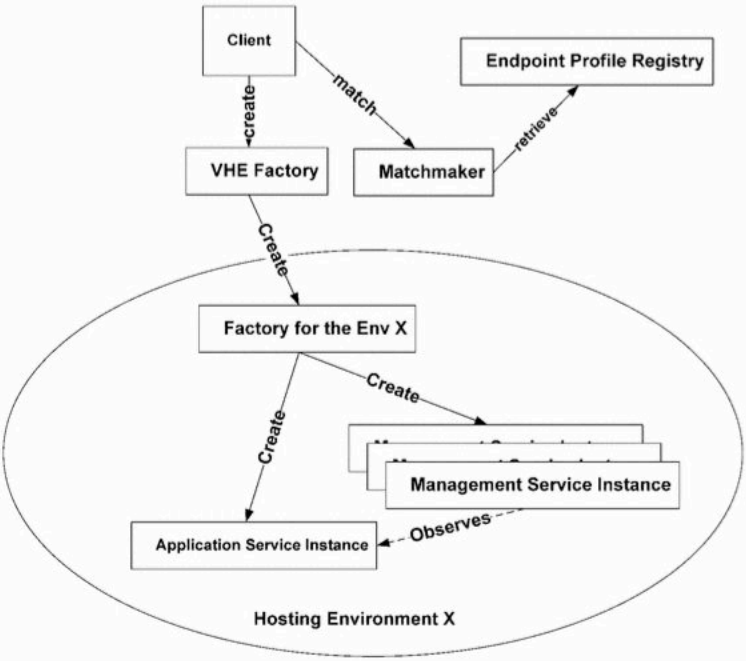

Fig. 3.4 Creation of instances in service oriented distributed infrastructures

3.4.3 Application Virtualization

This capability addresses the problem of integration and exposure of application services through a single access point (e.g. a Gateway) that is configured to manage the execution of the exposed capabilities and forward requests to them.

The capability allows an easy management of the application, taking into account policies and contracts, reducing the overhead of ASP/SP in managing the enabling infrastructure.

A common case of adoption of this capability relates to the need of exposing application capabilities (for direct usage or for composition) as network-hosted services in order to avoid direct and unmanaged access of VO resources by VO members.

3.4.3.1 High Level Design

Figure 3.5 graphically shows a pattern to address this problem.

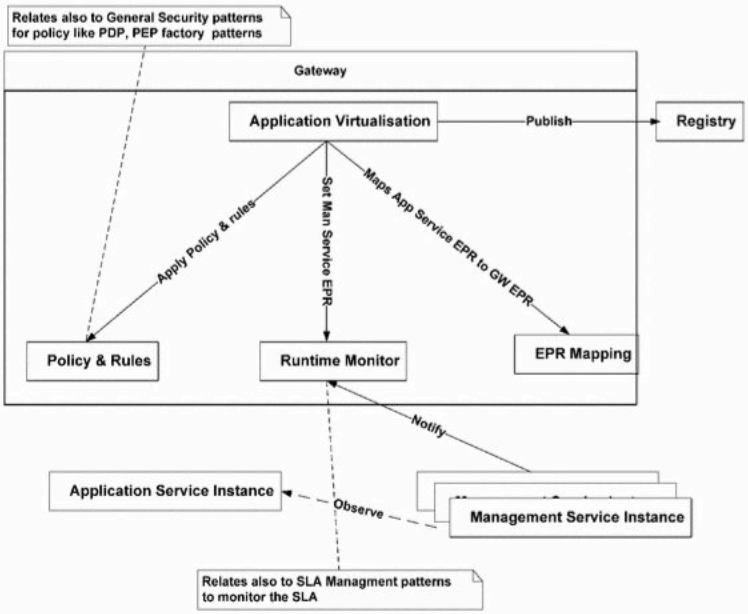

Fig. 3.5 Application Virtualization

In the picture there are notes indicating that some components relate to other BEinGRID technical areas. This means that, for instance, the Policy & Rules component can be designed and implemented according to the patterns proposed by the General Security area and the Runtime Monitoring component can be designed and implemented according to the SLA evaluation pattern proposed by the Service Level Agreement area. Interested readers can refer to the respective chapters of this book.

The Application Virtualization component follows the GoF Façade pattern and is responsible for invoking the other classes of the system in order to execute the virtualization process that consists of the following steps:

- Map the real endpoint reference of the application service instance into a virtual endpoint reference;
- Set the policies that govern who can access the application service instance and under what conditions;
- Provide the endpoint reference of the management services to a run-time monitor that is in charge of monitoring the execution (e.g. monitoring the SLA);
- Publish the virtual endpoint reference in a registry allowing other organisations/clients to discover the application service instance.

The Application Virtualization, the Runtime Monitor and the Management Service can iterate the GoF Observer pattern. Management Service Instances notify the

Runtime monitor with the updates of some parameters and the Runtime Monitor can notify violation to the Application Virtualization.

If the Application Virtualization component is also the Gateway, when a request for accessing a service arrives, the Application Virtualization can operate according the GoF Chain of Responsibility pattern and pass the request along a chain of handlers.

It is appropriate to apply this pattern when there is the need to:

- Decouple service access logic from the rest of the application
- Hide the complexities of accessing a service
- Have a single point providing common management
- Avoid direct access to resources.

3.4.3.2 Sample Implementation

The Application Virtualization component is a web service providing functionalities to create business capabilities required for the operational phase of the VO and configure infrastructural services for secure message exchange within the VO and monitoring & evaluation of the SLAs.

A high-level architecture of this component is shown in Fig. 3.6. It is possible to observe that the Policy & Rules component of Fig. 3.6 has been implemented via the triplet Secure Messaging Gateway (SOI-SMG), Authorization Service (SOI-AuthZ-PDP) and Security Token Service (SOI-STS) components of the General Security area while the Runtime Monitor of Fig. 3.6 has been implemented via the SLA monitoring and evaluation component of the SLA area. The Automatic Resource/service discovery component, instead, is presented in the following section of this chapter.

The component can be used in the VO Creation and Dissolution phases. In terms of functionalities, in fact, it allows to execute two processes.

The first one, namely the Virtualization process, consist of the following steps:

(i) Creation of services instances (business and management) on the selected hosts,
(ii) Mapping of real endpoint reference to virtual one,

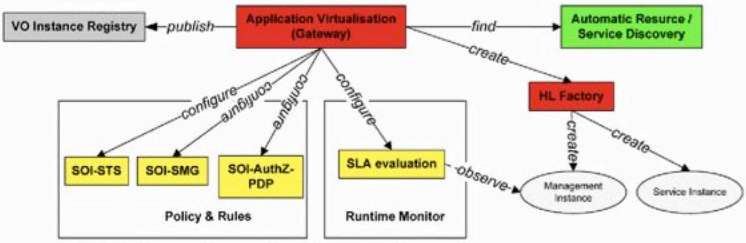

Fig. 3.6 High level architecture of the Application Virtualization component

(iii) Configuration of management services (SOI-SMG, SOI-AuthZ-PDP, SOI-STS and SLA evaluator), and
(iv) Publication of the virtual endpoint reference of the service instance into a VO Service Instance Registry. This process is executed at the end of the VO creation phase when partners that have promised to offer a service or an application in a VO need to configure their environment in order to allow secure and manageable access to that particular service instance.

The second process, namely the Graceful Shutdown, cleans up and destroys the configuration of management services, and destroys business service instances. The process consists of the following steps:

(i) Remove the service instance entries from VO Service Instance Registry,
(ii) Clean up the management services,
(iii) Clean up the Gateway (e.g. remove its internal mapping between virtual and real endpoint references), and
(iv) Destroy the business & management service instances.

3.4.4 Automatic Resource/Service Discovery

This capability addresses the recurring problem of resource and service discovery inside a VO based on a given set of functional requirements the resources need to fulfill.

The capability improves the traditional process of resource and service discovery with the adoption of semantic models and technologies.

The problem is common in several business experiments analysed for which, for instance, the scheduling and deployment of applications depend upon a number of different kinds of information, such as current workload, current application deployment, current network topology and so on.

3.4.4.1 High Level Design

Figure 3.7 shows a pattern to address this problem.

The basic idea behind the above presented design is to provide an interface to different information resources such as workload monitors, network configuration, and current application deployment information.

The participants are:

- *Resource Discovery:* interface to the subsystem. It delegates client requests to appropriate subsystem objects.
- *Run Time monitor:* A proxy for components such as Ganglia [12] and Hawkeye [18] that collect workload information from endpoints.
- *Deployment:* an interface to a database storing information about current applications deployed on endpoints.

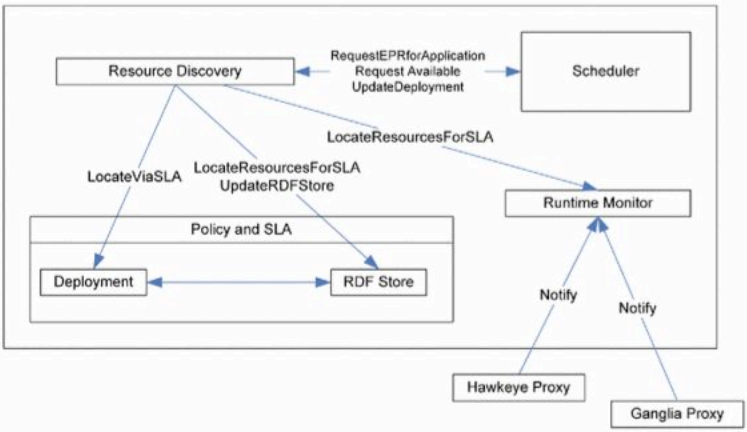

Fig. 3.7 Automatic Resource/Service discovery

- *RDF Store:* An interface to a flexible storage system based on RDF [22], which can store a variety of information without the need for a fixed schema.
- *Scheduler:* A client of the Resource Discovery subsystem.

It is appropriate to apply this pattern when the resource scheduling and allocation to endpoints depends on information from a variety of data sources, including static and dynamic information.

3.4.4.2 Sample Implementation

A sample implementation of the capability is the Automatic Resource Discovery component [16] developed in the BEinGRID project.

The Automatic Resource Discovery component is concerned with storing and retrieving Grid system information, such as Grid topology, computing and storage resources. It may also be used to store application-specific information. A key feature of the component is that the data is stored in an ontology, which supports reasoning over hierarchical data. In addition, system administrators can add deductive rules, which are automatically invoked when information is retrieved. The Automatic Resource Discovery has been designed to work with the Globus Toolkit 4 (GT4) [24] and it basically integrates a semantic layer on top of the GT4 Monitoring and Discovery System (MDS) [13].

The component is based on the RDF standard. This describes data in terms of classes, properties, and class members (instances, or objects), and informally, the data in the repository can be divided into the schema, defining classes, properties, and the relationships between classes, and instance data defining members of the classes and values for the properties.

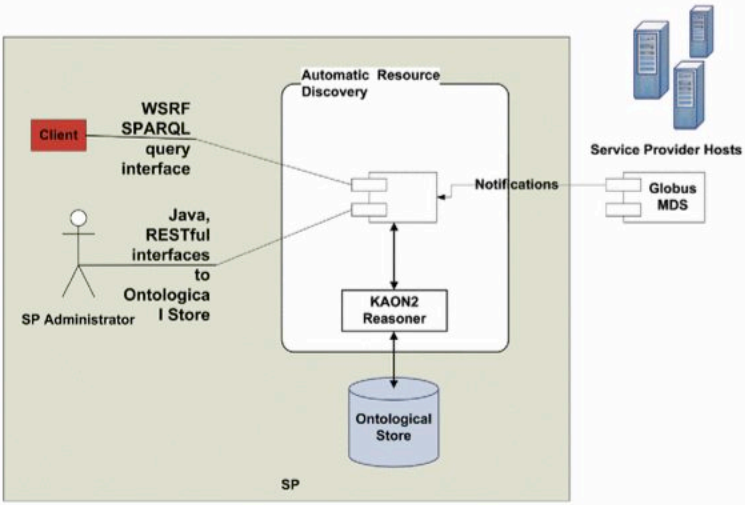

Fig. 3.8 High level architecture of the Automatic Resource Discovery component

Figure 3.8 presents a high level architecture of the component.

Resource Discovery can be done in several ways. The most popular one in a Grid environment is the adoption of the GT4 MDS. The Automatic Resource Discovery component is built on GT4 MDS and it augments the MDS index service by providing Query Service capable of executing SPARQL [23] queries.

The main advantages of using the component compared with services such as GT4 MDS is that it provides a simpler and less ad-hoc user interface, an extended information set, a common repository, and an interface for application-specific information. It reasons over an ontology rather than simple string matching of requirements against stored values.

3.5 A Sample Scenario and Integrated View of the Components

The following section presents a sample scenario involving the software components developed by the VO area. The purpose of the scenario is to show how the components can be adopted in the different phases of the VO lifecycle.[5]

The scenario defined is an application service provision scenario based on the Service Federation approach described previously in Sect. 3.2. The end-user asks for the provision of an application selecting it among a portfolio of applications that

[5] In accordance with the majority of the research projects that have investigated the VO paradigm, we consider the following phases of a VO lifecycle:

can be offered by an Application Service Provider (ASP). The ASP belongs to a VO Breeding Environment[6] (VBE a.k.a. Network of Enterprises), and as such it has a network of business relationships with Application Providers, Resource Providers, and Service Providers. The ASP identifies a business opportunity and decides to create a VO for the provision of the application. To this purpose, the ASP uses capabilities for VO Management.

Capabilities for VO Management can be offered in different ways. Figure 3.9 shows the two extreme cases graphically.

In the first case, on the left-hand side of the picture, there is the existence of a separate trusted third party ("VO Manager") in charge of establishing governance and rules for the federation. The VO Manager supports all the VO lifecycle phases providing services to set-up the VO, its identity management, and its infrastructure management. It is worth noting that this case differs from the Hub & Spoke model since there is not a main contractor and all the members of the VO are considered as peer (this is shown with the dashed lines in the picture).

In the second case, on the right-hand side of the picture, the VO Manager is actually a management & governance layer that can be distributed among the participant of the VO. Each independent members must deploy its own instance of the components, which must be configured to recognise the other peers of the VO.

The two approaches presented above have advantages and disadvantages: for example, the distributed management layer approach avoids the necessity for the trusted third party "VO Manager" but requires that VO members deploy VO Management services. Of course, there can also be intermediate situations where some of the functionalities (e.g. VO Registries) are offered by the VO Manager stakeholder and some are distributed among other stakeholders.

Table 3.2 presents the scenario's stakeholders, their operational and business objectives, and the business model each stakeholder follows.

The following sections details how the components can be used in the VO lifecycle phases.

- *VO Identification and formation*: it deals with identification of a goal, identification of potential partners, services, resources to achieve the goal, negotiation of agreements and policies, secure federation.
- *VO Creation*: it deals with the set-up of the VO infrastructure and creation of concrete instances of resources and services promised by the participants to achieve the goal.
- *VO Operation & Evolution*: it deals with the execution and monitoring of the tasks and business processes to achieve the goal of the VO, as well as with the management of the evolution of the collaboration (e.g. partner and service replacement, monitoring of the performance of the VO).
- *VO Dissolution*: is carried when the objectives of the VO have been fulfilled. During dissolution, the VO structure is dissolved and final operations are performed to remove all configurations, release resources of the partners, store the knowledge acquired for future collaboration.

[6]According to the ECOLEAD project, a VO Breeding Environment (VBE) represents an association or pool of organizations and their related supporting institutions that have both the potential and the will to cooperate with each other through the establishment of a "base" long-term cooperation agreement and interoperable infrastructure. When a business opportunity is identified by one member, a subset of these organizations can be selected to form a VO.

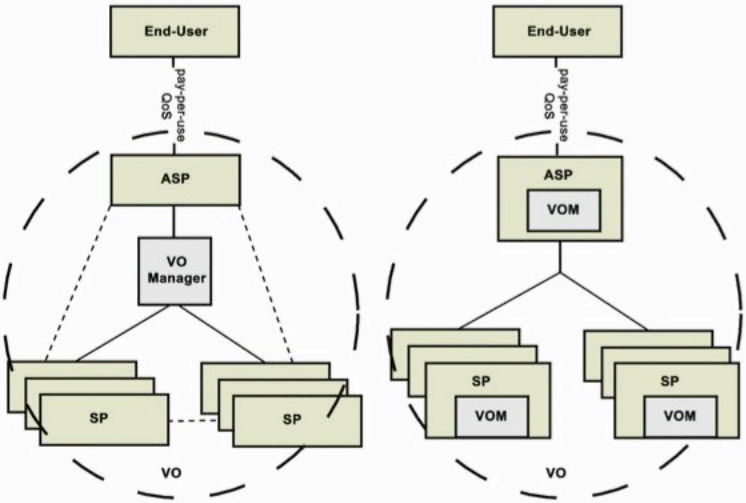

Fig. 3.9 Overview of the scenario with the VO Manager as stakeholder (*left-hand side*) and as fully distributed management layer (*right-hand side*)

3.5.1 VO Identification & Formation

The main purpose of this phase is to identify potential members of the VO, negotiate agreements, and start the secure federation process (i.e. create a circle of trust among them).

In this phase, the ASP needs to discover potential members of the VO on the basis of the capabilities they can offer. Of course, the ASP is aware of the application it needs to provide to the end-user and of the required capabilities. The ASP can query an internal catalogue containing all the partners of its VBE. Once the list of potential partners has been retrieved, ASP selects the ones it would like to collaborate with and sends an invitation to them.

The invited members, in case of acceptance, negotiate and/or sign agreements (including SLA agreements) and a policy model is defined for the VO on the basis of its objectives and of the specific members' policies. Lastly, identities of the VO members are translated into VO-wide credentials. The members are then published into a registry.

Most of the requirements of this phase can be addressed by the VO Set Up component, namely for partner identification and selection, secure federation start-up, and the publication or update of members in a VO.

Table 3.2 Scenario Stakeholders and their business opportunities

Stakeholders	Operational objective	Business objectives/values	Business model followed
Application Service Provider (ASP)	To provide application as a service on a pay-per-use model and on the basis of QoS.	To gain revenue from the provision of applications. Agility in providing its business (e.g. on-demand creation of virtual organisation to achieve the business). To save the costs of hosting all the services required for an application and of the management of infrastructure. Reduction of the total cost of ownership by outsourcing parts of the value chain. Transparent use of the Grid.	ASP as Grid User: an ASP has its existing system and wants to add some peripheral functionality. To this purpose, ASP acts as a user of the underlying Grid.
VO Manager (in case of stakeholder)	To provide capabilities to federate members (business enterprises) establishing the governance structure, rules and practices for the federation. To create and configure the underlying infrastructure for application execution.	To gain revenue from provision of federation and infrastructure management services. Cheap and fast access to Grid Computing facilities (Enabler).	Grid Enabler: it offers services to enable the collaboration between the organisations.
Service Provider (SP)	To provide VO business capabilities and resources.	To sell services and resources. Additional market by selling "component services" or "added values services", which may not be completely related to an existing business process.	Grid Service Provider: it provides service to many clients (following the classical model) however the services are of a different granularity and not necessarily consumed by the end client but also by other services.

3.5.2 VO Creation

The main purpose of this phase is the creation of business capabilities and configuration of the VO Infrastructure. This implies selection of the VO resources as well as the selection and creation of the VO service instances, and configuration of infrastructure services.

This phase can be addressed with the adoption of the Application Virtualization and Automatic Resource Discovery components.

The Application Virtualization is the component responsible for starting the process of (1) the creation of business instances that a provider has promised to offer and (2) the configuration of the VO infrastructural services. As described in the previous sections, this is done via the execution of the Virtualization Process.

The Automatic Resource Discovery component can be used to select the most suitable hosts inside the provider domain. In this scenario, the selection happens as part of the Virtualization Process.

It is worth mentioning that the approach adopted by the Application Virtualization component focuses on the separation of concerns between application provision and management of application execution. The virtualized application is exposed via a Gateway provider and access is controlled by the security services. A clear benefit of this approach is that the gateway abstracts the actual resources from the user, and enables a highly configurable and dynamic protection layer.

3.5.3 VO Operation and Evolution

In this phase, the identified partners contribute to the actual execution of the VO tasks by executing their business processes/applications. In our case, the VO has been created in order to provide an application to the end-user. Important features in this phase are the VO performance monitoring, policy enforcement (at the gateway), and exception monitoring and alerting.

When a VO member fails completely or behaves inappropriately, the VO manager may need to dynamically replace such a partner. This evolution may involve discovering new business partners, re-negotiating terms, and providing configuration information, as done in the identification and formation phase.

The operational phase is not addressed by the VO area components but can be addressed by components of other BEinGRID technical areas, such as the Security and SLA ones.

The evolution phase is partially addressed and only for the VO Member replacement. This is done in the same way as VO Identification & Formation phase via the VO Set Up.

3.5.4 VO Dissolution

The dissolution phase is carried out when the objectives of the VO have been fulfilled. During dissolution, the VO structure is dissolved and final operations are performed to remove all configurations and to release partner resources. On completion of this step, the members of a VO return to be members of a VBE.

This phase is partially executed by the VO Set Up and the Application Virtualization components.

In the case of the Application Virtualization, the "Graceful Shutdown" process is executed.

This pragmatically means the execution of a process that removes all the entries relating to the service instance to be destroyed from the Service Instance Registry, removes all the configuration information from the security services (or destroying the security service instances created for the specific instance) and SLA services. If all the clean-up steps are executed without exceptions, the actual service instance is destroyed.

After this operation, the VO members should return to being a VBE member. This is executed by the VO Set Up component that removes VO context information from the VO Member Registry.

3.6 Lessons Learnt

With respect to the analysis and support of the case studies, the VO Management area has summarised its experience in terms of identification of good practices, presentation of the main lessons learnt, and production of some recommendations for business cases relating to business-to-business collaboration.

During the analysis and support of the case studies, three different cases of collaboration and, in general, adoption of the VO paradigm have been observed. For each one of these cases, a good practice has been identified that can be followed by other business cases having similar requirements. These are:

- The case of the Grid implementation in the textile sector [3]: this is a good practice concerning the adoption of VOs for static collaboration.

  In this kind of collaboration VO members are well known and do not generally change during the lifecycle of the VO. Agreements, if present, are generally defined a priori and there is trust a priori between participants.

  The case of the digital district for textile seems to be a good practice for this kind of collaboration. The approach adopted by this business experiment is to use a Grid portal and portlet to integrate Grid technologies for resource sharing and collaborative tools.
- The case of the virtual hosting environment for online gaming [2], which identifies a good practice concerning the adoption of the VO for ad-hoc dynamic collaboration.

With ad-hoc dynamic collaboration, we refer to the case in which case the VO members have to be dynamically identified on the basis of the business goal of the VO. Of course, after the identification phase, policies and the VO agreement have to be negotiated and, generally, there is no trust a priori among the partners, so trust & identity management is a key factor for the success of this kind of collaboration.

- The case of the Grid study in oil & gas simulations [4], that appears to be a good practice concerning the re-use of already existing VO infrastructure (such as the EGEODE one [8]) for sharing of computational and data resources.

  In some business cases, it may be useful to re-use existing VO and solutions already developed in the e-science community provided they fit well within the scenario. This is the case, for example, of business applications such as the finance, automotive, pharmaceutical, applications etc., that foresee as mission critical the execution of simulation, analysis of data sets and, in general, present HPC features.

In terms of lessons learnt, we have understood that, despite a strong research interest in the VO paradigm, the implementation of VO for ad-hoc dynamic collaboration (referred also as dynamic VO management) is still immature in terms of interest and adoption in e-business.

Moreover, we have observed that a current pattern (also followed by one of the BEinGRID case study) is trying to re-use existing research infrastructures in e-business mainly developed in e-science contexts. The objective is also to rely on already existing VO and solutions for VO management for computational and data resources sharing. Even if this seems to be the natural choice, re-using these existing infrastructures and solutions is suitable just for specific business cases that foresee as mission critical the execution of simulation, analysis of large dataset and, in general, present HPC-like features.

Other business cases for which, for example, ad-hoc dynamic collaboration is required or that foresee provision of services as applications should avoid this approach since, at least in its current state, the above solutions do not offer capabilities required for ad-hoc dynamic collaboration such as, for example, a rich trust management model, ability to separate among collaboration contexts and to react to contextual changes, etc. These business cases should instead consider re-using or building on top of results and findings of other projects such as TrustCoM, Akogrimo, NEXTGRID, BREIN.

Lastly, with respect to the capabilities for Virtual Organization management provided by Grid middleware, this area has observed that none of the most adopted Grid middleware offers all the capabilities required for VO management. According to our experience, this limitation has a negative impact on the adoption of VOs mainly in business domains. In addition, despite the promised "paradigm shift" (from e-science to e-business) current implementations of the most adopted middleware allows/encourages the adoption of VO paradigm mainly for computational resource sharing without paying adequate attention to the Business-to-Business collaboration aspects that underpin a commercially viable use of the VO paradigm in any business context.

In contrast, we observe the emergence of complementary technologies, such as Web Services-based Federated Identity Management or Web 2.0 based on models that place human and knowledge at the centre of the process. We believe this area to be useful to investigate emerging technologies to fulfil some lacks of the most adopted grid middleware.

A final consideration focuses on Security for VO as well as Semantics, two challenging areas proposed by the VO Management area which has had a low interest with respect to the expected one. The main reason, in our opinion, is that they have been considered by the case studies at the same time too difficult to address in the project lifetime and not mission critical. This is a big mistake mainly for security aspects that, if not addressed in early stages of development, may render difficult the process of re-engineering a prototype and, in some cases, may also prevent a potentially good solution to gain its market.

On the basis of these lessons learnt, we propose some recommendations. We essentially propose to follow one of the identified good practice, to take into consideration Grid as well as complementary technologies to address issues relating to VO management, to not underestimate the importance of security if you want to use a collaborative model and, lastly, to re-use the components developed by the BEinGRID project to gradually introduce dynamicity in collaborative scenarios.

3.7 Business Benefits

There are several business benefits associated to the results of the VO Management area.

It is worth mentioning that in general the results of the VOM area promotes an innovative model for VO that:

- Foresees an enhanced identification and formation phase, via selection of capabilities and members on the basis of SLAs, Identification of the risk associated to a collaboration, adoption of trust to mitigate risks;
- Relies on distributed trust management model;
- Fosters the adoption of Virtualization mechanisms of application and resources, via concepts such as the Virtual Hosting Environment (VHE) and B2B Gateway.

In terms of business benefits, for instance, the VO Set-up component and the capabilities implemented allows for agility in responding to new needs/requirements and improved time-to-market (by set-up of a VO when a new opportunities arises); improved trust in Business to Business interactions, and dealing with the geographical and organizational distribution of teams and computational resources.

In terms of innovation, with respect to other solutions for VO management, the model of the VO Set-up is better suited to the way enterprises thrive nowadays where new opportunities rise and fall quickly and where the environment is very prone to change. The VO Set-up allows for more flexible, business-driven interactions. Trust is established from the VO Set-up through to the security components in particular the Security Token Service.

The Application Virtualization, instead, addresses the separation of concerns between application provision and SOI operational management. The virtualized application is exposed via a Gateway and the configuration of infrastructure services (potentially provided by third parties) for managing non-functional aspects of the application is done in a transparent way for the application consumer. So, the added value is mainly in the automatic configuration of third party management services such as SLA and security. The adoption of the Gateway avoids direct access to the resources of a SP and access is controlled by the security services.

In terms of business impact, the Application Virtualization allows ASPs to expose their applications in a simple and manageable way without being involved in the management of the enabling infrastructure. This increases flexibility and allows a separation of concerns between application provision and management, and enables the transition towards a SaaS model.

In terms of exploitation opportunities, the VO Set Up component can be used in combination with components of the security area of the BEinGRID project to manage the life-cycle of circles of trust between providers targeting the Federated Identity Management market.

For the Application Virtualization, the selected strategy for this component is to be used in combination with components of the security and SLA areas of the BEinGRID project to coordinate different service execution environments to allow secure and manageable application exposure.

The idea behind this strategy is to exploit this component as a brokerage solution for different cloud providers.

3.8 Conclusion

We draw our conclusions from two perspectives.

The first perspective is the one of the VO Management area that, in our opinion, have achieved results that can be considered satisfactory. The components designed and developed cover a wide set of functionalities required to support the VO life-cycle mainly in terms of governance. We believe useful also the results in terms of patterns, capabilities and requirements that may help in improving already existing architectural solutions.

The VO Set-up is, in our opinion, a quite interesting and distinctive development with respect to other solutions for VO management, the Automatic Resource Discovery is a good improvement of the MDS of GT4 and represent a interesting work in integrating a semantic layer on top of the most adopted Grid middleware, and eventually the Application Virtualization develops a simple but effective approach to configure infrastructural services (potentially also provided by third party) for business application execution in an automatic and transparent way.

The components have been designed and developed to be modular and re-usable in several contexts. Our objective is to allow the deployment of the components into business scenarios with less effort as possible.

The second perspective for our conclusions is the one of adoption of Grid technologies for VO management in business contexts. For this perspective, the conclusion is that the traffic light is currently yellow.

Most of the visible work done so far with Grid technologies is the creation of wide research infrastructures (e-Infrastructure in EU, cyberinfrastructure in US) and so the re-use of this work also in business scenarios, as already evidenced previously in this chapter, appears to be a reasonable choice. But today this comes with a cost: mainly resource sharing aspects of the VO paradigm can be adopted by business scenarios re-using such work.

If computational resource sharing is not key and there are requirements for ad-hoc dynamic collaboration, such as agreement negotiation, trust establishment among partners, most adopted Grid middleware and solutions are still immature with the exception, of course, of some results coming from specific projects. The BEinGRID project is a source of requirements, capabilities, patterns and components, business practices, etc. that may potentially turn to green the traffic light by allowing improvement of current Grid solutions and sped-up their evolution from e-science to e-business.

References

1. Brein European IST-FP6 project, http://www.eu-brein.com/
2. Business Experiment 09 (BE09) The VHE for on line gaming application, http://www.beingrid.eu/be9.html
3. Business Experiment 13 (BE13) Virtual Laboratory for Textile, http://www.beingrid.eu/be13.html
4. Business Experiment 18 (BE18) Seismic Processing and Reservoir Simulation, http://www.beingrid.eu/be18.html
5. Business Experiment 20 (BE20) TAF GRIDS, http://www.beingrid.eu/be20tafgrids.html
6. L.M. Camarinha-Matos, I. Silveri, H. Afsarmanesh, A.I. Oliveira, Towards a framework for creation of dynamic virtual organizations, in Collaborative Networks and Their Breeding Environments (Springer, Berlin, 2005)
7. European Collaborative Networked Organisations Leadership Initiative, European IST-FP6 project, http://ecolead.vtt.fi/
8. Expanding GEOsciences on DEmand (EGEODE), http://www.egeode.org/
9. I. Foster et al., The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, http://www.globus.org/alliance/publications/papers/ogsa.pdf
10. A. Gaeta, F. Orciuoli, N. Capuano, D. Brossard, T. Dimitrakos, A service oriented architecture to support the federation lifecycle management in a secure B2B environment, in Proceeding of the Workshop Experiences on Service Oriented Infrastructure and the Grid as Foundation for the Next Generation of Business Solutions, eChallenges 2008, Stockolm, Sweden, 22–24 October 2008
11. E. Gamma, R. Helm, R. Johnson, J.M. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software (Addison-Wesley, Reading, 1995)
12. Ganglia—A Scalable Distributed Monitoring System for High-performance Computing Systems, http://ganglia.info/
13. Globus Toolkit Information Services: Monitoring & Discovery System (MDS), http://www.globus.org/toolkit/mds/
14. Gridipedia Technical Solution—VO Set-up, http://www.gridipedia.eu/vo-setup.html

15. Gridipedia Technical Solution—Application Virtualization, http://www.gridipedia.eu/application-virtualization.html
16. Gridipedia Technical Solution—Automatic Resource Discovery, http://www.gridipedia.eu/automatic-resource-discovery.html
17. Gridipedia Technical Solution—Security Token Service, http://www.gridipedia.eu/security-token-service.html
18. Hawkeye—A Monitoring and Management Tool for Distributed Systems, http://www.cs.wisc.edu/condor/hawkeye/
19. NextGrid IST-FP6 project, http://www.nextgrid.org/
20. OASIS UDDI Spec TC, http://uddi.org/pubs/uddi-v3.0.2-20041019.htm
21. O.F. Rana, D.W. Walker, Service Design Patterns for Computational Grids, 7 July 2003
22. Resource Description Framework (RDF), http://www.w3.org/RDF/
23. SPARQL Query Language for RDF, http://www.w3.org/TR/rdf-sparql-query/
24. The Globus Toolkit, http://www.globus.org/toolkit/
25. TrustCoM European IST-FP6 project, http://www.eu-trustcom.com/