International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 1– No.1, November 2012 – www.ijais.org

A Novel SLA based Task Scheduling in Grid Environment

S.Selvarani
Associate Professor
Department of Information Technology,
Tamilnadu College of Engineering, Coimbatore,
India

G.Sudha Sadhasivam, PhD.
Professor
Department of Computer Science and Engineering,
PSG College of Technology, Coimbatore, India

ABSTRACT

Service Level Agreements (SLAs) are used in different domains and on different levels to establish agreements on Quality of Service (QoS) between a service provider and a customer. SLAs can be based on general agreements like framework agreements. In a grid environment SLA can be technical agreements which deal with properties of services and resources. Research and development in grid computing started addressing electronic Service Level Agreements where negotiation is usually done automatically by appropriate instances of the service provider and service consumer. Today's internet facility is used to specify SLAs with new improvements which satisfy customer expectations.

The main idea is to provide different levels of service by forging agreements between the user and resource owner. Such agreements are agreed on the basis of different constraints expressed by the user and/or the resource owner. The use of Service Level Agreement (SLA) gives rise to a fundamentally new approach for job scheduling on the grid.

This paper provides an overview of the issues surrounding the design of a fundamentally new infrastructure for job scheduling on the grid, which is based on the notion of SLA's. Such SLA's are negotiated between a client and a provider based on job categorization and resource clustering.

Keywords

Service Level Agreement, Quality of Service, Job Categorization, Resource clustering

1. INTRODUCTION

A common requirement in distributed computing systems such as Grid is to coordinate access to resources located within different administrative domains than the requesting application. The coordination of Grid resources is complicated by the frequently competing needs of the resource consumer or application and the resource owner [8]. The application needs to understand and affect resource behaviour and often demands assurances as to the level and type of service being provided by the resource. The resource owner may want to maintain local control and discretion over how their resources are used.

A common means of reconciling these two competing demands is to establish a procedure by which the parties can negotiate a service level agreement (SLA) that expresses a resource provider contract with a client to provide some measurable capability or to perform a specified task. SLA allows a client to understand what to expect from resources without requiring detailed knowledge neither of competing workloads nor of resource owners policies.

Many Grid Computing scenarios involve jobs with complex workflows which are somehow mapped down onto computational resources; the user is typically expected to have some assurances about when the job will run. Computational resources are typically controlled by batch queuing systems, which don't offer any guarantees or any other information about run times. The exception is when advance reservation is used. Advance reservation adversely affects resource utilization, and therefore the resource owner's income, and so is undesirable. Advance reservation is a restricted level of service.

The main idea is to provide different levels of service by forging agreements between the different parties such as user, resource owner. Such agreements are forged on the basis of different constraints expressed by and agreed between the user and the resource owner and essentially specify a desired and agreed level of service. Use of Service Level Agreements gives rise to a fundamentally new approach for job scheduling on the Grid.

Service Level Agreements are contracts between a service provider and their customers that describe the service, terms, guarantees, responsibilities and level of the service to be provided. They have been widely used by network operators and their customers. They can be used by any service provider for computing, storage, data transport etc. [5]

Programs are used for dynamic negotiation and creation of SLA. An SLA can be negotiated dynamically every time before the service is provided. Services to be negotiated are resource provisioning services; they are required to provide higher level services, such as solving a complex problem and running a given application. These services require the use of various resources like computing nodes, network connections, storage areas or any combination of these. Resource consumption evolves in time and is sometimes dependent on the successful completion of previous tasks. An orchestrator communicates on behalf of customers end users in our use cases with several local resource managers to negotiate and create SLAs dynamically.

This paper proposes a new infrastructure for job scheduling on the Grid, which is based on the notion of Service Level Agreements (SLAs). In the proposed method jobs are categorized based on their QoS parameters and resources are clustered based on the CPU count and MIPs of resources available in the grid and found greater enhancement based on cost and time of job execution. [6,10]

We ask that authors follow some simple guidelines. In essence, we ask you to make your paper look exactly like this document. The easiest way to do this is simply to download the template, and replace the content with your own material.

2. RELATED WORK

An SLA is a bilateral agreement, typically between a service provider and a service consumer. There has been a number of research works in SLA based resource management in grid environment. SLA's were in use since 1960's [9]. In recent years SLA's were widely used for providing network services and outsourcing IT functions. [9].

Shifting of grid towards service-oriented architecture led to the adoption of SLA's for allocating resources. In recent years SLA's were used in grid architecture [6]. Issues related to the usage of SLA's for resource management on the grid have been considered [12].

Ludwig et al [1] proposed technologies for forming SLA's in XML-based representations such as WSLA. The CGF GRAAP working group defined emerging technologies like negotiation protocol WS-Agreement [7].

GRUBER[7] devised a Grid resource SLA broker, in which resource usage policy and resource access policy are separated. GRUBER while performing job scheduling does not take into account the SLAs. Yun Fu et al [13] modelled a dynamic SLA based scheduling algorithm called SLASH which is based on price and penalty. Maui [11] proposed a job scheduler with policy specifications and policy enforcement used in supercomputers and clustered systems.

3. ARCHITECTURE OF IMPROVED SLA BASED JOB SCHEDULING ALGORITHM

The block diagram of the architecture is shown in Figure 1 below [2]. It contains a job categorizer, Resource provider and the SLA based job scheduler. The job categorizer is used to categorize the submitted jobs according to some QoS parameters such as number of CPU's needed to execute that job, security level and required OS. Resource providers will provide the available resources, which are clustered by the resource clustering block based on the above QoS requirements. As shown in figure 2 the SLA based job scheduler has three phases namely SLA creation, SLA monitoring and SLA enforcement. It will create SLA's, monitor the QoS parameters specified in the SLA's, enforces SLA against violation if any and negotiate with the resource providers.

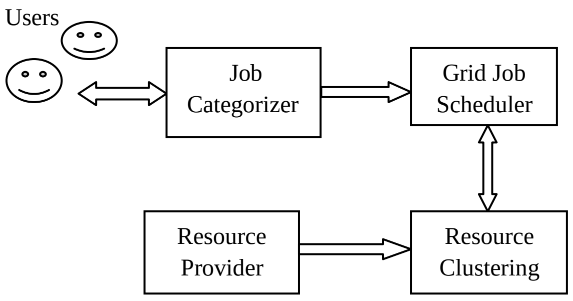

Figure 1. Architecture of Improved SLA based job Scheduler

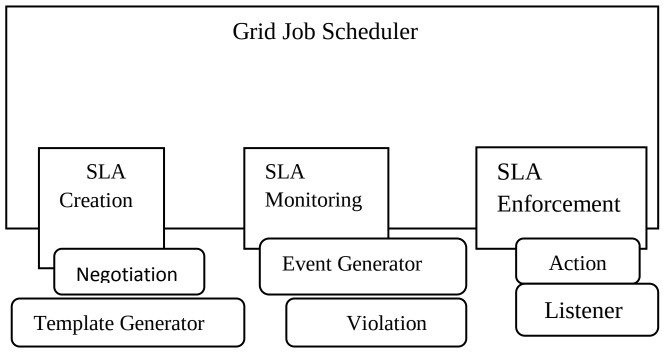

Figure 2. Phases of Grid Job scheduler

3.1 SLA Life cycle

The SLA life cycle consists of three phases namely SLA creation, SLA monitoring and SLA enforcement as shown in figure 3 below. The SLA life cycle starts with SLA creation phase. In this phase, the resources are selected for negotiation [2] based on their deviation value against the parameters specified in the job request that is obtained from algorithm 1. Algorithm 2 is used to negotiate with the selected resource providers to get their commitment. The SLA specifies the QoS parameters and penalty. The SLA monitoring phase monitors these QoS parameters. If there is any violation, they are informed to the SLA enforcement phase that enforces the SLA against the violation and generates necessary actions. [3]

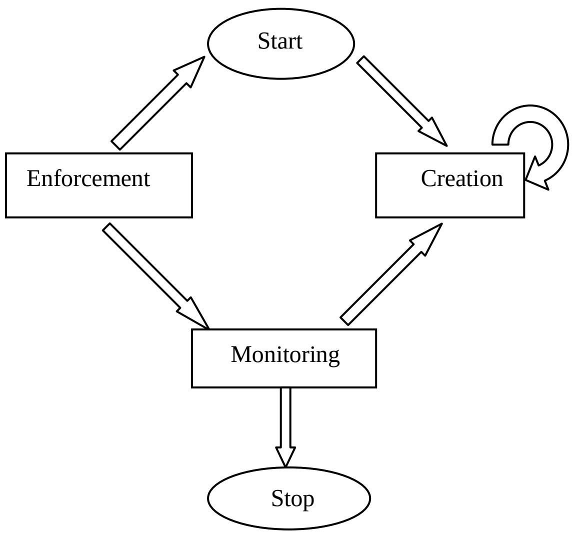

Figure 3. SLA life cycle model

3.2 Negotiation model

As shown in Figure 4 below the SLA template generator creates an SLA template and sends it to each resource providers who submit their individual commitment against current TSLA. The resource providers communicate with the negotiation module and create a tentative SLA using the received information from the resource provider and send it to the decision maker of the resource provider. The decision maker either rejects or accepts the tentative SLA. If it is accepted by any resource provider the storage module stores it in the database. If not accepted renegotiation is carried out and the rejected resource providers will be replaced by new ones from the grid scheduler

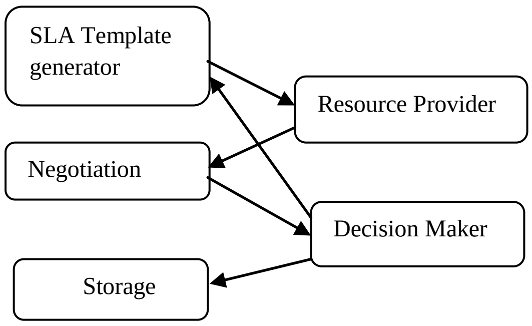

Figure 4. Negotiation Process

4. IMPROVED SLA BASED JOB SCHEDULING ALGORITHM

In our approach a user submits his/her job request to the job categorizer along with some QoS parameters such as number of CPU's needed to execute that job, security level and required OS. The jobs are categorized based on the above QoS parameters and then they are submitted to the Grid job scheduler. Resource providers will prioritize the available resources based on the above QoS parameters. This prioritization of resources will reduce the processing cost and time to a great extend. The submitted jobs are compared with the resources available in the resource provider. There are three possible outcomes for the above comparison

Case 1: The QoS parameters of the submitted job match exactly with the QoS parameters of one of the available resource.

Case 2: The QoS parameters of the selected resource are more capable than the QoS parameters of the submitted job

Case 3: The QoS parameters of the selected resource are less capable than the QoS parameters of the submitted job

In the first two cases the job can be executed with the matched resource provider without any problem. So in those cases negotiation is not necessary. But in the third case the resources are not sufficient to execute the submitted job, which leads for a negotiation with resource providers for the successful execution of the submitted job. In such negotiations agreements are used between user job and a resource provider. These agreements specify terms and conditions while executing a job and any violation of these agreements will lead to penalty and enforcement phase. These include Task Service Level Agreement (TSLA), Resource Service Level Agreements (RSLA). [5]

4.1 Improved SLA based job scheduling algorithm:

1. User submits a job to the job categorizer with all QoS parameters
2. All the jobs are categorized based on the Security level; Number of CPU's required to execute the job and Operating system preferred.
3. Providers submit their resources to the Grid job scheduler.
4. Resources are clustered based on the QoS parameters like Security level, Number of CPU's required to execute the jobs and OS available.
5. From the job categorizer jobs are compared with resource clusters.
6. If the submitted job QoS parameters exactly match with the available resources, jobs are submitted to the matched resource cluster.
7. If the capability of the available resources is greater than the QoS parameters of the submitted jobs, the jobs are submitted to the resource cluster.
8. In the above two cases SLA creation is not necessary. But in the third case, where the capability of available resources is not sufficient to allocate the jobs, SLA has to be created, because it requires one or more resource providers.
9. The scheduler initiates the SLA creation process
10. Upon successful creation of the SLA the Grid Scheduler submits the jobs to that resource cluster and the SLA monitoring Engine monitors for any violation of SLA.
11. If any violation of SLA is found, an event is generated by the SLA event generator of the SLA enforcement engine and it is notified to the enforcement engine of SLA and it generates corrective actions

4.2 Negotiation Model algorithms

The negotiation model consists of two modules Deviation based Resource Scheduling algorithm (DRS) and Negotiation Protocol (NP)

4.2.1 Deviation based Resource Scheduling Algorithm (DRS)

This algorithm is based on the percentage deviation coefficient. The values of the resource parameters specified in the job request are compared with the actual values of the parameters available in the resources. This comparison results in deviation value. Resources are selected based on this deviation value. This will reduce the number of negotiations needed.

Algorithm

1. For every available resources i = 1 to N and number of resource parameters j = 1 to m calculate the percentage deviation of i^{th} available resource for each j^{th} parameter specified in the job request against the available resources parameters using the equations

```
    $D_{ij} = A_v(t) – R_j(t) / A_v(t) * 100$   if $R_j(t) > A_v(t)$
    $D_{ij} = A_v(t) – R_j(t) / R_j(t) * 100$   if $R_j(t) < A_v(t)$
2.  Find $D_{ij\text{-}max}$ the maximum value among all $D_{ij}$
    Find $D_{ij\text{-}min}$ the minimum value among all $D_{ij}$
3.  For all values for $D_{ij}$
    if $D_{ij} >= 0$, divide $D_{ij}$ by $D_{ij\text{-}max}$ and store it in D
      ( $D = D_{ij} / D_{ij\text{-}max}$ )
    if $D_{ij} < 0$, divide $D_{ij}$ by $D_{ij\text{-}min}$ and store it in D
      ( $D = D_{ij} / D_{ij\text{-}min}$ )
4.  For all the available resources
    if D = 0, it is an exact match
    else if D > 0 and D <= +1, it is plug-in match
    else it is subsume match.
5.  The jobs are submitted for exact match and plug-in
    match  and subsume match leads to SLA negotiation
```

4.2.2 Service Agreement Protocol (NP)

The proposed Service Agreement protocol consists two components the Grid scheduler and resource provider and five messages, send, receive, accept, reject and commit

Pseudo code of SLA negotiation

```
For every resource providers
    if (message == “send”)
        send_empty_template();
        flag = true;
    if (message == “receive”)
        receive_filled_template();
    if(message == “send” and flag == true)
        send_tentative_SLA();
    if(message == “accept”)
        send_commit();
    if(message == “reject”)
        exit();
    else
        send_error();
```

4.3 The SLA Creation Phase

This phase consists of SLA template generator, SLA negotiation and Storage.

SLA template generator: Generates the template that consists of the SLA terms like earliest job start time, Latest job finish time, reserved time for job execution, Number of CPU nodes required and agreed price.

SLA negotiation: It collects the commitments of each of the LRMs, generates a tentative SLA and sends it to each resource providers who submit their individual commitment against current TSLA

Storage: The LRM will decide either accept or reject the tentative SLA. If it is accepted by all resource providers it will be stored in SLA data base. If not the resource providers who rejected the tentative SLA will be replaced with new resource providers from the grid scheduler and re-negotiation starts. After successful creation of SLA, the list of accepted resource providers is sent to the VO creation service.

SLA Monitoring

The SLA Monitoring Engine monitors the resource/task information which they receive in some specific intervals. Any violation of QoS parameters is notified to the event generator. According to the type and source of violation the event is generated by the event generator.

SLA Enforcement

Events are listened from monitoring engine and necessary actions are taken by the action generator based on events.

5. EXPERIMENTAL RESULTS

GridSim Simulation Toolkit has been used to create the simulation environment [4]. The inputs to the simulations are total number of gridlets, average MI of Gridlets, Granularity size and CPU clock cycle. The MIPS of each resource is specified in Table 1. The performance of the SLA based task scheduling has been compared with the performance of SLA based task scheduling with resource clustering.

This paper adopts GridSim 4.1 [4] to simulate the algorithms of task scheduling given above. GridSim 4.1 provides a series of core function for the establishment and simulation of heterogeneous distributed computing environment, particularly suitable for simulation and research of task scheduling in Grid environment. We simulated the algorithm with eight nodes, 10 seconds of granularity time, average MI of tasks 10 and CPU clock cycle 30. We have tabulated the results in Table 2 and Table 3 below

Table 1. MIPs of grid resources

Resource	MIPS
R1	24
R2	39
R3	120
R4	60
R5	72
R6	66
R7	180
R8	216

Table 2. Simulation of processing time for SLA and Improved SLA algorithms

No. of Gridlets	SLA algorithm	Improved SLA Algorithm
	Processing time in seconds	
50	505.4	416.01
100	1007.72	521.34
150	1508.33	620.41
200	2012.07	670.29
250	2507.34	782.23
300	3007.14	850.72

Table 3. Simulation of processing cost for SLA and Improved SLA algorithms

No. of Gridlets	SLA algorithm	Improved SLA Algorithm
	Processing cost in Rs.	
50	3333.34	1650.19
100	7966.67	2816.06
150	13440	3987.855
200	16940	7342.115
250	20440	7514.9
300	23940	11014.01

The MIPS of the resources is computed by multiplying the Total Processing elements in each resource by the MIPS of each processing element in the resource and the Processing cost is computed by multiplying Total CPU time for task execution and cost per second of the resources.

We have experimented the above algorithms and obtained results for SLA based task scheduling and SLA based task scheduling with resource clustering. We have compared the time of execution, cost for both existing and proposed algorithms.

a) Comparison of simulation time to complete task execution using SLA based task scheduling algorithm and SLA based task scheduling with resource clustering algorithm.

Simulations were conducted to analyze and compare the above algorithms in terms of processing time. Figure 5 shows a column chart for comparing the results of the above algorithms with average gridlet is length of 10 MI, deviation percentage 10% and granularity size 10 seconds.

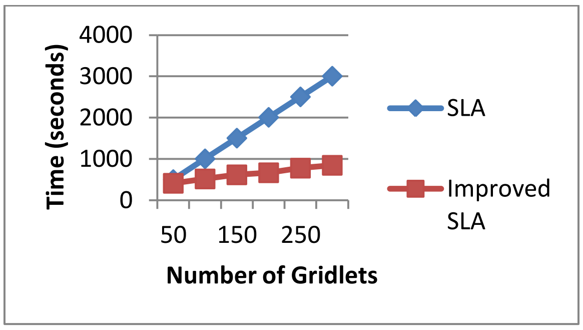

Figure 5. Simulation time comparisons of SLA and Improved SLA algorithms

The above graph proves that the time taken to complete task execution using SLA based task scheduling with resource clustering algorithm is very less when compared with time taken for execution of tasks using SLA based task scheduling algorithm.

a) Comparison of execution costs to complete task execution using SLA based task scheduling algorithm and SLA based task scheduling with resource clustering algorithm.

Simulations were conducted to analyze and compare the above algorithms in terms of processing cost. Figure 6 shows a column chart for comparing the results of the above algorithms with average gridlet is length of 10 MI, deviation percentage 10% and granularity size 10 seconds.

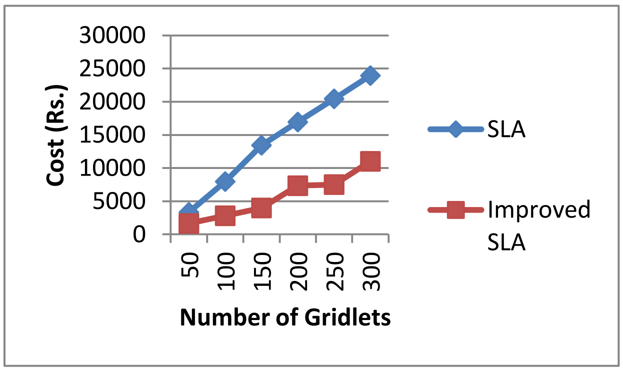

Figure 6. Comparison of Cost for SLA and Improved SLA algorithms

The above graph proves that the cost spent to complete task execution using SLA based task scheduling with resource clustering algorithm is very less when compared with the cost spent for execution of tasks using SLA based task scheduling algorithm.

6. CONCLUSION

This paper discusses job scheduling in a grid environment using a new algorithm called SLA based task scheduling with resource clustering. By doing research and analysis of the proposed algorithm, it aims task scheduling with minimum task execution time, minimum cost and maximization of utilization of resources. By using a new method of scheduling to cluster the resources according to their QoS parameters like Security level, Number of CPU's required to execute the jobs and OS available, the above aims are achieved. GridSim simulation Toolkit is employed to carry out and simulate the tasks assignment algorithm, and distributed task scheduling. The results are compared with SLA based task scheduling algorithm. The conclusion is that the scheduling algorithm employed is better than the existing algorithm.

This proposed algorithm only takes the initial research on task scheduling in grid environment. However many issues remain open. Further improvement should be done to handle more complicated scenario involving dynamic factors such as dynamically changing grid environment and other QoS attributes. The improvement of this algorithm should concentrate on discussing simultaneous instead of independent task scheduling in heterogeneous grid computing environment

7. ACKNOWLEDGMENT

The authors convey their heartfelt thanks to Dr.C.Kalaiarasan, Principal, Tamilnadu College of Engineering, Coimbatore and Mr.K.V.Chidambaran, Director Cloud Computing Group, Yahoo Software Development (India) Pvt Ltd, Bangalore for providing them the required facilities to complete the project successfully. This project is carried out as a consequence of the PSG-Yahoo Research programme on Grid and Cloud computing

8. REFERENCES

[1] Alexander keller, Heiko Ludwig, “The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services”, *Journal of Network and Systems Management*, Volume 11 , Issue 1,pp 57-81, 2003

[2] P.Balakrishnan, S. Thamarai Selvi and G.Rajesh Britto, “GSMA based Automated Negotiation Model for Grid Scheduling”, accepted to publish in SCC 2008WIP, IEEE Congress on Services, SERVICES 2008, July 8-11, 2008, Honolulu, Hawaii, USA

[3] P.Balakrishnan, S. Thamarai Selvi and G.Rajesh Britto “Service Level Agreement based Grid Scheduling”. In *proceedings of the 2008 IEEE International Conference on Web Services*

[4] R. Buyya, Manzur Murshed, “GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing” *Concurrency Computation: Practice and Experience, Vol. 14, No. 13-15,* 2002, pp. 1507-1542

[5] K. Czajkowski, I. Foster, C. Kesselman, and S. Tuecke, “Grid Service Level Agreements” , in J. Nabrzyski, J. M.Schopf, and J. Weglarz, editors, *Grid Resource Management*, pages 119–134. Kluwer Academic Publishers, 2004.

[6] K. Czajkowski, A. Dan, J. Rofrano, S. Tuecke, and M. Xu. *Agreement-based Grid Service Management (OGSI-Agreement)*. Global Grid Forum, GRAAP-WG Author Contribution, 12 Jun, 2003.

[7] C. Dumitrescu, I. Foster, “GRUBER: A Grid Resource SLA based Broker”, i*n EuroPar*, Lisboa, Portugal, 2005.

[8] I. Foster, C. Kesselman. The Grid, "Blueprint for a Future Computing Infrastructure [M]", USA, Morgan Kaufmann Publishers, 1999

[9] Jill Dixon and Melany Blackwell. Service Level Agreements. A framework for assuring and improving the quality of support services to faculties. In *Proceedings of the 2002 Annual International Conference of the Higher Education Research and Development Society of Australasia (HERDSA)*, 2002.

[10] Jin, L., V. Machiraju and A. Sahai. 2002, "Analysis on Service Level Agreement of Web Services". HPL-2002-180. HP Laboratories.

[11] MAUI, Maui Scheduler, Center for HPC Cluster Resource Management and Scheduling, www.supercluster.org/maui

[12] D.G.A. Mobach, B.J. Overeinder, and F.M.T. Brazier. A resource negotiation infrastructure for selfmanaging applications. In *Proceedings of the 2nd IEEE International Conference on Autonomic Computing (ICAC 2005)*, Seatle, WA, 2005.

[13] Yun Fu, Amin Vahdat, “SLA Based Distributed Resource Allocation for Streaming Hosting Systems” *http://issg.cs.duke.edu*

[14] I. Foster, C. Kesselman, S.Tuecke, “The Anatomy of the Grid: Enabling Scalable Virtual Organizations”, *International Journal of Supercomputer Applications*, 15(3), 2001.

[15] Moab Grid Scheduler (Silver), Available at http://www.supercluster.org/projects /silver/

[16] Community Scheduler Framework, Available at http://www128.ibm.com/developerworks/grid/library/gr meta.html

[17] A. Dan, K. Keahey, H. Ludwig, and J. Rofrano, “Guarantee Terms in WS-Agreement”, Technical report, Grid Resource Allocation Agreement Protocol (GRAAP) Working Group Meetings, 2004.

[18] J. MacLaren, R. Sakellariou, K.T. Krishnakumar, J. Garibaldi, and D. Ouelhadj, “Towards Service Level Agreement Based Scheduling on the Grid”, Workshop on Planning and Scheduling for Web and Grid Services (in conjunction with ICAPS-04), 2004