A Recursive Architecture for Hierarchical Grid Resource Management [⋆]

Eduardo Huedo [*], Rubén S. Montero, Ignacio M. Llorente

Departamento de Arquitectura de Computadores y Automática, Facultad de Informática, Universidad Complutense, 28040 Madrid, Spain.

Abstract

Grid resource management has been traditionally limited to just two levels of hierarchy, namely local resource managers and metaschedulers. This results in a non manageable, and thus not scalable, architecture, where each metascheduler has to be able to access thousands of resources, which also implies having a detailed knowledge about their interfaces and configuration. This paper presents a recursive architecture allowing an arbitrary number of levels in the hierarchy. This way, resources can be arranged in different ways, for example, following organizational boundaries or aggregating them by similarity, while hiding the access details. An implementation of this architecture is shown, as well as its benefits in terms of autonomy, scalability, deployment and security. The proposed implementation is based on existing interfaces, allowing for standardization.

Key words: Distributed architectures, System Management, Standards

1 Introduction

The high expectations raised by grid computing have favored the development and deployment of a growing number grid infrastructures, like EGEE [1],

[⋆] This research was supported by Consejería de Educación of Comunidad de Madrid, Fondo Europeo de Desarrollo Regional (FEDER) and Fondo Social Europeo (FSE), through BioGridNet Research Program S-0505/TIC/000101, by Ministerio de Educación y Ciencia, through the research grant TIN2006-02806, and by European Union, through BEinGRID project EU contract IST-2005-034702

[*] Corresponding author. Tel: +34 91 394 76 03. Fax: +34 91 394 75 27.
Email address: `ehuedo@fdi.ucm.es` (Eduardo Huedo).

[1] http://www.eu-egee.org

TeraGrid[2] or OSG[3]. However, interoperability between these grids is still limited, so reducing the potential large-scale application of grid technology, in spite of efforts made by grid community, mainly within the Open Grid Forum[4]. Current approaches to federate different grid infrastructures usually have a single layer of (one or more) metaschedulers or grid workload managers with *plain* access to several underlying grid infrastructures. To achieve interoperation when no common or standard interfaces are available, each of these metaschedulers use the appropriate adapter to interface each different middleware stack [1].

Nowadays, grid technology is mainly applied to multiple research fields with unquestionable success. But, same way that happened with the Internet, grid technology will be key for business development, giving birth to numerous business models, like IT outsourcing, on-demand and utility computing, application service providers, and so on. However, grid computing still does not meet the requirements for enterprise adoption, like complexity hiding, service orientation, transparent access and stability [2], mainly due to the architecture currently used for grids.

The grid architecture presented in this work uses different layers of metaschedulers arranged in a *hierarchical* structure. Moreover, each target grid is handled as another resource, in a *recursive* way, using the same interfaces for resource management. This architecture allows a straightforward federation of grid infrastructures, observing organizational boundaries, as well as an easier interoperation of grid middlewares, and provides better control over shared, but still owned, resources. Therefore, it will encourage research and business organizations to start sharing resources with partners, and will foster the growth of a healthy computing market of resource and service providers and consumers [3].

The rest of the paper is organized as follows. After discussing some related work in Section 2, Section 3 introduces the recursive architecture for hierarchical resource management in grids. To demonstrate its viability and suitability, Section 4, 5 and 6 present an implementation, some use cases of the proposed technology, and some experiences and results, respectively. Finally, Section 7 ends up with some conclusions and hints about future work.

[2] http://www.teragrid.org
[3] http://www.opensciencegrid.org
[4] http://forge.ogf.org/projects/gin

2 Related Work

There are several efforts to classify grid resource management systems, showing the different alternatives typically followed [4,5]. For example, hierarchical approaches have been widely used for information [6] and data management [7] services, but their application for resource management services (i.e. brokering, scheduling, execution and so on) have been limited to just two levels: the scheduler or LRMS, and the metascheduler, or resource broker.

A recursive architecture for grid resource management has been previously applied to federate LCG and GridX1 infrastructures [8], where a GridX1 user interface is hosted in a LCG computing element. However, this solution imposes software, middleware and network requirements on worker nodes. The Globus project is also interested on this kind of recursive architectures, and is working on Bouncer [9], which is a Globus job forwarder initially conceived for federating TeraGrid and Open Science Grid infrastructures.

Based on a study of Internet and similar network-based systems, some authors have recently proposed a three-level hierarchical architecture for the InterGrid [10] (as an analogy with the Internet), enabling peering of grids and exchange of resources. It has the InterGrid Gateways (IGGs) on top, coordinating between the different grids, followed by the IntraGrid Resource Managers (IRM), taking care of resource allocation, using the resource shares assigned by Resource Providers (RP), on the bottom.

Previously, Condor's Flocking and GlideIn mechanisms [11] allowed job transfers across Condor pools' boundaries or the deployment of remote Condor daemons, respectively. Nevertheless, these solutions were not based on standards and require the same resource manager to be installed on all resources. In this sense, the Open Grid Forum's Grid Scheduling Architecture research group (GSA-RG) [5] is working on a standard architecture for the interaction between different metaschedulers.

Other non-hierarchical approaches have been proposed. For example, peer-to-peer (P2P) techniques have been widely used for resource discovery [12] and job scheduling. For example, the DIANA scheduling system is implemented as a P2P system [13], making use of a P2P network to track the available resources on the Grid. However, these techniques are valid only when dealing with similar peers. For example, the infrastructure of an international partner grid and that of a SME are not at the same level, so they can not be considered peers. Nevertheless, they can serve to join desktop PCs, different sites or, as in the InterGrid case, separate grids.

[5] http://forge.ogf.org/projects/gsa-rg

3 Recursive Architecture

The Globus Toolkit[6] has become a *de facto* standard in Grid computing. Globus services allow secure and transparent access to resources across multiple administrative domains [14], and serve as building blocks to implement the stages of Grid scheduling. Resource management is maybe the most important component for computational grids, although it could be also extended to other non-computational resources. The Globus layer provides a uniform interface to many different Distributed Resource Management (DRM) systems, allowing the development of grid workload managers that optimize the use of the underlying computing platforms.

Following an hourglass model, grid workload managers or metaschedulers should have access to a wide range of resources provided through a limited, standardized set of protocols and interfaces. The Globus core grid middleware provides this set. Just as in the Internet, the protocols and interfaces are provided through IP. The main innovation of our architecture is the use of Globus Toolkit services to recursively interface to the resource management services available in a whole Globus-based grid. This allow us to create the required virtualization technology in order to provide a powerful abstraction of the underlying grid resource management services.

It is sometimes difficult to expose all the functionality of a metascheduler just using Globus interfaces, but the functionality usually needed is rather simple, since complex features (like workflow management or parametric execution) are handled in client tools. This is in line with *end-to-end arguments*, which suggest that functions placed at low levels of a system may be redundant or of little value when compared with the cost of providing them at that low level, and that low-level mechanisms to support these functions are justified only as performance enhancements [15]. For example, the arguments that are used in support of RISC architecture, encryption, and operating system kernels are similar to end-to-end arguments. But the best-known example is the architecture of the Internet [16], that fostered the spectacular development of Web and communication technologies in the past decade.

As a performance enhancement, thus compatible with *end-to-end arguments*, it is possible to access local resources (i.e. those located in the same administration domain as the metascheduler) using other interfaces with more functionality and less overheads, for example a simple remote shell (SSH preferably) or DRMAA [17]. For example, GridWay provides SSH MADs to access local resources in an opportunistic way and, in the future, it will provide DRMAA MADs to interact directly with LRMS (Local Resource Management

[6] http://www.globus.org

Systems).

Besides the already discussed benefits, the recursive, hierarchical approach for grid resource management presents additional advantages, in terms of [18]:

- **Decentralized control**: the operation inside each level of the hierarchy is autonomous, with different policies for user access control, identity mapping, scheduling (e.g. assigning different priorities for local and external users), resource sharing (e.g. when to accept, suspend or cancel external jobs to preserve the performance of local ones [19]), accounting, billing, etc.
- **Standardization and interoperability**: the use of, *de-facto* standard, Globus interfaces (WSRF, GridFTP, GSI...) provides innumerable benefits, since it provides an environment in which final users, ISVs and technology providers can undertake investments with greater confidence. Also, there is no need to deploy new services, easing the configuration and keeping well-known firewall settings.
- **Access transparency**: the access to another infrastructure is performed using the same interfaces for resource management. Therefore, there is no need to change existing tools, like workflow engines, parametric tools or LRMS.
- **Scalability**: there are less entities at each level, therefore processes like information dissemination or scheduling perform better and take advantage of spatial locality. For example, external resources should be used only when local ones are overloaded.
- **Reconfigurability**: this solution allows a gradual migration, from fully in-house provision to fully outsourced. This will help to deal with the obstacles for adoption such as enterprise skepticism and IT staff and management resistance.
- **Security**: configuration details can be hidden and only one machine should be accessible from the Internet with a limited set of services, so drastically reducing firewall requirements.

However, this architecture shows higher overheads, which can be reduced by means of file and information caches, using performance models to summarize the dynamic monitoring information about resource availability [20], and taking care of the nature of jobs when deciding whether to forward or execute them locally.

4 Implementation Details

The *Grid Resource Allocation and Management* (GRAM) service is the core of the resource management pillar of the Globus Toolkit. GRAM operates in conjunction with a number of schedulers, including Condor, PBS, SGE and a simple "fork" scheduler. GRAM provides a plugin architecture for extensibility. When the Job Manager is invoked by the Managed Executable Job service to process a job request, it maps the request to a local scheduler using the appropriate plugin. Each plugin provides a set of programs and scripts that map job requests to specific scheduler commands such as submit, poll or cancel. Latest version of GRAM (WS-GRAM or GRAM4) is based on the WSRF (Web Services Resource Framework) specification [7].

GridWay [8] is an open source metascheduling technology that provides a decentralized and modular architecture for resource brokering and workload management, in dynamic and *loosely-coupled* Grid environments [21,22]. GridWay provides an environment for submitting, monitoring, synchronizing and controlling jobs, which is very similar to that found on typical LRMS, in fact, GridWay implements the DRMAA standard [23].

A GridGateWay offers the possibility of encapsulating a virtualized grid inside GRAM, using GridWay as the underlying LRMS, as shown in Figure 1. To interface GridWay through GRAM, a new scheduler adapter has been developed along with a scheduler event generator. Also, a scheduler information provider has been developed in order to feed MDS (Monitoring and Discovery Service) with scheduling information. For file transfer, both GridFTP and RFT can be used.

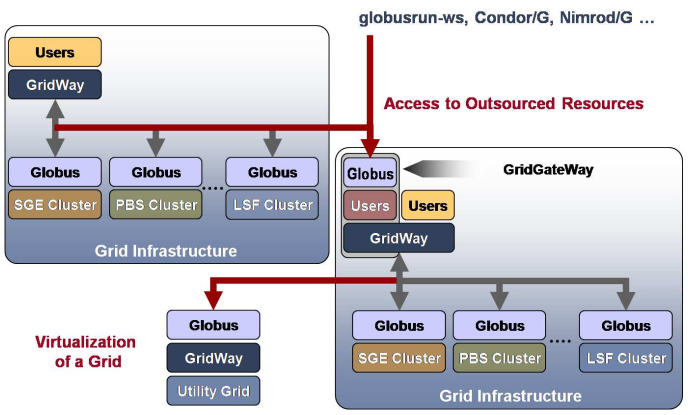

Fig. 1. GridGateWay solution based on the Globus Toolkit and the GridWay Metascheduler.

This way, a GridGateWay provides the standard functionality required to implement a gateway to a enterprise, partner or outsourced grid. It is worth noting that any other workload management tool could be used for the GridGateWay, but the fact that GridWay provides a LRMS-like interface makes easier its integration with GRAM.

[7] http://www.oasis-open.org/committees/wsrf/

[8] http://www.gridway.org

5 Use Cases

Next sections provide use cases of the proposed GridGateWay technology, from a simple WSRF interface for GridWay, through grid federation and transparent access from a cluster, to a Utility Computing service.

5.1 A WSRF Interface for GridWay

The GridGateWay enables the remote access to GridWay's metascheduling capabilities through a WSRF interface. Not all the functionality is accessible this way (e.g. workflows are not available), but in turn it provides an interface that can be used with Globus clients (e.g. globusrun-ws), is accessible through Globus APIs (Globus Java Core, Java CoG) and also can be put to work with the various workflow tools based on Globus, like, for example, the one offered by Java CoG (Karajan).

For example, AstroGrid-D [9], which is a community grid within the German D-Grid initiative [10], uses GridGateWay technology for job submission from different Globus clients to a central GridWay instance [24], as shown in Figure 2.

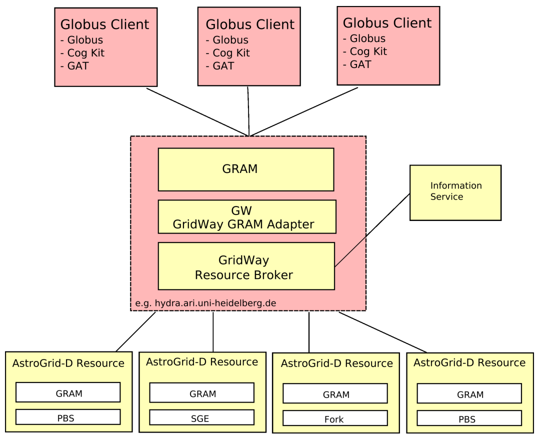

Fig. 2. Scheme of job submission in AstroGrid-D using GridGateWay technology [24].

5.2 Federation of Grid Infrastructures

[9] http://www.gac-grid.org
[10] http://www.dgrid.de

A GridGateWay can serve as an entry point to a grid infrastructure. For example, GRIDIMadrid [11] uses GridGateWay technology to access the EGEE [12] infrastructure on demand. Figure 3 shows how one instance of the GridWay metascheduler manages resources of the GRIDIMadrid regional grid and outsources jobs to a GridGateWay that virtualizes the EGEE international grid. To have a fully federated infrastructure, it would be possible that EGEE resource brokers also use a GridGateWay to send jobs to GRIDIMadrid, avoiding cycles. Section 6 provides some results related to this.

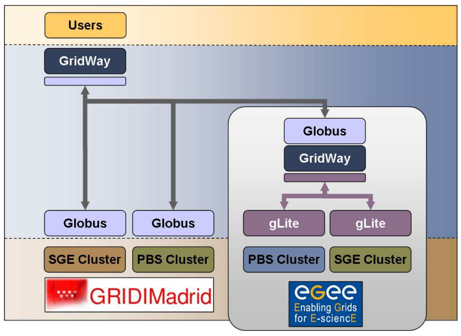

Fig. 3. Federation of GRIDIMadrid and EGEE infrastructures.

5.3 From Cluster to Grid Computing

Most local resource management systems, like Sun Grid Engine (SGE), PBS or Condor, provide a way to interact with other resource managers and, in particular, with GRAM. For example, Figure 4 shows a SGE domain with the usual local queues and a special grid queue (that can be configured to be available only under certain special conditions), that enables SGE to submit jobs to a GRAM service or to GridWay. The GRAM service can, in turn, encapsulate a GridWay Metascheduler (thus conforming a GridGateWay) that gives transparent access to another grid infrastructure [25].

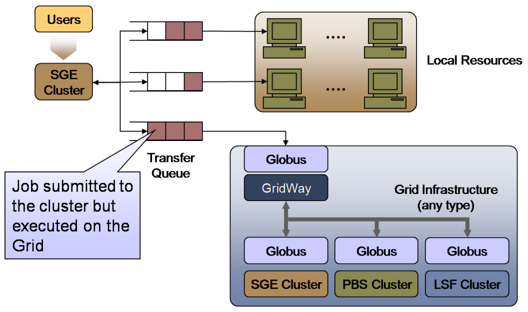

Fig. 4. SGE transfer queue to Globus in order to access a GridGateWay.

[11] http://www.gridimadrid.org
[12] http://www.eu-egee.org

The deployment of a utility computing solution involves a full separation between the provider and the consumer. The consumer requires a uniform, secure and reliable functionality to access the utility computing service and the provider requires a scalable, flexible and adaptive infrastructure to provide the service. The solution should be based on standards and allow a gradual deployment in order to obtain a favorable response from the application developers and the information technology staff [3].

The proposed architecture overcomes such challenges by means of its standard functionality for flexible integration of diverse distributed resources. The GridGateWay acts as a utility computing service, providing a uniform standard interface based on Globus protocols and services for the secure and reliable submission and control of jobs, including file staging, on grid resources. For example, the use of GridGateWays is being evaluated in the BEinGRID [13] EU project in the context of the Business Experiment #14 for on-demand provision of resources.

6 Experiences and Results

In order to illustrate the overheads imposed by the recursive architecture, Figure 5 shows the throughput achieved at UCM when accessing resources from EGEE both directly and through a GridGateWay. We used a application composed of 100 tasks, each taking about 10 seconds to execute. As expected, there are differences in latency (response time) and throughput in each case, but the use of a GridGateWay supposes a performance loss of only 10.85%. Notice that this performance loss has been obtained with a small application requiring a few seconds to execute. Since the overheads are independent on the computational time required by the application, they will suppose a smaller fraction of the total time if more demanding applications were used.

Related to the use case presented in Section 5.2, Figure 6 shows the throughput achieved at UCM when simultaneously accessing local resources and a GridGateWay to the EGEE infrastructure, using the same application. Notice that, in this case, the whole EGEE infrastructure is accessed as a single GRAM resource, allowing the definition of compulsory grid-wide policies for external users as long as the GridGateWay is the only entry point.

[13] http://www.beingrid.eu

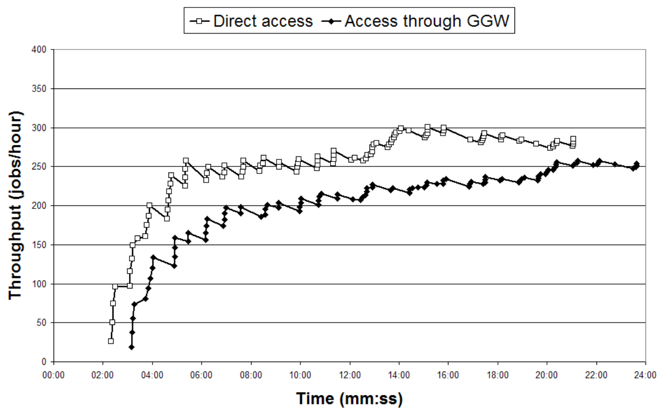

Fig. 5. Throughput achieved at UCM when accessing resources from EGEE both directly and through a GridGateWay.

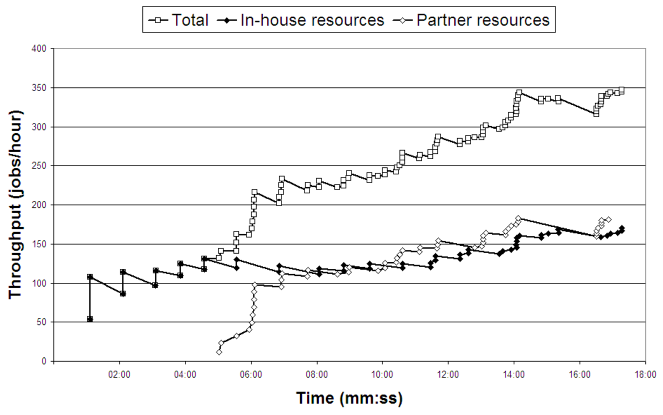

Fig. 6. Throughput achieved at UCM when simultaneously accessing local resources and a GridGateWay giving access to the EGEE infrastructure.

7 Conclusions

We have shown the potential benefits of a recursive architecture for hierarchical grid resource management. To better illustrate them, we have presented some implementation details, real use cases and experiences. The main strength of the proposed approach is the use of Globus interfaces for grid infrastructure access and federation. This makes existing tools ready to be used without modifications and enormously simplifies interoperation.

There are a lot of lines for future research and work, including the investigation of resource sharing policies, the use of simulation tools to evaluate complex scenarios, policies and techniques, the development of mechanisms for authorization and identity mapping, or the definition accounting and billing methods to implement utility computing provision models.

Acknowledgements

This work makes use of results produced by the Enabling Grids for E-sciencE project, a project co-funded by the European Commission (under contract number INFSO-RI-031688) through the Sixth Framework Programme. EGEE brings together 91 partners in 32 countries to provide a seamless Grid infrastructure available to the European research community 24 hours a day. Full information is available at http://www.eu-egee.org.

References

[1] E. Huedo, R. S. Montero, I. M. Llorente, A Modular Meta-Scheduling Architecture for Interfacing with Pre-WS and WS Grid Resource Management Services, Future Generation Computer Systems 23 (2) (2007) 252–261.

[2] H. Stockinger, Grid Computing: A Critical Discussion on Business Applicability, IEEE Distributed Systems Online 7 (6).

[3] I. M. Llorente, R. S. Montero, E. Huedo, K. Leal, A Grid Infrastructure for Utility Computing, in: Proc. 15th IEEE Intl. Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, 2006, pp. 163–168.

[4] K. Krauter, R. Buyya, M. Maheswaran, A Taxonomy and Survey of Grid Resource Management Systems for Distributed Computing, Software - Practice and Experience 32 (2) (2002) 135–164.

[5] A. Kertsz, P. Kacsuk, A Taxonomy of Grid Resource Brokers, in: Proc. 6th Austrian-Hungarian Workshop on Distributed and Parallel Systems, 2007, pp. 201–210.

[6] C. Mastroianni, D. Talia, O. Verta, Evaluating Resource Discovery Protocols for Hierarchical and Super-Peer Grid Information Systems, in: 15th Euromicro Intl. Conf. Parallel, Distributed and Network-based Processing, 2007, pp. 147–154.

[7] A. Finkelstein, C. Gryce, J. Lewis-Bowen, Relating Requirements and Architectures: A Study of Data-Grids, J. Grid Computing 2 (3) (2004) 207–222.

[8] A. Agarwal, M. Ahmed, A. Berman, et al., GridX1: A Canadian Computational Grid, Future Generation Computer Systems 23 (5) (2007) 680–687.

[9] C. Baumbauer, S. Goasguen, S. Martin, Bouncer: A Globus Job Forwarder, in: Proc. 1st TeraGrid Conf., 2006.

[10] M. D. de Assunção, R. Buyya, S. Venugopal, InterGrid: A Case for Internetworking Islands of Grids, Concurrency and Computation: Practice and Experience. In press.

[11] J. Frey, T. Tannenbaum, M. Livny, I. Foster, S. Tuecke, Condor-G: A Computation Management Agent for Multi-Institutional Grids, Cluster Computing 5 (3) (2002) 237–246.

[12] P. Trunfio, D. Talia, H. Papadakis, et al., Peer-to-Peer Resource Discovery in Grids: Models and Systems, Future Generation Computer Systems 23 (7) (2007) 864–878.

[13] R. McClatchey, A. Anjum, H. Stockinger, A. Ali, I. Willers, M. Thomas, Data Intensive and Network Aware (DIANA) Grid Scheduling, J. Grid Computing 5 (1) (2007) 43–64.

[14] I. Foster, Globus Toolkit Version 4: Software for Service-Oriented Systems, in: Proc. IFIP Intl. Conf. Network and Parallel Computing, 2005, pp. 2–13.

[15] J. H. Saltzer, D. P. Reed, D. D. Clark, End-to-End Arguments in System Design, ACM Trans. Computer Systems 2 (4) (1984) 277–288.

[16] E. B. Carpenter, Architectural Principles of the Internet, RFC 1958 (Jun. 1996).

[17] H. Rajic, et al., Distributed Resource Management Application API Specification 1.0, Document GFD-R.22, DRMAA Working Group – Open Grid Forum (2004, updated 2007).

[18] H. Stockinger, Defining the Grid: A Snapshot on the Current View, J. Supercomputing 42 (1) (2007) 3–17.

[19] O. S. José, L. M. Suárez, E. Huedo, R. S. Montero, I. M. Llorente, Resource Performance Management on Computational Grids, in: Proc. 2nd Intl. Symp. Parallel and Distributed Computing, 2003, pp. 215–221.

[20] C. Vázquez, E. Huedo, R. S. Montero, I. M. Llorente, A Performance Model for Federated Grid Infrastructures, in: 16th Euromicro Intl. Conf. Parallel, Distributed and Network-based Processing, 2008, pp. 188–192.

[21] E. Huedo, R. S. Montero, I. M. Llorente, A Framework for Adaptive Execution on Grids, Software - Practice and Experience 34 (7) (2004) 631–651.

[22] I. M. Llorente, R. S. Montero, E. Huedo, A Loosely Coupled Vision for Computational Grids, IEEE Distributed Systems Online 6 (5).

[23] J. Herrera, E. Huedo, R. S. Montero, I. M. Llorente, GridWay DRMAA 1.0 Implementation - Experience Report, Document GFD-E.104, DRMAA Working Group – Open Grid Forum (2007).

[24] R. Spurzem, P. Berczik, I. Berentzen, et al., From Newton to Einstein: N-body Dynamics in Galactic Nuclei and SPH Using New Special Hardware and AstroGrid-D, J. Physics: Conference Series 78 (2007) 012071 (6pp).

[25] C. Vázquez, J. Fontán, E. Huedo, R. S. Montero, I. M. Llorente, Transparent Access to Grid-Based Compute Utilities, in: Proc. 7th Intl. Conf. Parallel Processing and Applied Mathematics, 2008, in press.