Evaluation of a Utility Computing Model Based on the Federation of Grid Infrastructures *

Tino Vázquez, Eduardo Huedo, Rubén S. Montero, and Ignacio M. Llorente

Departamento de Arquitectura de Computadores y Automática
Facultad de Informática
Universidad Complutense de Madrid, Spain
{tinova, ehuedo}@fdi.ucm.es, {rubensm, llorente}@dacya.ucm.es

Abstract. Utility computing is a service provisioning model which will provide adaptive, flexible and simple access to computing resources, enabling a pay-per-use model for computing similar to traditional utilities such as water, gas or electricity. On the other hand, grid technology provides standard functionality for flexible integration of diverse distributed resources. This paper describes and evaluates an innovative solution for utility computing, based on grid federation, which can be easily deployed on any infrastructure based on the Globus Toolkit. This solution exhibits many advantages in terms of security, scalability and site autonomy, and achieves good performance, as shown by results, mainly with compute-intensive applications.

1 Introduction

Utility is a computing term related to a new paradigm for an information technology (IT) provision which exhibits several potential benefits for an organization [1]: reducing fixed costs, treating IT as a variable cost, providing access to unlimited computational capacity and improving flexibility, thereby making resource provision more agile and adaptive. Such valuable benefits may bring the current fixed-pricing policy of IT provision to an end, where computing is carried out within individual corporations or outsourced to external service providers [2].

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. Moreover, the solution should be based on standards and allow a gradual deployment in order to obtain a favorable response from application developers and IT staff.

* 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 research grant TIN2006-02806, and by European Union, through BEinGRID project EU contract IST-2005-034702.

In a previous work [3], we have proposed a solution for federating grids which can be deployed on a grid infrastructure based on the Globus Toolkit (GT). Such solutions demonstrates that grid technology overcomes utility computing challenges by means of its standard functionality for flexible integration of diverse distributed resources.

A similar approach for the federation of grid infrastructures has been previously applied to meet LCG and GridX1 infrastructures [4], hosting a GridX1 user interface in a LCG computing element. However, this solution imposes software, middleware and network requirements on worker nodes. The Globus project is also interested in this kind of *recursive* architectures, and is working on Bouncer [5], which is a Globus job forwarder initially conceived for federating TeraGrid and Open Science Grid infrastructures. There are other approaches to achieve middleware interoperability, for example between UNICORE and Globus [6] and between gLite and UNICORE [7].

On the other hand, MOAB Grid Suite from Cluster Resources[1] is a grid workload management solution that integrates scheduling, management, monitoring and reporting of workloads across independent clusters. Moreover, Condor's Flocking and GlideIn mechanisms [8] provide similar functionality, allowing job transfers across Condor pools' boundaries or the deployment of remote Condor daemons, respectively. Nevertheless, these solutions are not based on standards and require the same workload manager to be installed in all resources. In this sense, the Open Grid Forum's Grid Scheduling Architecture research group[2] is working on an standard architecture for the interaction between different metaschedulers.

Finally, other projects, like Gridbus [9] or GRIA [10], are developing components for accounting, negotiation and billing, in order to provide end-to-end quality of service driven by computational economy principles.

In this work, we use a technology for the federation of grid infrastructures to build a utility model for computing, which provides full metascheduling functionality, and it is flexible, scalable and based on standards. The rest of this paper is as follows. Section 2 presents a solution for building utility grid infrastructures based on the Globus Toolkit and the GridWay Metascheduler by means of GridGateWays. The architecture of a GridGateWay is described and evaluated in Section 3, while Section 4 shows some experimental results. Finally, Section 5 presents some conclusions and our plans for future work.

2 Utility Grid Infrastructures

A grid infrastructure offers a common layer to integrate non-interoperable computational platforms by defining a consistent set of abstraction and interfaces for access to, and management of, shared resources [11]. Most current grid infrastructures are based on the Globus Toolkit [12], that implements a collection of high level services at the grid infrastructure layer. These services include, among

[1] http://www.clusterresources.com
[2] http://forge.gridforum.org/projects/gsa-rg

others, resource monitoring and discovery (MDS), resource allocation and management (GRAM), file transfer (RFT) and a security infrastructure (GSI). The Globus layer provides a uniform interface to many different DRM (Distributed Resource Manager) systems, allowing the development of grid workload managers that optimize the use of the underlying computing platforms.

It has been predicted [13] that *outsourced grids*, managed by dedicated service providers, will supply resources on demand over the Internet. Different studies suggest that growing network capacity will allow businesses and consumers to draw their computing resources from outsourced grids apart from enterprise grids.

The technological feasibility of the utility model for computing services is established by using a novel grid infrastructure based on Globus Toolkit components and the GridWay Metascheduler[3] [14]. It is well known that Globus Toolkit services provide a uniform, secure and reliable interface to heterogeneous computing platforms managed by different DRM systems. The main innovation of our model is the use of Globus Toolkit services to recursively interface to the services available in a federated Globus based grid.

A set of Globus Toolkit services hosting a GridWay Metascheduler, what we call a *GridGateWay*, provides the standard functionality required to implement a gateway to a federated grid. Such a combination allows the required virtualization technology to be created in order to provide a powerful abstraction of the underlying grid resource management services and meets the requirements imposed by a utility computing solution [3]. The GridGateWay acts as the utility computing service, providing a uniform standard interface based on Globus interfaces, protocols and services for the secure and reliable submission and control of jobs, including file staging, on grid resources.

Application developers, portal builders, and ISVs may interface with the utility computing service by using the Distributed Resource Management Application API (DRMAA)[4]. DRMAA is an Open Grid Forum (OGF)[5] standard that constitutes a homogeneous interface to different DRM systems to handle job submission, monitoring and control, and retrieval of finished job status.

The grid hierarchy in our utility computing model is clear. An enterprise grid, managed by the IT Department, includes a GridGateWay to an outsourced grid, managed by the utility computing service provider. The outsourced grid provides pay-per-use computational power when local resources are overloaded. This hierarchical grid organization may be extended recursively to federate a higher number of partner or outsourced grid infrastructures with consumer/provider relationships. Figure 1 shows one of the many grid infrastructure hierarchies that can be deployed with GridGateWay components. This hierarchical solution, which resembles the architecture of the Internet (characterized by the end-to-end argument [15] and the IP hourglass model [16]), involves some performance overheads, mainly higher latencies, which will be quantified in Section 4.

---

[3] http://www.gridway.org
[4] http://www.drmaa.org
[5] http://www.ogf.org

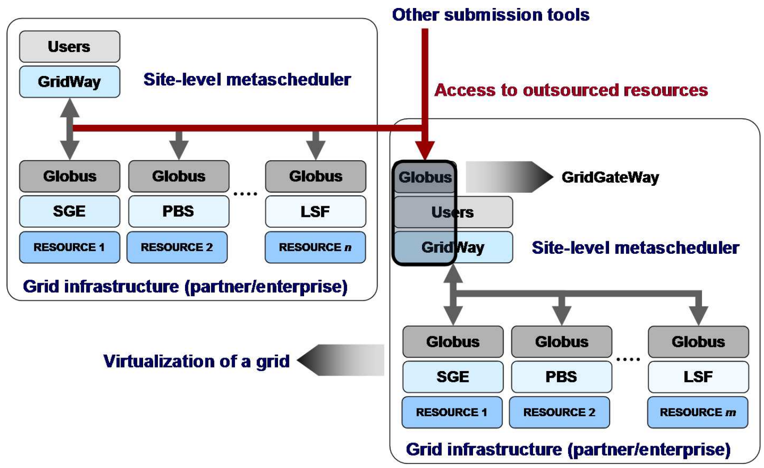

Fig. 1. Utility computing solution based on the Globus Toolkit and the GridWay Metascheduler.

The access to resources, including user authentication, across grid boundaries is under control of the GridGateWay service and is transparent to end users. In fact, different policies for job transfer and load balancing can be defined in the GridGateWay. The user and resource accounting and management could be performed at different aggregation levels in each infrastructure.

A grid involves standardization, so utility solutions will spread provided that grid technology is available. On the other hand, the cultural and business model changes required for adopting the utility model should be gradual, starting with access to a local workload manager, followed by an in-house enterprise grid and finally moving onto outsourced services. These adoption steps are transparent to the end user in the proposed solution, since he always interfaces with a given GridGateWay using DRMAA standard.

3 Architecture of the GridGateWay

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 with scheduling information. Therefore, the main functionality of the GridGateWay is provided by the GridWay Metascheduler, although accessed through the standard interfaces provided by the Globus Toolkit. Moreover, the GridGateWay takes advantage of recent improvements in the GridWay Metascheduler, like multiple user support, accounting, client and resource fault-tolerance, and new drivers to access different grid services. It is expected that new functionality related to the GridGateWay will be added to GridWay, like new scheduling policies for enterprise, partner and utility grid infrastructures, security and certificate management policies, billing, etc.

The presented architecture has a number of advantages in terms of security and scalability. Regarding security, only the GridGateWay should be accessible from the Internet, and only Globus firewall requirements are imposed. Moreover, it is possible to restrict the dissemination of system configuration outside site boundaries, and resource access and usage is controlled and accounted in the GridGateWay. Finally, different certificate mapping policies can be defined at each level, and the set of recognized Certificate Authorities (CA) can also be different. Regarding scalability, with this architecture the scheduling process is divided into different levels, where a different scheduling policy can be applied. Also, there is no need to disseminate everywhere all system monitoring information. For example, resource information can be aggregated and then published by means of the r_∞ and $n_{1/2}$ parameters, as proposed previously by the authors [17].

4 Experiments

4.1 Measurement of GridGateWay Overheads

In order to measure the overheads imposed by the GridGateWay architecture, we set up a simple infrastructure where a client run an instance of the GridWay Metascheduler interfacing with a GridGateWay (running another instance of the GridWay Metascheduler accessed through Globus Toolkit services) which, in turns, interfaces with a Globus resource managed by the Fork job manager. The application used was a simple `/bin/echo` (using the executable in the remote resource), so the required computational time was negligible and the file transfer costs were minimum (basically the standard input/output/error streams).

The average Globus overhead in both the GridGateWay and the resource was 6 seconds. The average scheduling overhead in the client machine was 15 seconds, since the scheduling interval was set to 30 seconds, and in the GridGateWay is only 5, since the scheduling interval was set to 10 seconds in order to improve the response time. Therefore, the difference in execution time between a direct execution and an execution through a GridGateWay can be as low as 11 seconds. This difference could be higher when more file transfers are involved.

4.2 Enterprise Grid Resource Provision from a Partner Grid through a GridGateWay

Now, in order to evaluate the behavior of the proposed solution, we set up a more realistic infrastructure where a client run an instance of the GridWay Metascheduler interfacing local resources in an enterprise (UCM, in this case), based on GT4 Web Services interfaces, and a GridGateWay that gives access to resources from a partner grid (*fusion* VO of EGEE, in this case), based on GT pre-Web Services interfaces. The simultaneous use of different MADs to access multiple partner grid infrastructures has been demonstrated before [18, 19]. In fact, that could be an alternative for the coexistence of different grid infrastructures, although based on distinct middleware (GT2, GT4, LCG, gLite...).

In this configuration, draco is the client machine, providing access to the enterprise grid, and cepheus is the GridGateWay, providing access to the partner grid (see Figure 2). Notice that, in this case, the GridGateWay is hosted in the enterprise. However, in a typical business situation it would be hosted in the partner infrastructure. The characteristics of resources from UCM are shown in Figure 3, as provided by GridWay command `gwhosts`, while those from EGEE (*fusion* VO) are shown in Figure 4.

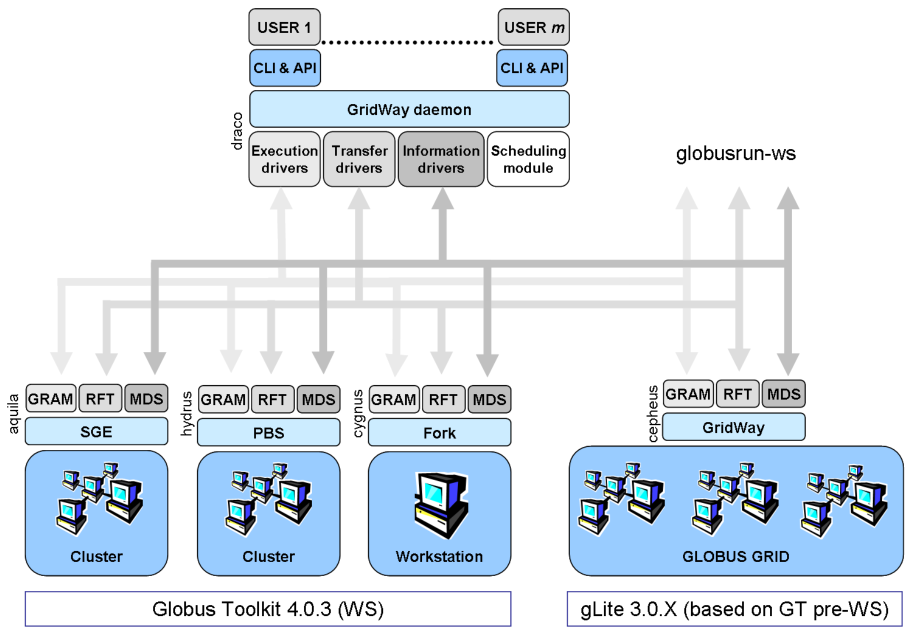

Fig. 2. Experimental environment.

Notice also that, with the GridGateWay architecture, it is possible to access resources with non WS (Web Services) interfaces, like those present in the LCG computing element of gLite 3, using WS interfaces, like those present in the new services provided by GT4, which are based on the Web Services Resource Framework (WSRF). Therefore, the interface the client sees is more homogeneous. Moreover, the access pattern of EGEE resources requires the client to open a GridFTP or GASS server. Nevertheless, GridWay doesn't need such server to access WS-based resources. With the proposed architecture, there is a GridFTP server already started in the GridGateWay, so there is no need to open incoming ports in the client, and the firewall requirements results solely in allowing outgoing ports. This is very important when the access to grid re-

```
ehuedo@draco:~$ gwhost
HID OS              ARCH    MHZ %CPU  MEM(F/T)     DISK(F/T)     N(U/F/T) LRMS                    HOSTNAME
0   Linux2.6.16-2-6 x86    3216    0  831/2027 114644/118812        0/0/1 Fork                    cygnus.dacya.ucm.es
1   Linux2.6.16-2-a x86_6 2211   100  671/1003   76882/77844        0/2/2 SGE                     aquila.dacya.ucm.es
2   Linux2.6.16-2-6 x86    3215    0  153/2027 114541/118812        0/0/1 Fork                    draco.dacya.ucm.es
3   Linux2.6.16.13- x86    3200  200    10/512 148855/159263        0/0/2 SGE                     ursa.dacya.ucm.es
4   Linux2.6.16-2-a x86_6 2211   100  674/1003   76877/77844        0/2/2 PBS                     hydrus.dacya.ucm.es
5   NULLNULL        NULL      0    0       0/0           0/0   6/665/1355 GW                      cepheus.dacya.ucm.es
```

Fig. 3. Resources in enterprise grid (UCM). Notice that cepheus, acting as a GridGateWay, appears as another resource with GW (GridWay) as LRMS (Local Resource Management System) and provides no information about its configuration, but only about free and total nodes.

```
ehuedo@cepheus:~$ gwhost
HID OS              ARCH    MHZ %CPU  MEM(F/T)     DISK(F/T)     N(U/F/T) LRMS                    HOSTNAME
0   Scientific Linu i686   1001    0   513/513           0/0    3/103/224 jobmanager-lcgpbs       lcg02.ciemat.es
1   ScientificSL3.0 i686    551    0   513/513           0/0       0/3/14 jobmanager-lcgpbs       ce2.egee.cesga.es
2   Scientific Linu i686   1000    0 1536/1536           0/0       0/2/26 jobmanager-lcgpbs       lcgce01.jinr.ru
3   Scientific Linu i686   2800    0 2048/2048           0/0       0/0/98 jobmanager-lcgpbs       lcg06.sinp.msu.ru
4   Scientific Linu i686   1266    0 2048/2048           0/0      0/26/58 jobmanager-pbs          ce1.egee.fr.cgg.com
5   Scientific Linu i686   2800    0 2048/2048           0/0    0/135/206 jobmanager-lcgpbs       node07.datagrid.cea.fr
6   ScientificSL3.0 i686   3000    0 2048/2048           0/0    0/223/352 jobmanager-lcgpbs       fal-pygrid-18.lancs.ac.u
7   Scientific Linu i686   2400    0 1024/1024           0/0    0/139/262 jobmanager-lcgpbs       heplnx201.pp.rl.ac.uk
8   Scientific Linu i686   3000    0 2048/2048           0/0       0/0/60 jobmanager-pbs          cluster.pnpi.nw.ru
9   Scientific Linu i686   1098    0 3000/3000           0/0       0/5/16 jobmanager-lcgpbs       grid002.jet.efda.org
10  Scientific Linu i686   2800    0 1024/1024           0/0      0/30/32 jobmanager-lcgpbs       ce.hep.ntua.gr
11  Scientific Linu i686   2000    0   492/492           0/0        0/2/7 jobmanager-lcgpbs       ce.epcc.ed.ac.uk
```

Fig. 4. Resources in partner grid (*fusion* VO of EGEE). Notice that the access to these resources is configured in cepheus, acting as a GridGateWay.

sources is generalized and performed from ISV applications, using the DRMAA standard.

In the case of EGEE resources, and in order to not saturate the testbed (wich is supposed to be at production level) with our tests, we limited the number of running jobs in the same resource to 10, and the number of running jobs belonging to the same user to 30.

First of all, we want to compare the direct access to EGEE resources and the access through a GridGateWay. In order to do that, in a first experiment we submitted the jobs directly to the GridWay instance running on cepheus. And, in a second experiment, we submitted the jobs to the GridWay instance running on draco with cepheus, acting as a GridGateWay, as the unique resource.

In this case, the application used was the distributed calculation of the π number as $\int_0^1 \frac{4}{1+x^2}dx$. Each task computes the integral in a separate section of the function and all results are finally added to obtain the famous 3.1415926435... Therefore, the required computational time is now higher (about 10 seconds on a 3.20GHz Pentium 4 for each task) and file transfer costs include the executable and the standard input/output/error streams (about 10KB per task).

Figure 5 shows the throughput achieved in EGEE (*fusion* VO) resources when accessed directly and when they are accessed through the GridGateWay. The number of tasks submitted was 100. As expected, there are differences in latency (response time) and throughput when directly accessing the resources

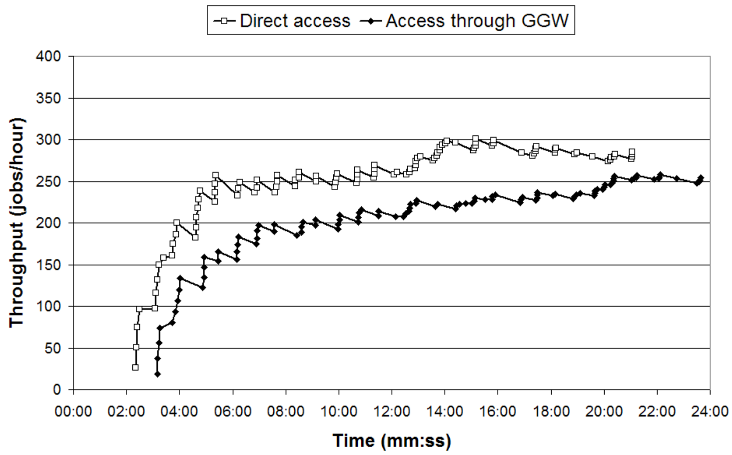

Fig. 5. Throughput achieved in EGEE (*fusion* VO) when accessed directly and when accessed through a GridGateWay.

and when using the GridGateWay. A throughput of 284.8 jobs/hour was achieved with direct access, versus 253.9 jobs/hour with the access through the GridGateWay. Therefore, the use of a GridGateWay supposes a performance loss of only 10.85%. Notice that this performance loss has been obtained with an application requiring only 10 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 when more demanding applications are used.

Figure 6 shows the throughput achieved in UCM when provisioning partner resources from EGEE (*fusion* VO) through a GridGateWay. The number of tasks submitted was 100 again. In this case, there are also differences in latency between in-house and partner resources. Besides network connection and the use of a GridGateWay, it is also due to the production status of partner resources, as they are under heavy usage. The aggregated throughput achieved was 347.5 jobs/hour. In-house and partner resources contributed almost equally (170.3 and 181.4 jobs/hour, respectively) to the aggregated throughput.

5 Conclusions and Future Work

We have presented and evaluated a solution that meets the requirements for utility computing, namely: (i) a uniform, secure and reliable functionality to access the utility; (ii) a scalable, flexible and adaptive infrastructure to provide the service; and (iii) a solution based on standards and gradually deployable. Moreover, the presented performance results are promising.

This innovative utility solution for computing provision, which can be deployed on a grid infrastructure based on existing Globus Toolkit components

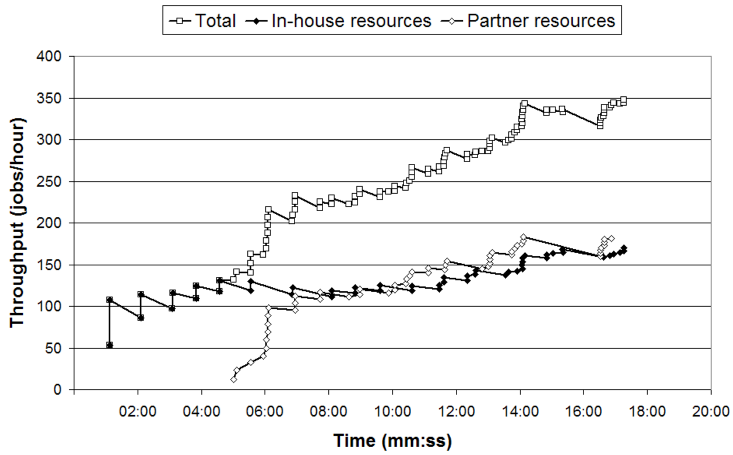

Fig. 6. Throughput achieved in UCM when provisioning resources from EGEE (*fusion* VO) through a GridGateWay.

and related tools, will allow companies and research centers to access their in-house, partner and outsourced computing resources via automated methods using grid standards in a simpler, more flexible and adaptive way. Moreover, the proposed solution has many advantages in terms of security, scalability and site autonomy. Initial results show that the performance loss is low (about 10% for very short tasks), and it would be even lower with applications requiring more computational time.

Future work will include the fine tuning of components to reduce latency and increase throughput, as well as the development of scheduling policies considering these factors, latency and throughput, in order to reduce the total execution time of a whole workload, and also taking into account resource ownership to reduce the associated cost (in terms of real money, resource usage or partner satisfaction). New components for negotiation, service level agreement, credential management, and billing are currently being developed in the context of the Grid4Utility project[6].

Acknowledgments

We want to acknowledge all institutions belonging to the *fusion* VO[7] of the EGEE project for giving us access to their resources.

[6] http://www.grid4utility.org

[7] http://grid.bifi.unizar.es/egee/fusion-vo

References

1. Murch, R.: Introduction to Utility Computing: How It Can Improve TCO. Prentice Hall (2005)
2. Carr, N.G.: The End of Corporate Computing. MIT Sloan Management Review **46**(3) (Spring 2005) 67–73
3. Llorente, I.M., Montero, R.S., Huedo, E., Leal, K.: A Grid Infrastructure for Utility Computing. In: Proc. 15th IEEE Intl. Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE 2006), IEEE CS Press (2006)
4. Walker, R., Vetterli, M., Impey, R., Mateescu, G., Caron, B., Agarwal, A., Dimopoulos, A., Klektau, L., Lindsay, C., Sobie, R., Vanderster, D.: Federating Grids: LCG Meets Canadian HEPGrid. In: Proc. Computing in High Energy and Nuclear Physics (CHEP 2004). (2004)
5. Baumbauer, C., Goasguen, S., Martin, S.: Bouncer: A Globus Job Forwarder. In: Proc. 1st TeraGrid Conf. (TeraGrid 2006). (2006)
6. Breuer, D., Wieder, P., van den Berghe, S., von Laszewski, G., MacLaren, J., Nicole, D., Hoppe, H.C.: A UNICORE-Globus Interoperability Layer. Computing and Informatics **21** (2002) 399–411
7. Mallmann, D., Riedel, M., Twedell, B., Streit, A.: Interoperability Plan for UNICORE. Technical Report MSA3.3, EGEE-II (2006)
8. Frey, J., Tannenbaum, T., Livny, M., Foster, I., Tuecke, S.: Condor-G: A Computation Management Agent for Multi-Institutional Grids. Cluster Computing **5**(3) (2002) 237–246
9. Buyya, R., Venugopal, S.: The Gridbus Toolkit for Service Oriented Grid and Utility Computing: An Overview and Status Report. In: Proc. 1st IEEE Intl. Workshop on Grid Economics and Business Models (GECON 2004), IEEE CS (April 2004) 19–36
10. Surridge, M., Taylor, S., Roure, D.D., Zaluska, E.: Experiences with GRIA Industrial Applications on a Web Services Grid. In: Proc. 1st Intl. Conf. e-Science and Grid Computing (e-Science05), IEEE CS (2005) 98–1005
11. Foster, I., Tuecke, S.: Describing the Elephant: The Different Faces of IT as Service. Enterprise Distributed Computing **3**(6) (July-August 2005)
12. Foster, I.: Globus Toolkit Version 4: Software for Service-Oriented Systems. In: Proc. IFIP Intl. Conf. Network and Parallel Computing (NPC 2005). Volume 3779 of Lecture Notes in Computer Science., Springer-Verlag (2005) 2–13
13. Claunch, C.: Gartner Predicts: The Future of IT. In: Gartner's annual Symposium/ITxpo. (2002)
14. Huedo, E., Montero, R.S., Llorente, I.M.: A Framework for Adaptive Execution on Grids. Software - Practice and Experience **34**(7) (2004) 631–651
15. B. Carpenter, E.: Architectural Principles of the Internet. RFC 1958 (June 1996)
16. Deering, S.: Watching the Waist of the Protocol Hourglass. In: 6th IEEE International Conference on Network Protocols. (1998)
17. Montero, R.S., Huedo, E., Llorente, I.M.: Benchmarking of High Throughput Computing Applications on Grids. Parallel Computing **32**(4) (April 2006) 267–279
18. Vázquez-Poletti, J.L., Huedo, E., Montero, R.S., Llorente, I.M.: Coordinated Harnessing of the IRISGrid and EGEE Testbeds with GridWay. J. Parallel and Distributed Computing **66**(5) (May 2006) 763–771
19. Huedo, E., Montero, R.S., Llorente, I.M.: A Modular Meta-Scheduling Architecture for Interfacing with Pre-WS and WS Grid Resource Management Services. Future Generation Computer Systems **23**(2) (February 2007) 252–261