Proceedings of the 2010 14th International Conference on Computer Supported Cooperative Work in Design

An Extension of GridSim for Quality of Service

Reda Albodour, Anne James and Norlaily Yaacob
Distributed Systems and Modelling Research Group
Faculty of Engineering and Computing
Coventry University,
Coventry, CV1 5FB, UK
{r.albodour, a.james, n.yaacob}@coventry.ac.uk

Abstract— **GridSim is a well known and useful open software product through which users can simulate a Grid environment. At present Qualities of Service are not modeled in GridSim. When utilising a Grid a user may wish to make decisions about type of service to be contracted. For instance performance and security are two levels of service upon which different decisions may be made. Subsequently during operation a grid may not be able to fulfill its contractual obligations. In this case renegotiation is necessary. This paper describes an extension to GridSim that enables various Qualities of Service to be modeled together with Service Level Agreements and renegotiation of contract with associated costs. The extension is useful as it will allow users to make better estimates of potential costs and will also enable grid service suppliers to more accurately predict costs and thus provide better service to users.**

Keywords- GridSim, Grid, Quality of Service, Service Level Agreement, Renegotiation

I. INTRODUCTION

In Grid Computing applications where resources are spread over dispersed locations and run by different organizations with differing policies, it is important that the user is given a guarantee that their tasks be completed, and completed within specific guidelines that the user may wish to request. To achieve this, agreements need to be made between the resource providers and the users themselves, protecting and outlining the requirements, policies and rights of both parties.

Service Level Agreements (SLAs) contain descriptions of the users, their jobs, their requirements, as well as, the description of the Service Providers, the services they provide and their locations. These contract are vital for the operational integrity of a Gird Computing application. There has been quite a lot of interest lately in Qualities of Service (QoS) and SLA for Grid Computing (Gerndt et al 2009, Sakellariou and Yarmolenko 2008, von Laszewski et al 2005). Our work defines the qualities of service that need to be specified, defines the structure of Service Level Agreements and develops methods for exception handling, for instance when SLAs cannot be fulfilled. The work will then be implemented and added into a simulation toolkit, finally resulting in a more realistic, complete and updated simulation package.

The paper is structured as follows. Section 2 outlines the Qualities of Services that our model supports. Section 3 GridSim is described. Section 4 introduces the SLA and SLA templates used in our model. Section 5 introduces our framework other important elements. Section 6 discusses the implementation issues before concluding and adding further remarks.

II. REPRESENTING QUALITIES OF SERVICES

Quality of service is an important concept as it offers a basis for informing users about the sort of service they might expect. Often users will just wish to specify the time by which they would expect their job to be completed and would be happy to pay a predefined amount to achieve that. Other users may wish to reserve resources in advance for a particular time period. Others, who are more computationally aware, may wish to specify the services they expect more precisely through defining specific criteria. Currently, most Qualities of Services are defined as low level parameters that are, to the non-experienced computer user, vague and complex . Defining QoS at a higher level not only makes their measurement simpler but also helps users identify their requirements faster, more accurately and more realistically (Bhatti et al 2003; Rio et al 2003, Albodour 2008).

Another challenge is that of measuring Qualitative QoS. While Quantitative QoS such as bandwidth have standard measurement metrics that can be used, for example bandwidth can be measured by Mbps; Qualitative QoS such as *reliability* and *availability* do not and must therefore be defined with suitable metrics.

While there has been work in this field done in both identification of real application QoS and in simulation application QoS, our work identifies the higher level QoS that the user can control and request, both the quantitative and the qualitative ones. This provides the user with a clearer and more realistic approach to identifying their QoS before including them in our simulation package extension, ultimately allowing the users of the applications to be simulated and multiple application runs can be executed for more comprehensive results. We have also introduced measurement criteria for Qualitative QoS and introduce the measurement of resource reliability in this paper.

A. Quantitative QoS:

- Guaranteed number of Resources (Computational).
- Access period (Range of dates)

978-1-4244-6763-1/10/$26.00 ©2010 IEEE

- Memory per core (in MB)
- Average power of single CPUs (in GHz)
- Storage required (in GB)
- Bandwidth Required (in Mbps)
- Short term storage requirements.
- Required time of completion (time deadline in hours)

B. Qualitative QoS:

- Resource availability (in %)
- Resource reliability (in %)

C. Other QoS considerations:

Differing policies between different organizations and their resources have to be taken into account, as well as, authentication and authorization of users who may transcend their own domain into others in search of resources that meet their requirements for their applications, not to mention, the actual differing security domains within a single institutions. It is therefore vital for these considerations to be used in any simulation that tests an application that is to be run on top of a real Grid platform. It is for this reason that we have identified multiple levels of users explained later in this paper section, meeting some of the access control demands of modern applications.

III. GRIDSIM

The Simulation of Grid Computing applications is necessary because of the difficulties in testing applications on real Grid test beds; price, limited number and administrative complexities are some of those difficulties. The simulation solution is the logical substitute. Moreover, simulation allows these applications to be run multiple times, and experiments to be repeated for best results, accurate analysis and concrete development. However, current Grid Simulation tools are limited and do not provide the user with the ability to completely simulate real-life environments. Issues such as negotiation between the user and Service Provider, Contractual Agreements between them and Guaranteed Qualities of Services are only brushed upon and insufficiently covered. For applications in certain fields that aim to use the Grid Computing infrastructure as a vessel to carry out their operations in the future, the functions mentioned above and important. Before proceeding, it is important to introduce the toolkit we have chosen to expand and implement our model in which is GridSim (Buyya 2009) While flexible and intelligent, lacks the functionalities mentioned above, the functionalities the makes simulating, compute intensive, data intensive, on demand, real time medical applications accurately reflecting their operation in a real environment. The decision to use GridSim was down to its ability to support many more functionalities than other simulation tools, as well as the flexibility in which it was designed which allows additions and alterations to be made. Moreover, GridSims layered architectural model makes it easy to understand and add to. The following image illustrates this architecture. GridSim was the best available option of current Grid Simulation tool. However, even GridSim version 5.0. The architecture of GridSim is shown in Figure 1.

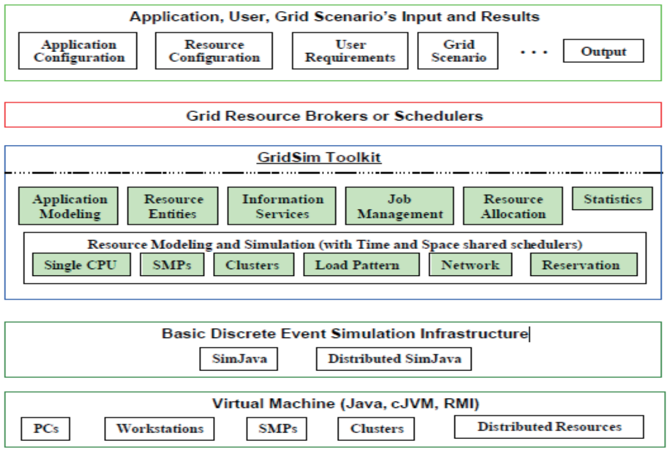

Figure 1. Figure 1: GridSim Architecture (Sulistio 2008)

GridSim's higher level operation can be summarized in four distinct operational steps. The first step is to identify and create Grid resources. These resources could be of different sizes and configurations and are created in relation to the experiment to be carried out. The next step is the creation of an application to use these resources. These applications are defined as a collection of "Gridlets" or Grid jobs that are created. The third step is the creation of the Grid user, which is the entity that interacts with the broker. This interaction leads to the coordination of the scheduling requirements for the simulation experiment. The final step is the entity responsible for allocating resources to the jobs scheduled in the experiment(Yagoubi et al 2008, Caminero et al 2007, Venugopal et al 2004,).

IV. SERVICE LEVEL AGREEMENT AND SERVICE LEVEL AGREEMENT TEMPLATES

In our model we aim to expand the capabilities of the GridSim to add new features to support the processes of negotiation and renegotiation of SLAs. This will make GridSim a more realistic simulation package that can be used with a wider range of Grid Applications and can support a broad range of QoS requirements, which may vary according to differing circumstances of users, even when running the same job. Users will be able to input more specific QoS requirements. The introduction of databases into the simulation process will allow automated renegotiation to be simulated. GridSim has been developing since its initial release. Gridsim 5.0 allows the users to develop their own scenarios, schedulers and allocation policies. These, in addition, to the recently added advanced resource reservation capabilities have given the simulation package new dimensions.

The successful migration of Grid computing from the purely scientific domain into the business-oriented service marketplace relies on the delivery of the QoS we have explained above. Therefore a clear relationship must be identified between the users of these resources and services and those who provide them. These relationships are governed by electronic contracts between them.

These resource providers belong to different organizations with different policies and different definitions of metrics and resource capabilities. For example, the simple measurement of the cost of using a resource is not as simple as its first impression suggests. One resource's cost might be measured in US$ while the other could be by the GBP£, and so forth. Our expansion of the simulation software has sought to accommodate such issues. The first step was providing the user with an interface. Using this interface, the users can input their QoS requirements into a drop down list based menu with specific metrics for QoS. These metrics were outlined earlier in this paper.

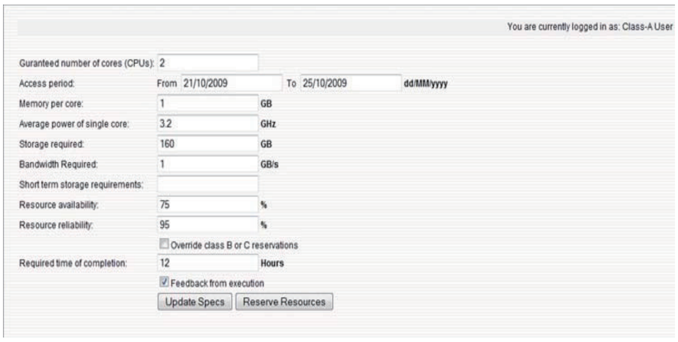

Figure 2. Figure 2:Interface Design Preview for Upper Tier User

Once the user has input their requirements, they are stored in a template SLA file. These SLAs are given IDs and stored in the database that is attached to the toolkit. The Resource providers use the same template to input the characteristics of the resources that they want to advertise. This effectively means that the user and resource providers will be speaking the same language when negotiation commences, minimizing mismatched metric measurements. The meaning of the names and units used in the standard template will made available to all Grid users (see Figure 2).

The use of templates allows both the users to specify their requirements clearly and the Service Providers to advertise and offer their services and resources. In real environments this implies that the interaction between humans in creating Service Level Agreements is minimized, which in turn leaves little space for error and is in the interest of both parties in creating a viable electronic contract.

This also reduces negotiation time significantly, as the matchmaking process is more direct and less complex. This saves both time and cost. The outcome of this negotiation process is a contract that guarantees that the resources are allocated to the task for the entire duration in which they should be allocated and according to the agreed upon guidelines.

V. Other important elements of the model

A. *Databases*

The introduction of databases into the simulation process is a novel approach that we propose for all phases of the simulation run, adding to the realistic execution of application runs in a simulation environment. User information, including their IDs, Levels and virtual Organizations are kept in these databases. Moreover, GIS information and SLA templates are also held in these databases.

The characteristics for every resource that are advertised by the resource provider will also be held in the database, as well as, the success and failure rate for every specific resource over a specific number of runs. This will aid in the measuring qualitative QoS of the resource before it is allocated to another user and other tasks. The Resources IDs are kept in a table inside the GIS where they register, and if they are available for global use, their IDs are registered with the global GIS. These IDs are used as pointers to parse the database of resources and retrieve the information on the resource with the matching IDs maintaining up to date information on each resources at a specific period of time.

These databases also hold the Service Level Agreements themselves, in case they need to be referenced according to a pre-stated condition between the user and Service Provider, such as the failure to deliver a specific percentage of the QoS required. If a breach of contract occurs, then renegotiation is invoked and referencing the original agreement is important, before a new agreement is reached and replaces the original one.

Others objectives and usages of introducing databases into the simulation process will be discussed in future work and are not within the scope of this paper.

B. *Types of Users*

Three different levels of users are introduced instead of a single level. Every user will be assigned an ID that relates to the level that they are a part of. For simplicity and the purposes of this paper, we have called them Level A, B and C. Each level of users identifies the privileges that its community has access to; both in terms of resources and the amount of requirements they can set for allocation of the resources they have access to.

The highest levels of users, Users of class A, have access to both local and global resources. Local resources being those within the same Virtual Organization and global ones are those who are registered with the global Grid Information Service (GIS) by their owners and are advertised to all connected Local administrative domains. Another advantage of being part of this category for the user is the ability to specify a much wider and more concrete set of QoS they require from resources, both in terms of computational resources and storage resources. Finally, users in this category have the highest priority, giving them easier access to resources, the ability to request specific resources at peak times, the ability to override a reservation from users in the other two lower levels when they request specific resources. This priority is only employed within the user's local administrative domain and does not expand to other virtual organizations, where the priority is assigning local resources to local users.

Levels B and C enjoy lesser privileges than those given to users in the highest level. Both of these levels of users have a more limited set of requirements. They can request with level C users limited to storage resources. It is important to note that even though Level B users will are able to make use of computational resources, they will not be able to access resources not registered with their local GIS at peak times, which are weekday working hours.

C. Grid Information Services (GIS)

The Grid Information Services Consists of two types; Local GIS and Global GIS:

1) Local GIS

The first step that Service Providers do in order to advertise their resources is register them with the local GIS, which maintains an update copy of all available registered resources as well as a communication link with the Global GIS.

The resource IDs are kept in the local GIS in a maintained array. Each ID points towards a row in the resource table in the database where the characteristics of every resource are advertised by their provider and are stored according to eh metrics that are specified in the SLA templates we have introduced.

Once the resource has been allocated, its ID is removed from the available resources array making it inaccessible until the task is complete, therefore guaranteeing the rights of the user which it has been allocated to. Once the resource has finished its allocation period, the information is updated in the database, and the ID is returned once again into the table of available resources.

2) Global GIS

A Global GIS holds an array with the IDs of available resources in local GISs which are allowed to be assigned to tasks not within their own VO. This list is updated routinely through communication between the global GIS and the local GIS. It is worth noting that there is no direct communication between the Service Provider and the global GIS. All communication between them is done *via* the local GISs.

D. Framework components (see Figure 3):

1) *Resource Manager(RM)*
2) *Access Control Manager*
3) *Allocation Manager*
4) *Accounting Manager*

E. Framework operational phases:

1) *The initiation phase*
2) *The operational phase:*
   a) *Because of contract breach.*
   b) *Because of user request.*
3) *The termination phase*

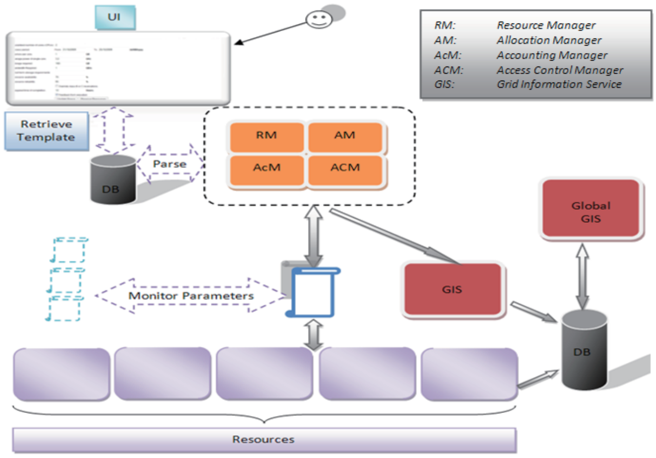

Figure 3. Figure 3: Our Framework

F. The proposed framework will be responsible for:

- Validating the operational workflow
- Dynamically retrieve status information on resources
- Select the appropriate resources
- Assure the selected resource conform to the privileges of the correct user authorization
- Manage the negotiation process
- Manages resource allocation
- Allocate resources according to user requirements.
- Manage accounting and billing.

When the Service Providers want to make resources available, they are advertised and registered with their local GIS. A list of available resources and their IDs is maintained and updated by the GIS on a regular basis. The GIS is connected to a database through which information on resources can be obtained. This information is provided by the service providers and is used in order to negotiate SLAs when the resource is to be allocated to a specific job.

Once the user enters their requirements, according to the tier they belong to, these requirements are stored, using the XML templates into a database. These templates are parsed and the information contained within them is retrieved in order to allocate the resources to the user. Once these requirements have been found, the local GIS is asked for information on behalf of the user on the resources available that fit the exact requirements. The GIS checks the list of resource IDs that it has available to make sure that they are not allocated to another job during the time where the other request requires them. If the required resources are found to be available over the period requested, they are allocated to the user. The GIS updates its list accordingly and removes the resource from the available resources list.

The local resources may not be sufficient or compliant to the requirements of the user. If that happens, the GIS contacts the others GISs to retrieve information on available resources registered with them, however, this presents a logistics issue. In reality, while a user may have full access to local resources, they may not be allowed full access to resources that are registered with other Virtual Organizations. Moreover, even if a user has the permission to access these resources, the other Virtual Organization may restrict access to those resources. This is an issue that is not covered within this research, but it is mentioned for completeness.

VI. IMPLEMENTATION ISSUES

In our model we aim to expand the capabilities of the GridSim package to add new features to support the processes of negotiation and renegotiation of SLAs. This will make GridSim a more realistic simulation package that can be used with a wider range of Grid Applications and can support a broad range of QoS requirements, which may vary according to differing circumstances of users, even when running the same job. Users will be able to input more specific QoS requirements and the introduction of databases into the simulation process will allow automated renegotiation to be simulated. GridSim has been developing since its initial release. GridSim 5.0 new features such as, resource failures can be simulated. It also allows the users to develop their own scenarios, schedulers and allocation policies. These, in addition, to the recently added advanced resource reservation capabilities have given the simulation package new dimensions (Sulistio et al 2008).

In order to allocate the resources successfully, a new scheduler is developed. This scheduler conforms to the model that we have presented and provides a new addition where the pre-allocation, the allocation and the post allocation process for resources are addressed. Moreover, in order in section 2 we introduced criteria that should be measured. These criteria can be used for QoS specification. They can also be used as metrics during the processing of a job to ensure that the processing is being carried out according to agreed QoS. To accommodate these criteria alterations and expansions to some of the GridSim classes are required.

In addition, a new method for dynamic renegotiation is introduced. Information on the execution of any job is retrieved over specific intervals during the allocation period. A new entity is introduced for this purpose, called a re-scheduler.

If any contract violations occur during the job execution, the re-scheduler is invoked and a renegotiation process occurs. Much like the original scheduler, it selects the appropriate resources to replace those that have not been performing and allocates them accordingly. The main difference is it does so dynamically during the execution.

Figures 4 and 5 show the two different outputs of a simple experiment which outlines resource allocation according to the *Reliability* Parameter introduced earlier in this paper. In figure 4, the user's request is fulfilled by the resource which advertises its characteristics. The user can then send their jobs knowing that their requirements have been met. In Figure 5, the resource characteristics do not meet the user requirements and therefore the job cannot be allocated to that resource, the feedback is sent back to the user, while the search for a resource that satisfies the *Reliability* requirements of the user carries on.

```
Output - dsm.coventry.gridsim (run-single) #2
Initialising...
Creates one Grid resource with name = GridResource_0
Creating a grid user entity with name = User_0, and id = 9
User_0:Creating 4 Gridlets
Starting GridSim version 5.0
Entities started.
User_0:Received ResourceCharacteristics from GridResource_0, with id = 6
User_0:Sending Gridlet_0 to GridResource_0 with id = 6 at time = 11.56
User requires Gridlet Relibility0.62
Resource Reliability0.7
Ack = true

User_0:Sending Gridlet_1 to GridResource_0 with id = 6 at time = 84.2
User_0:Sending Gridlet_2 to GridResource_0 with id = 6 at time = 84.2
```

Figure 4. Figure 4: Reliability Requirements are met

```
Output - dsm.coventry.gridsim (run-single) #3
Initialising...
Creates one Grid resource with name = GridResource_0
Creating a grid user entity with name = User_0, and id = 9
User_0:Creating 4 Gridlets
Starting GridSim version 5.0
Entities started.
User_0:Received ResourceCharacteristics from GridResource_0, with id = 6
User_0:Sending Gridlet_0 to GridResource_0 with id = 6 at time = 11.56
User requires Gridlet Relibility0.62
Resource Reliability0.52
Cannot schedule due to reliability not meeting User_0 requierments..searching for alternate resources
```

Figure 5. Figure 5: Reliability Requirements are not met

Current Scheduling algorithms in Grid Computing such as the FCFS algorithm and the EASY Backfilling do not provide the functionalities required by users in time-sensitive compute sensitive applications as they provide a scheduling method for jobs. They are simple with FCFS Non-pre-emptive method, with a scheduling process that is essentially a queue of jobs that vary in length treated under the First In First out (FIFO) concept. EASY Backfilling is slightly more intelligent as it allows the shorter jobs to move forward in the queue if the required resources for their execution are available (Skovira et al 1996 and Franke et al 2007)

While Queues are employed within our model, they are only used in special cases where the resources that meet the requirements are all in use and therefore a queue based on FCFS is used at the resource with the shortest estimated time for availability. Our model, through searching for the appropriate resources and allocating them to users according o their privileges and QoS requirements as shown in figure 6 provides a more realistic and functional approach to resource applications for modern Grid applications. Its deployment within the GridSim toolkit allows the user to run simulations of Grid applications, more comprehensively and allows the simulation of application with specific run requirements and guidelines, QoS.

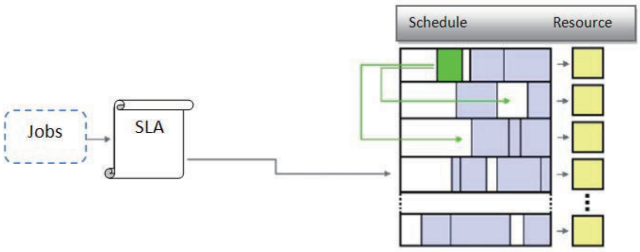

Figure 6. Figure 6: Searching for the appropriate resources

VII. CONCLUSION

In this paper we have described an extension to a successful Grid simulation tool, Gridsim. The extension enables varying QoS to be expressed and maintained. This is achieved through the identification a number of resource characteristics that affect performance. Such characteristics can be used both to specify QoS and as metrics to measure performance against an agreed SLA. Complications may arise if resources fail or if infrastructures across Grids differ and cause deterioration of service. In this case renegotiation of SLA may need to occur. The necessary extension of existing classes and the required introduction of new classes have been described.

The introduction of a new framework that performs the tasks required as well as novel measurement techniques for specific high level QoS within this paper paves the way for realistic main stream applications to be simulated, as well as more future expansions to be carried out.

REFERENCES

[1] Albodour R, James A, Yaacob N ,Godwin A (2008) , *QoS Requirements for a Medical Application on the Grid*, Computer Supported Cooperative Work in Design IV: 11th International Conference, CSCWD 2007, Melbourne, Australia, April 2007, Revised Selected Papers, LNCS 5236, Springer

[2] Belalem G, *Yagoubi* B (2008), Hybrid Negotiation Mechanism for Resolution Conflicts between Replicas in Data Grid, MUE 2008, pp215-220

[3] Bhatti S, Sorenson S-A, Clarke P and Crowcroft J (2003). *Network QoS for Grid Systems*, International Journal of High Performance Computing Applications, 17, 3, pp 219-236

[4] Buyya R (2009), CLOUDS Lab to release GridSim Toolkit 5.0, press announcement, available at http://buyya.wordpress.com/2009/10/12/clouds-lab-to-release-gridsim-toolkit-5-0/

[5] Sulistio A, Cibej U, Venugopal S, Robic B and Buyya R (2008), A Toolkit for Modelling and Simulating Data Grids: An Extension to GridSim, Concurrency and Computation: Practice and Experience, Volume 20, Number 13, Pages: 1591 - 1609, ISSN: 1532-0626, Wiley Press, New York, USA.

[6] Caminero A, Sulistio A, Caminero B, Carrión C, Buyya R (2007), *Extending GridSim with an architecture for failure detection*, ICPADS 2007, pp 1-8

[7] Gerndt H M, Rana O F, Ziegler W, and Von Laszewski G (2009), *Service Level Agreements in Grids*, Dagstuhl Seminar Proceedings, available at http://drops.dagstuhl.de/opus/volltexte/2009/2027

[8] Skovira J, Chan W., Zhou H., Lifka D. (1996) The EASY-LoadLeveler api project, In proceedings of the workshop on Job Scheduling Strategies for Parallel Processing, LNCS, colume 1162, pp. 41-47.

[9] Rio M, di Donato A, Saka F, Pezzi N, Smith R, Bhatti S and Clarke P (2003), *Quality of Service Networking for High performance GRID applications*, Journal of Grid Computing. 1 ,4, pp 329-343.

[10] Sakellariou R and Yarmolenko V (2008). *Job Scheduling on the Grid: Towards SLA-Based Scheduling.* In Lucio Grandinetti (editor), High Performance Computing and Grids in Action, Volume 16 in the Advances in Parallel Computing series, IOS Press, , pp. 207-222.

[11] Sulistio A, Poduval G, Buyya R, Tham C-K (2007), *On incorporating differentiated levels of network service into GridSim*, Future Generation Comp. Syst., 23, 4, pp 606-615

[12] Franke C, Lepping J, Schwiegelshohn (2007), *Greedy Scheduling with Complex Objectives,* Proceedings of the 2007 IEEE Symposium on Computational intelligence in scheduling.

[13] Venugopal S, Rajkumar Buyya R, Winton L J (2004), *A grid service broker for scheduling distributed data-oriented applications on global grids.* Middleware for Grid Computing 2004, pp 75-8.