Application Deployment using Catallactic Grid Middleware

Liviu Joita, Omer F. Rana
School of Computer Science and the Welsh eScience Centre, Cardiff University
Queen's Buildings, 5 The Parade, Roath, Cardiff, CF24 3AA, UK
Phone: +44 (0)29 2087 4812
{l.joita, o.f.rana}
@cs.cardiff.ac.uk

Pablo Chacín, Isaac Chao, Felix Freitag, Leandro Navarro
Computer Architecture Department, Technical University of Catalonia
Jordi Girona 1-3, Campus Nord D6
Barcelona 08035, SPAIN
Phone: +34 (93)401-1055
{pchacin,ichao,felix,leandro}
@ac.upc.edu

Oscar Ardaiz
Department of Mathematics and Informatics, Public University of Navarra
Campus de Arrosadia, Pamplona 31006, SPAIN
Phone: +34 (948)168076
oscar.ardaiz@unavarra.es

ABSTRACT

In this paper we describe an application deployment using a Catallactic Grid-enabled middleware, which is based on the Catallaxy "free market" self-organisation approach described by von Hayek [7], who understood the market as a decentralised coordination mechanism opposite to a centralised command economy. The implementation makes use of Globus Toolkit, JXTA and WSRF. The paper envisages the resource virtualization in the WSRF context as the main driver for a proper connection middleware-base platform (on the broad scenario of grid applications).

Category and Subject Descriptors:

C.2.4 [Computer Systems Organization] Distributed Systems

General terms:

Design, Experimentation

Keywords:

Middleware, Grid, Economic-based Allocation

1. INTRODUCTION

There has been significant interest in utilising an economic paradigm for exchanging Grid resource and services [3]. A key motivation behind this approach is the capability to schedule access to services based on a market mechanism (such as auctions), thereby allowing a more fair and efficient approach to share resources in high demand. Most existing approaches rely on the existence of a centralised broker that coordinates resource access, and is generally implemented over existing Grid middleware. We propose an alternative approach, based on the Catallaxy mechanism proposed by von Hayek [7], which does not need to support such centralised brokers. Catallaxy makes use of a "free market" self-organisation approach, which enables prices within the market to be adjusted based on particular demands being placed on particular scarce services.

MGC '05, November 28- December 2, 2005 Grenoble, France

A key issue is to utilize this approach as the basis for realizing resource allocation in Application Layer Networks (ALNs). The term ALN integrates different overlay networks, such as Grid and Peer-2-Peer (P2P) systems, as virtual application-based interconnection topologies that may be implemented over physical topologies of the Internet. We use the term "agent" to refer to an autonomous service provider or user, having the capability to update and modify the services they offer/use, and to determine how much information about these services should be made accessible to others.

The Catallaxy approach is a coordination mechanism for systems consisting of autonomous decentralized agents, and based on constant negotiation and price signalling between agents [4]. Catallaxy is a way to inform the individual (agent) about the knowledge that may be contained by other agents, and provides an exchange of information that leads to the generation of prices which comply with the value every individual (agent) assigns to the respective information [2]. Catallaxy therefore leads to the development of self-organizing individuals (agents) that are highly dynamic, thereby leading to systems which behave in a Peer-2-Peer fashion. Such an approach is particularly suited to "Open Systems", where detailed knowledge about particular agents may not be known apriori.

1.1 Catallaxy and Grid markets

The Catallactic "free-market" mechanisms can be applied to Grids. In Grids two interrelated markets appear: the resource and service markets. In the Grid resource market, resource providers sell their computational, storage, bandwidth or tool resources to resource buyers. The traded good are physical resources which will be used by buyers to execute their own applications. In the Grid resource market there are a large number of resource seller and resource buyer, thus Catallactic "free-market" mechanisms are needed.

In the Grid service market service providers sell services to service clients. The traded good in the service market are services which provide a particular application functionality: a transcoding service, a query execution service, a molecule docking service. The service market buyer is interested in a using a particular application, not in using his own application code. In the Grid service market there are a large number of service seller and service buyer, thus Catallactic "free-market" mechanisms are also needed.

Moreover, service providers can buy resources at the Grid resource market to provide services in the Grid service market.

F.e. a transcoding service provider can buy computational resources in the Grid resource market to execute its transcoding application for a particular client request. Both markets are dependant on each other but can operate autonomously applying catallactic mechanisms as has been shown by simulations in [2].

1.2 Query execution application and Catallaxy mechanism

The application prototype Cat-COVITE utilises concepts of a prior application for managing interactions between product designs, specification engineers and users within the building/construction industry [10]. This existing application involves searching through distributed product catalogues (modelled as a Web Services-enabled database) using a Grid-based distributed search strategy. The particular approach adopted in the Cat-COVITE application is employable in a significant number of other industrial applications which make use of distributed databases. In this way, the lessons learned from this application, and integration with the Catallactic middleware may find use by a very wide community.

1.3 Introduction to WS-Agreement

Web Service Agreement (WS-Agreement) protocol specification has been developed by the GRAAP Working Group (Grid Resource Allocation and Agreement Protocol WG) of the Scheduling and Resource Management (SRM) Area of the Global Grid Forum (GGF) [8]. WS-Agreement is an XML language protocol for specifying an agreement between a resource/service provider and a consumer [13]. It is generally aimed to be a one-shot interaction, and is not directly intended to support negotiation. However, it can form a useful basis on which negotiation between two parties may be conducted. WS-Agreement is used in Cat-COVITE, and forms the basis for choosing between multiple service and resource providers. The service provider acts as the agreement provider, while the service consumer as the agreement initiator. Section 3.2 will further detail the concepts and requirements of WS-Agreement in the Cat-COVITE prototype context.

2. CATALLACTIC MIDDLEWARE

The implementation of Catallaxy in real world Grids requires the design of Catallactic middleware which offers a set of generic negotiation mechanisms, allowing specialized strategies and policies to be dynamically added as plugins. It is intended that the middleware offers a set of high level abstractions and mechanisms to locate and manage resources, locate other trading agents, engage agents in negotiations, and adapt to changing conditions. Technical issues that need to be evaluated in the context of a scalable Catallactic middleware are discussed in this section.

2.1 Requirements

Scalability in highly dynamic environments. The Catallactic middleware should be able to address scenarios with thousands of nodes in a highly dynamic environment, where nodes enter and leaves the network frequently. The dynamism in the network configuration implies that information about the system should be maintained at a minimum (avoiding global topological information) and that updates must be easy and efficient.

Handle heterogeneous environments. Scale also implies a high level of heterogeneity in applications, the underlying platform, resources, service properties of providers, and availability of nodes (some will be quasi permanent, other will enter and leave).

Compatibility with different base platforms. Different base platforms should be supported, thereby leading to the definition of generic APIs. Some adaptors may be needed to translate this generic model to the specific model used by each platform. This translation mechanism could harm the performance of the system if transformations are complex or frequent.

Component self-organization. The dynamicity of the network prevents an a priori configuration of the peers or the maintenance of centralized configuration files. Peer need to discover continuously the network characteristics and adapt accordingly, what requires a distribution of some important system functions like security, resource management, topology management, among other, which have been traditionally reserved to very specialized nodes.

Support different implementation architectures. The middleware may be deployed under different configurations. Each component should therefore not make any assumptions about a specific distribution of functionalities. Also, different architecture models will lead to different interaction patterns between the base platform, the applications and the Catallactic middleware. Under some scenarios, the applications will make request for resources to the base platform, which will in turn, forward it to the Catallactic middleware (probably, using a component specifically modified to interact with it). In other scenarios, the application will make requests directly to the Catallactic middleware (probably, using a component specifically modified to interact with it) which will interact with the base platform to fulfil it.

2.2 Concepts and design guidelines

A P2P approach has been adopted, leading to the following properties: Decentralization, there is neither single or centralized coordination nor administration point. Symmetric interaction between peers, all peers are simultaneously clients and servers both requesting and providing services. Non-deterministic topology, At any moment in time, the overall topology of a P2P network is completely unpredictable. The set of nodes that makes up the network may vary constantly. Dynamic and virtual allocation of communication paths, due to communication paths between peers are created dynamically based on various factors, like network conjunction or intermediate peers' state.

To achieve these objectives, economic agents are decoupled from the underlying ALN. Due to the potential variability of the ALN's topology, the middleware needs to implement different algorithms to adapt to different scenarios (for example, adaptation to sudden changes in the network or disruptions). Therefore, agents should not need to be aware of the overlay topology or make any assumption about its communication mechanisms. However, isolating the economic agents from the agent discovery process should preclude the integration of economic information (for example, success ratio of negotiations with other agents) into the adaptation mechanisms used by the middleware.

Many of the APIs for the different Catallactic middleware layers will handle information that depends on the specific application domain and base platform used for implementation. For example,

the resource discovery will return a list of resource descriptions, which depends on the kind of resources used by application: processors for a Grid, bandwidth for a Content Distribution Network (CDN) etc. An XML schema is therefore supported and can be extended or specialized to each specific implementation.

2.3 Architecture

A layered architecture provides a clear separation of concerns between the layers. This facilitates the construction of a more adaptable system, as the upper layers can be progressively specialized (by means of pluggable rules and strategies) into specific application domains. The following 5 layers are supported:

Application Layer: is given by the domain specific end user applications like collaboration tools, problem solving environments, and many others. Applications rely on the base platform for functions like communication and platform level resource management. However, applications can have application level resources, like a virtual meeting room in a collaboration tool or a matrix resolution algorithm in a scientific environment. The interaction model between the application layer and the Catallactic middleware is application and middleware dependent. Application can interact directly with the Catallactic middleware (becoming Catallactic enabled applications) to manage their resources or they can interact transparently by means of the base platform they are built on.

Economics Algorithms Layer: Implements economic algorithms for resource allocation. These algorithms should be domain and platform independent. This layer includes a set of interacting agent services that play the roles of Sellers and Buyers in service and resource markets. Also, in this layer are extensions and specializations of the functionalities provided by the underlying framework, to adapt them to the specific ALN and the resource allocation polices in place.

Economics Framework Layer: offers the primitives that support the implementation of Catallactic algorithms, such as finding peers agents to negotiate, starting negotiation, making a bid, etc. It is dependent on the agent platform being used, but should be independent of the application domain and the base platform. This layer is structured as a set of basic entities that model the interaction of trading agents in a market to exchange goods. These abstract entities are the building blocks of the Catallactic algorithms.

Peer Agent Layer: Platform that hosts the Catallactic agents offering a generic P2P application model with abstractions for the discovery and communication mechanism, and a generic interface with the underlying platform. This layer covers the basic functions that will be used by all implementations; it is responsible for interfacing with the underlying platform and complementing it when necessary.

Base Platform Layer: Supports applications and Catallactic middleware. It is (potentially) domain specific. The model of interaction with the Catallactic middleware depends on the architecture of the base platform, but in general will require the implementation of a connector, which routes the request for resources to the corresponding economic agents. In some cases, this might even require the re-implementation of some core platform components, like the GRAMs (Globus Resource Allocation Managers) in Globus [6].

3. CAT-COVITE APPLICATION – REQUIREMENTS AND CONCEPTS FOR CATALLAXY

In the original COVITE prototype [10], suppliers and purchasers collaborate to procure supplies for a particular construction project by using the COVITE application. These projects are usually unique, very complex and involve many participants from a number of organizations acting collaboratively. These participants work concurrently, thus requiring real time collaboration between geographically remote participants. Each consortium is in effect a virtual organization (VO). The application permits to search across a large number of supplier databases to retrieve products matching a criteria set by the purchasers or contractors. The application enables a search to be conducted, making use of the cluster of machines in a Grid network to retrieve the matching products.

The COVITE prototype application is divided into two functional services: Security Service and Multiple Database Search Service (MDSS). The Grid enabled MDSS enables searching across a large number of Supplier Databases (SD) using a Master Grid Service (MGS) instance via a cluster of machines in a Grid network. In this instance, the query is defined according to a data model that is specific to a given application domain. Arbitrary text queries (as in the Google.com search engine, for instance) are not allowed.

The COVITE application enables these VOs to plan, schedule, coordinate, and share components between designs and from different suppliers. The ability of a free-market economy to adjudicate and satisfy the needs of VOs, in terms of services and resources, represent an important feature of the Catallaxy mechanism. Such VOs could require large amount of resources which can be obtained from computing systems connected over simple communication infrastructure such as Internet. There are also possibilities for these VOs to try maximizing their own utilities on the market.

3.1 Cat-COVITE and the Catallactic Grid markets

Different components of the COVITE application can be mapped to actors in Catallactic Grid markets. The resulting Cat-COVITE application will permit application client to access larger sets of service and resource in a more cost-efficient manner. Figure 1 shows the Cat-COVITE components and related Catallactic agents as buyers and seller in the Grid service market and the Grid resource market. The Cat-COVITE application is composed of three main components, the Master Grid Service, (a type of Complex Service), the Query Job Execution Service, (a type of Basic Service), and query execution resources (computational resources).

The MGS Complex Service is the buyer entity in the service market, and the Query Job Basic Service is the seller entity on the service market.

A MGS Complex Service includes the following activities:

- Translates a request to a Basic Service - query service.
- Starts parallel negotiation with a number of agents representing Query Job Execution Services (Basic Services).

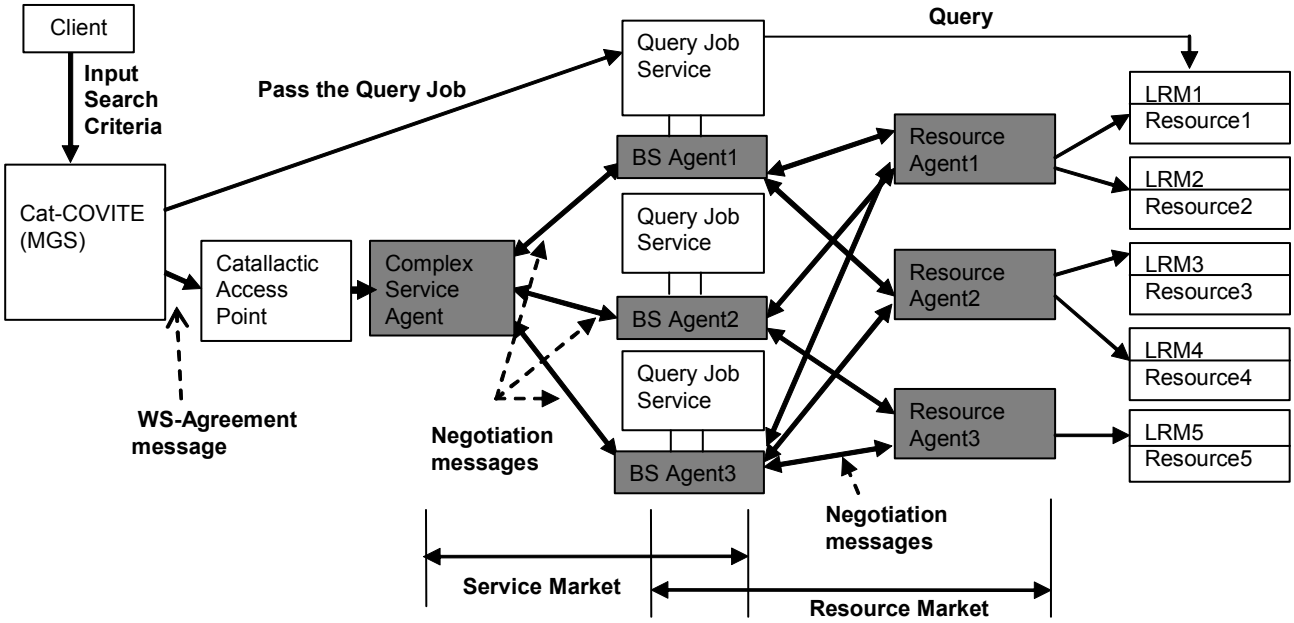

Figure 1 – Cat-COVITE markets and Catallactic Agents

- Sends a query to a list of Query Job Execution Services (Basic Services).

The Query Job Execution Basic Service involves query execution on a particular database and consists of:

- Query Job Execution Environment (offers the deployment of "slaves", which are able to execute the query).
- Translation of query to resource requirements.

Within the Cat-COVITE application, the Query Job Execution Basic Service needs to support response time and the quality and quantity of the search. With this goal the Query Job Execution Basic Service buys resources in the resource market. Resource seller entities are able to provide a set of resources via the Local Resource Manager (LRM). The Resource Agents act on behalf of these LRMs, which hide the physical resources behind them.

The Query Job Execution Basic Service is the buyer entity in the resource market, and the Resource Local Managers are the seller entity on the resource market. The main functionalities of Basic Service agent at the resource market are:

- Co-allocation of resources (resource bundles) by parallel negotiation with different resource providers (local resource manager entities).
- Informing the Basic Service about the outcome of the resource negotiation.

3.2 WS-Agreement – concepts and requirements in Cat-COVITE application

An agreement consists of several parts, according with the WS-Agreement draft [13]: the section of the agreement name, which is optional; the agreement context includes the parties to an agreement, reference to the service(s) provided in support of the agreement, and the lifetime of the agreement; the agreement terms, which describe the agreement itself, can contain: the *service description terms,* which provide information needed to instantiate or otherwise identify a service to which this agreement pertains. And the *guarantee terms*, which specify the service levels that the parties are agreeing to.

The example scenario in terms of the Cat-COVITE application proposed for the prototype interacting with the middleware it is as: "I (MGS) need to run a query search job. I (MGS) send an Agreement Offer (AO), based on the Agreement Template (AT) downloaded from the *Catallactic Access Point (CAP)*, to the *CAP* to finding a query job service. The *Complex Service Agent*, acting on behalf of the *Complex Service* (MGS) chosen by the *CAP*, negotiates with the *Basic Service Agents (*in the CATNETS environment) for query services to fulfill the job". The agreement template (AT) specifies the service description elements that are allowed by the factory which advertises it. We created an agreement template specific for Cat-COVITE application. The agreement offer (AO) is initiated by the agreement initiator (the *MGS).* The agreement acceptance is the same as the agreement offer if the agreement provider accepts the conditions of the offer. If the agreement provider doesn't accept the offer, the agreement initiator has to send another agreement offer. The agreement offer compliant with the Cat-COVITE agreement template looks like:

```
<?xml version="1.0" encoding="UTF-8"?>
<AgreementOfferLite>
  <Name>QueryServiceTemplateLite</Name>
  <Context>
    <AgreementInitiator>
      <Name>Your Name</Name>
    </AgreementInitiator>
  </Context>
  <Terms>
    <Executable> SELECT IDProduct, ManufacturerName, Price
FROM Product ORDER BY Price DESC
    </Executable>
    <PayForService>100</PayForService>
  </Terms>
</AgreementOfferLite>
```

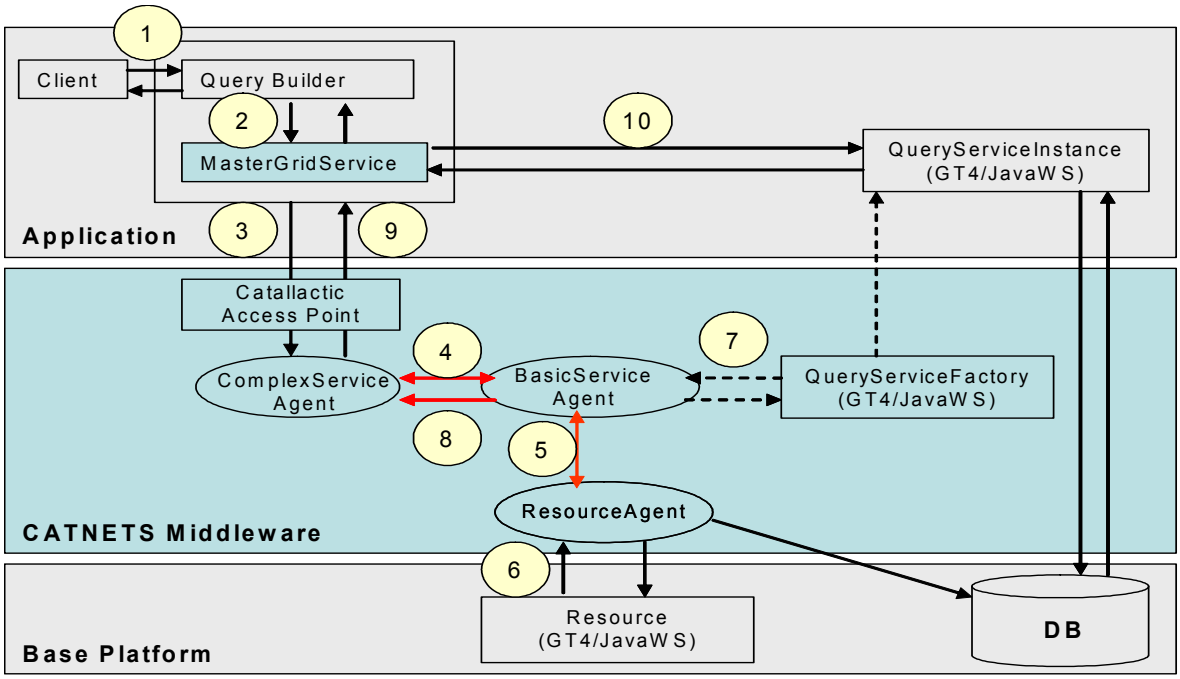

Figure 2 – Integration of Catallactic Middleware and Cat-COVITE Grid Application

4. PROTOTYPE IMPLEMENTATION

4.1 Middleware Implementation

The middleware is implemented as a set of simple, specialized agents using DIETs light-weighted agents platform [5]. Framework agents support the basic functions needed to implement economic algorithms, like access to markets, negotiations, traded goods, trading agents, etc. Peer Agent Layer agents implement the low level functionalities to support system execution: overlay network, object discovery, communications. A detailed description of middleware implementation is found in [1].

The Overlay Network, Object Discovery and Communication functions are implemented using JXTA Peer Resolver Protocol in a network of Rendezvous Peers that use a DHT to maintain and route messages among nodes [12]. The management of local resources, in this case services offered by the service providers, is based on the WSRF framework [14] offered by GT4.

Searches for grid services are propagated by decentralized mechanism implemented using JXTA Peer Resolver Protocol, allowing for complex multi-attribute queries. Remote nodes previously registered as resolvers for the appropriate type of query will use the GT4's Index Service to effectively resolve the query against the specific search attributes. In a first implementation this mechanism allows for basic service search, and will need to be further extended and tuned for performance, especially concerning the query algorithms on top of the JXTA overlay network.

4.2 Integrating Cat-COVITE with Catallactic Middleware

Here we describe how the application and the middleware can be integrated. Figure 2 depicts a high level view of the architecture, identifying the placement of logical components along the three layers: the application layer, the Catallactic middleware layer and the base platform layer.

At the application layer, the application must provide an interface to the middleware which must issue the requests for services to the middleware and use the references to service instances provided by the middleware to execute such services.

At the middleware layer, a set of agents provide the capabilities to negotiate for services and the resources needed to execute them. The Complex Service agent acting on behalf of the application initiates the negotiation. Basic Service and Resource agents manage the negotiation for services and resources, respectively. Also, a Service Factory is provided to instantiate the service on the execution environment selected during the negotiation process.

Finally, at the Base Platform layer, a Resource is created to manage the allocation of resources to the service. This resource represents the "state" of the service from the perspective of the middleware (notice, this doesn't mean the service is stateful from the perspective of the application).

The flow of information among the logical components can be summarized as follows: a Client issues a request to the application (1), which builds a query and requests the execution of query to the Master Grid Service, MGS (2). The MGS contacts a Catallactic Access Point asking for a WS-Agreement template for such a service. The MGS fills in the template and sends back an Agreement Offer (3).

The Complex Service Agent initiates catallactic mechanisms to find appropriate Basic Services and Resources. The Complex Service Agent uses discovery mechanisms implemented in the middleware Peer Agent Layer to locate Basic Service Agents providing Query Service. When a number of Basic Service Agents are discovered, it starts negotiations with one of them (4). In turn such Basic Service Agent must discover and negotiate with a

Resource Agent for query execution resources in the resource market (5). Negotiations are implemented by the Economic Framework Layer, where different protocols can be used depending on the agent's strategy.

When an agreement with a Basic Service Agent is reached, the Resource Agent instantiate a Resource to keep track of the allocated resources and returns to the Basic Service Agent a handle for this resource (6). Consequently Basic Service Agents use the Query Service Factory to instantiate the Query Service on the selected GT4 container (7).

Basic Service Agent returns to the Complex Service Agent the reference to the newly instantiated Query Service and the related resource(s) (8). The reference to the Query Service is returned to the MSG (9), which uses it to invoke the service, passing the query to be executed (10).

4.3 Physical Deployment on GT4 containers

The logical architecture depicted in the previous section can be implemented in different ways depending on the base platform used. We describe a specific implementation on a GT4 [9] based platform, and based on the following assumptions. First, the services are previously deployed on a set of GT4 containers. Second, the only "resource" considered in the negotiation are the "rights" to execute the service on a specific container. Finally, the service can be instantiated on a container using a generic factory.

The Application resides in a host (or series of hosts) where also resides the Master Grid Service (interface with the middleware) and the Complex Service Agent, which represents the application in the negotiation process. On each Grid Container (GT4) where the Query Job Execution Service is deployed, resides the corresponding Basic Service Agent, which negotiates with the Complex Service Agent for access to the Query Job Execution Service. In this container also resides the Resource Agent, which negotiates with the Basic Service Agent for the rights to execute the Query Job Execution Service in this container. Finally, a Resource is created as result of the negotiation process, which represents the "rights" to execute the service in this container.

5. CONCLUSIONS

The Catallactic middleware – based on the concept described by von Hayek has been presented, along with an application that makes use of this middleware. An implementation has also been achieved using GT, JXTA and WSRF.

We observed that the negotiation protocol will need to go beyond the current WS-Agreement specification to handle the complexities of the bargaining process. Also, WSRF specifications are still too general and do not offer a clear approach managing virtual resources.

We envisage the growing tendencies to service oriented architectures and resource virtualization as the main drivers for an increasing integration of the resource negotiation mechanisms offered by middleware with the base platforms on the broad scenario of grid applications.

Next steps include the development of measurement components to support the middleware, to assess the performance of the developed application. Also, we envision proving this architecture in other application models to evaluate qualitatively the architecture under diverse scenarios.

6. ACKNOWLEDGMENTS

This work was supported in part by the European Union under Contract CATNETS EU IST-FP6-003769, and the Spanish Government under Contract TIC2002-04258-C03-01.

7. REFERENCES

[1] Ardaiz O., Chacín P., Chao I., Freitag F., Navarro L. An Architecture for Incorporating Decentralized Economic Models in Application Layer Networks. Smart Grids Technologies Workshop, Utrecht, Holland, 2005.

[2] Ardaiz O., Artigas P., Eymann T., Freitag F., Navarro L., Reinicke M. The Catallaxy Approach for Decentralized Economic-based Allocation in Grid Resource and Service Markets. Special Issue on Agent-based Grid Computing, International Journal of Applied Intelligence, accepted.

[3] Buyya R., Abramson D., Giddy J., and Stockinger H. Economic Models for Resource Management and Scheduling in Grid Computing. The Journal of Concurrency and Computation: Practice and Experience (CCPE), Wiley Press, May 2002

[4] CATNET Project deliverables. D3: Catallaxy Evaluation Report. March 2003. Available at: http://research.ac.upc.es/catnet/pubs/D3.pdf

[5] Diet Agents, http://diet-agents.sourceforge.net/, Oct. 2005.

[6] Foster I., Kesselman C., Lee C., Lindell R., Nahrstedt K., Roy A. A Distributed Resource Management Architecture that Supports Advance Reservations and Co-Allocation. Intl Workshop on Quality of Service, 1999.

[7] Hayek F. A., Bartley W., Klein P., Caldwell B., The collected works of F. A. Hayek. University of Chicago Press, 1989.

[8] Global Grid Forum (2005), http://www.ggf.org/

[9] Globus Toolkit version 4.0, http://www.globus.org

[10] Joita L., Pahwa J. S., Burnap P., Gray A., Rana O., and Miles J. Supporting Collaborative Virtual Organisations in the Construction Industry via the Grid. Proceedings of the UK e-Science All Hands Meeting 2004, 31st Aug.-3rd Sept. 2004 Nottingham, UK.

[11] Joseph J:, Ernest M., Fellenstein C. Evolution of grid computing architecture and grid adoption models. IBM Systems Journal, Volume 43 , Issue 4 (January 2004)

[12] Traversat, Abdelaziz, and Pouyoul, Project JXTA: Loosely-Consistent DHT Rendezvous Walker. Sun Microsystems, Inc., http://www.jxta.org/project/www/docs/jxtadht.

[13] Web Services Agreement Specification (WS-Agreement), 28 June 2005 https://forge.gridforum.org/docman2/ViewCategory.php?group_id=71&category_id=659

[14] WS-Resource Framework, http://www.globus.org/wsrf/