

Resource Discovery using Brokering with Dispute Solving in Grid Environment

Prakash M*, Chandrasekaran Subramaniam**, Ravichandran T***

*Research Scholar, Jawaharlal Nehru Technological University, Hyderabad, Andhra Pradesh, India

**Dean & Professor, Information Technology Department, Rajalakshmi Engineering College, Chennai, India

***Principal, Hindustan Institute of Technology, Coimbatore, Tamil Nadu, India

salemprakash@yahoo.com, chandrasekaran_s@msn.com, dr.t.ravichandran@gmail.com

Abstract—The objective of the work is to propose a resource discovery model for distributed resources using grid brokering and dispute solving techniques in a grid environment. The synchronization between local and external schedulers and the effective utilization of the available resources can be achieved by dispute solving with periodical resource auditing processes by an agent technique. The discovery of resources after the dispute is the main focus of the work where the proposed Detachable Identifiable Secured COVERage (DISCOVER) algorithm discovers the resources in the case of resource conflicts and friction. The DISCOVER algorithm first detaches the misbehaving or faulty resources due to disagreement and identifies not only the compatible resources based upon the operational speed of the processes to complete the given task described by the Job Submission Description. The interoperability issues due to the varying operational environment like operating system, application and browser are considered in the discovery model. The model was checked with the integrated framework with Grid Bus Broker (GBB), a middleware Alchemi and SQL server.

Keywords— Grid Broker, Dispute Solver, DISCOVER algorithm, Inspecting Agent, Job Submission Description.

I. INTRODUCTION

The Grid scheduling is responsible for resource discovery, resources selection, job assignment and aggregation or group of resources over a decentralized heterogeneous system. The economics based resource management model is capable of producing optimal solution for multiple resources scheduling with static and dedicated resources to balance the processing loads efficiently [1]. But the model being a static one may not produce expected performance in the case misbehaving or faulty resources during run time. The earlier resource selection algorithm is based on input jobs, communication links and resource computational capability using statistical patterns of job insertion into system with different distributions [2]. This technique is based on the computational cost in terms of Round Trip Time (RTT). The notion of introducing an auditing process regarding the availability of the isolated hardware and application software resources for the assigned tasks is not considered which will not only increase the grid reliability but also reduce the round trip time for the processes.

The Quality Ant Colony Optimization (QACO) model uses the utility as heuristic information to finish task scheduling effectively, and it can maximize user's utilities at the same time [3]. The QoS dimensions like time, reliability, version, security and priority are considered in the task scheduling model without focusing the negotiation process with the grid users to tolerate the schedule slippage or change in the cost constraints. An improved ant algorithm becomes more sensitive with the power grid environment and more robust with heavy work load using execution quality prediction technique with the help of genetic algorithm [4]. The work does not address the nature of the jobs and their time of submission into the grid environment in that case the prediction may not yield the correct values but considered the scalability of the model for large work loads. The Job Submission Description is needed to avoid the above situation and then the grid manager can schedule the tasks. In the other scheduling algorithm concentrated the periodical resource reallocation for a grid application in which it migrates the ongoing tasks from a set of computing nodes to another set so as to fully employ the newly available computing power [5]. If there is a chance of resource conflict and dispute between two concurrent applications, the selection and migration of the running tasks into new computing power will be an issue.

The selected nodes that are based only on their recently observed resource capacities for scalability reasons can lead to poor deployment decisions resulting in application failures or migration overheads. Condor matchmaking relies on a centralized matchmaker for matching resources to applications [6]. The detachability and the ease of identification of resources is needed before reallocation to the jobs. In the tender and contract-net protocol using the Grid Resource Broker, the GRBs announce their task requirements and invite bids from Grid Service Provider GSPs and the interested GSPs evaluate the requirements and submit their bids. [7]. The Max-Min and slack suffrage based scheduling considers the dynamically arriving tasks with priorities and multiple deadlines were mapped using some heuristics approach [8]. The main focus of the paper is to propose a resource discovery model for the reallocation when resource conflict occurs and an algorithm to solve the disputes based on the job resubmission description through Grid Broker component.

The paper is organized as follows: Section II discusses a generic architecture framework for dispute solving in the case of resource conflicts with negotiation as interactions between the various component services. Section III explores a Job Sequence - Resource Conflict scenario in which the job sequence has to be modified in order to minimise the downtime due to conflicts between processes. This can be achieved by brokering the resources through resource auditor component. Section IV proposes a DISCOVER algorithm to identify the resources in a secured way after the conflicts. Section V describes the essential grid broker services needed to implement an Extensible Resource Compromise Agreement (xRCA) technique to evaluate the number of disagreement and the proposed affidavit acceptance ratio to solve the disputes and concludes the work.

II. GENERIC FRAMEWORK FOR DISPUTE SOLVING

The resource discovery models and techniques use a recent resource usage approach or resource bundling techniques with iterations without focusing the allocation in the case of disputes due to disagreement in runtime. A parallel composition of processes is used to identify the free and secured resources. The grid broker service component collects the information of grid resources like availability, cost and the completion time of the resources for any assigned jobs through collector agents. From the grid user, grid broker can also gathers information regarding the application complexity and the expected or anticipated cost to complete the job. The dispute solver service component identifies the conflicting hardware and software resources when there is any problem due to transient faults or disagreement after negotiation it may be due to the lack of synchronization in inter-process communication between the agents and the scheduler. The application security is only of the important factor it checks the existing security mechanism for both application and informs to the broker. The algorithm identifies a set of detachable and compatible, secured resources for the remaining jobs in the queue. The Inspecting agent inspects the resources if there is any disagreement due to transient faults and software incompatibility. The Resource auditor service component estimates both the hardware and software for the damage cost and cost over disputes and also audits the cost over software incompatibility and wait time of the job in the queue. There is a common access control mechanism by which a system grants or revokes the right to access resources as shown in Figure1. The proposed resource discovery starts with the prior reservation of the resources to schedule the tasks efficiently with a strategy called Dispute solver through conflict graph solving. The Broker component or service holds the static list of resources in the form of a table known as job – resource table. If the resources has been utilized by some other task, due to some reason the resources may not be released even after the expiry of its execution time and a forthcoming grid user needs that particular resource, then resource conflict will occur. To overcome this problem, the job may be cancelled or swapped thereby releasing the locked resource. The proposed work is modelled and evaluated as

control process flows for the effective scheduling of dynamic grid resource discovery and allocation.

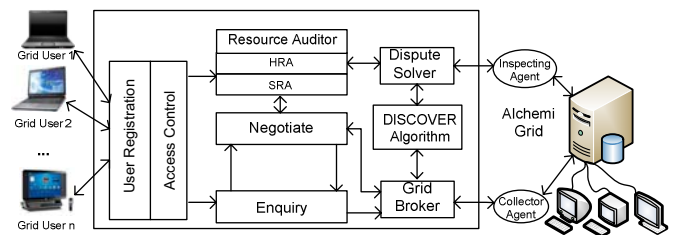


Figure 1. Resource Discovery using DISCOVER algorithm with Broker and Dispute solver

The interaction diagram starts from the grid user to grid broker. If grid user wants to submit a job that is to be executed, first the user is expected to register through a secured interface after signing an agreement with the grid manager. The Grid Manager (GM) forwards the request will all its credentials through the grid broker. Then based upon the application requirements of the submitted jobs, the grid broker collects the information about the hardware resource and software resource availability in the form of a Dynamic Resource Table from resource auditor. This information table is shared with the user to draw an agreement and acceptance for the cost of service demanded by him in the form of a registration and a signed document. If the grid user gets satisfied with the terms and conditions regarding the cost and the execution time of the jobs, then the job will be submitted to the grid environment through the grid broker as shown in figure 2. After submission, if there is a chance of transient failure in the grid then the job has to be resubmitted again in the next attempt. At this point grid broker negotiates with the grid user to compromise for the loss of time and if grid user accepts then the jobs are executed by the available resources. The proposed dispute solver component service accepts the failure notice from the resource manager and submits the nature of demand to the auditor component service to know the future recovery action or alternate solution for the current failure scenario. In fault free condition, the jobs will be executed with the help of fault free processes.

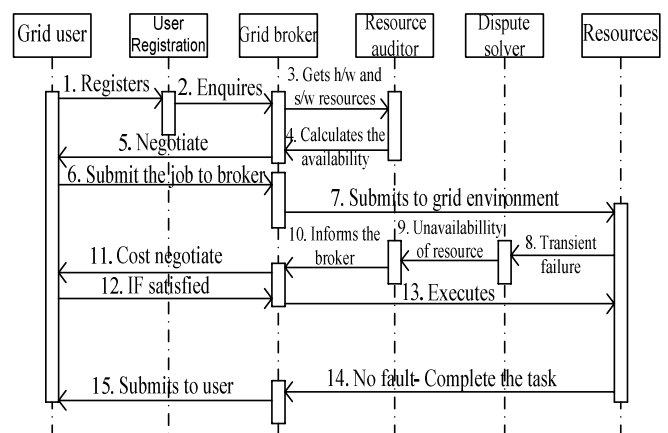


Figure 2. Resource Auditor- Dispute Solver Interactions

III. JOB SEQUENCE- RESOURCE CONFLICT & BROKERING

The Job Sequence- Resource Conflict (JSRC) may occur not only due the concurrent demand of the resources by different tasks in the job sequence but also due to the abruptly failed resources during run time. Apart from the grid resource information and infra structure services through grid broker service, a dispute solver service is needed in the case of resource conflicts that may deteriorate the performance of the overall grid services. In order to implement the dispute solver service, a resource auditing service is essential to continuously or periodically monitor the status of all the resources that are currently provisioned and de-provisioned by various applications in the grid. The JSRC can be solved through the proposed DISCOVER algorithm which identifies the detachable secured resources in the grid with the help of the resource auditor. The considered jobs are from J1 to J10 and the various resources from R1 to R9 while the Resource auditor audits the available software and hardware resources. Here the mapping of submitted jobs and the required resources are shown in figure 3.

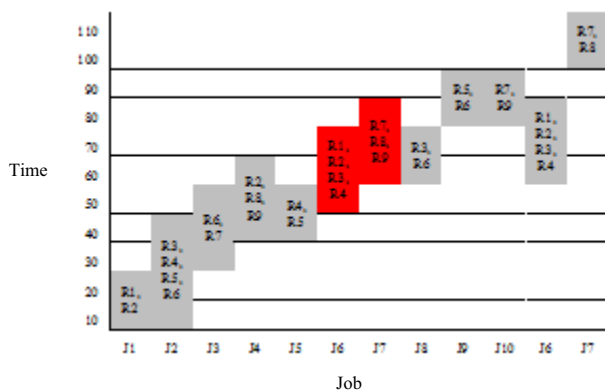


Figure 3. Job Vs Time (Resource Conflict)

While executing the Job 5, it required the resources R4 and R5, but the same resources are hold by J2 where these resources can work parallel so J2 and J5 are executed concurrently. In the case of J6 and J7 their required resources are R4 and R9 are hold by J5 and J4 respectively. They want to execute concurrently, in this case, after the release of the resource from J6 and J5 only J6 and J7 can complete their work. This is due to the resource conflict. The GRID resource disputes can be solved by following the stages like compromise and affidavit during run time as shown in the Figure 4. Due to lock or not release of the resource, after the job execution time the dispute will occur. Before submitting the job to the grid environment, the broker resolves the problem then grid bus broker sends to the grid environment. If there is any dispute it should be reported so that the broker has to be resolved. Generally due to the dispute, disagreement will occur even if the allocated execution time of the job is not met. Due to the disagreement resources may not be released, at this point conflict will occur. The one and only way is that the negotiation is done then we can make compromise and settle the job be resubmit otherwise i.e., negotiation fails then the agreement can be cancelled for the new case of

resubmit/cancel a job. A new process of affidavit is made to proceed the same.

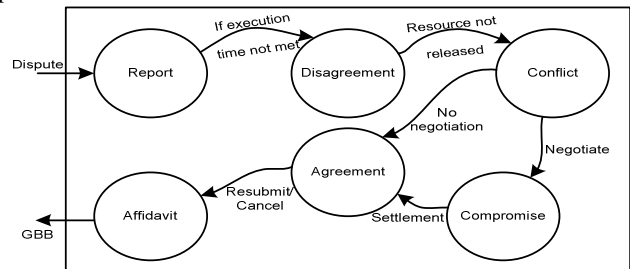


Figure 4. Dispute solving and Agreement acceptance

IV. DISCOVER ALGORITHM

The performance of the GRID application is depending on the success rate with which the disputes have been solved the corresponding management services embedded within the grid. The disputes once submitted are to be converted into style sheets form or in report template and reasoned into disagreement based on the application requirements and accepted specifications. Even under the normal working conditions, if the resource release time is not followed by the resource due to intermittent faults in the application side, then the conflict occurs that means the job is neither executed nor completed. This unwanted scenario will not only affect the resource utilization but also the overall performance factor of the grid. The negotiation starts with the user to come for a compromise in terms of cost reduction or penalty for the unprecedented delay through negotiation services. If that too does not arrive for an agreement within the stipulated time period fixed by the manager, then both the ends come to accept for an affidavit signed by the dispute solver component after retrieving the status from the auditor component service. Here, the DISCOVER algorithm is followed to detach the resources due to their non performance and identify the damages caused in terms of cost and completion time. Then the jobs are to be securely assigned to free and trusted resources to complete them within the accepted time as per the conditions mentioned in the affidavit at that instant of time in XML as shown below:

```

<disagreement>
  <resource = 'not available'>
    <execution time = 'not possible'>
  </disagreement>
</conflict>
  <report = 'jobid', ' cost', 'WCET', resource_id >
    <cost = negotiable | non negotiable>
  </conflict>
<compromise conflict>
  <agreement>
    <settlement>
      resource >> allocate | release
      affidavit = 'get back' | 'resubmit'
    </settlement>
  </agreement>
</compromise conflict>

```

The proposed DISCOVER algorithm in which the disputed resources are reallocated through agreement service shown:

```

DISCOVER_algorithm()
begin
  Global:
  R ← resource available, Rc ← resource conflict
  jobi → job for execution
  allocate_resource : Boolean
Detach_Identify::
  if (jobi.resource == jobi[not released])
  {
    jobi.Rc //Resource conflict
    if (jobi[resource released] == released)
      jobi.resource → allocate_resource();
    else
      negotiate ();
  }
  else jobi.Resource → allocate_resource ();
  if((jobi.Allocation + jobi.Execution) < jobi.Deadline)
  {
    disagreement( report);
    suspend( job);
    detach ( job, List);
    call auditor();
  }
  else
    allocation_resource → Success
end

Secure_allocation::
allocate_resource()
{
  jobi.Execution ← Executiontime
  jobi.Worstcase ← WCET
  jobi.Deadline ← Deadline
  jobi.Release ← Releasetime
  jobi.Worstcase == jobi.Execution + Tm
  // Tm → MeanRelease Time ≥ Execution time
  jobi.Deadline = jobi.Worstcase + Td
  // Td complete the job
  if(jobi.WorstCase < jobi.Deadline )
    negotiate();
  else
    return();
}

Resource_Coverage_Auditing::
auditor()
{
  notify_broker ();
  negotiate_grid_user();
  if grid_user accept= "YES"
    resubmit();
  else
  {
    cancel_job();
    release_resource();
  }
}

```

The jobs may be resubmitted to the manager or withdrawn from execution based on the hardware and software resources available during the next sessions. The various stages of dispute solving and agreement acceptance through affidavit is shown as a simple state transition diagram. The time complexity of the algorithm depends not only on the number of disagreements over a grid service session but also on the negotiation period. The computational decidability of the resource discovery over a dispute evaluates the finite selection probability of the free resource or the availability of the trusted resource for the job submitted that is forwarded through the affidavit process.

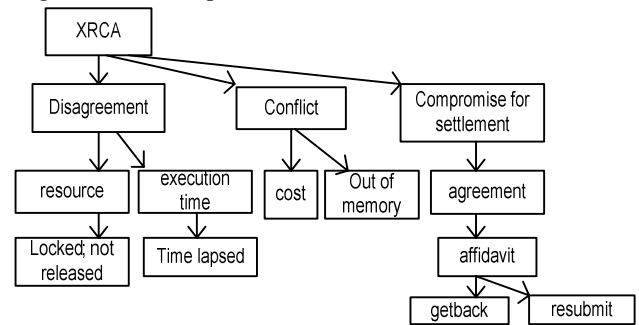


Figure 5. XRCA flow diagram

The extensible resource compromise agreement can be achieved if there is any disagreement or conflict in their resource and cost of their utilization of resource. The disagreement will happen when the resource is locked or its utilized by other resources. In other case if the execution time is more than time allocated then disagreement will occur. Due to the disagreement ie resource is locked and not released; at this point resource may be needed for other user due to this conflict will occur. Due to conflict, cost negotiation is done between the users. If there is any disagreement or conflict between the users a compromise settlement can be made by new agreement. For a new agreement a affidavit can be made either to resubmit the job to the other new resource or the get back to submit the job the same resource.

Computational Model of evaluating the proposed algorithm:

Let T be grid service session time and the number of disagreement be 'n' out of N jobs submitted over that session. The Disagreement Ratio may be immediately calculated by the manager service component as in equation (1).

$$\text{The Disagreement Ratio} = n / N \quad \text{----- (1)}$$

The probability of the need to look at the disagreements may be determined by the product of the severity factor S_i and criticality factor C_j of the task submitted.

The hearing period is determined as the product of equation(2)

$$T_{\text{hearing}} = T * (n/N) * S_i * C_j \quad \text{----- (2)}$$

Affidavit acceptance ratio is denoted by N_{RCA} that can be determined as the ratio between the numbers of tasks resubmitted to total number of disagreements through the respective services activated by the manager, broker and auditor component services in that grid application session.

V. GRID BUS BROKER ENVIRONMENT

The grid bus broker is designed to support both computational and data grid and it can run several jobs on the resources with different middleware at the same time. The broker can operate in a number of different hardware and software configurations. The prerequisites on the broker side are Java and the prerequisites for computing resource that is Alchemi 1.0 in the remote side. The requirements for installing the grid environments are .Net framework and SQL Server 2000 for manager machine (grid environment). The grid environment, is created with the .Net framework 1.1 and SQL Server is installed in a node with Alchemi Manager to act as a Manager. After configuring the Alchemi manager, the Alchemi Executor is installed and configured for n-nodes to support a grid environment to act as a sub-node for manager as shown in figure 6. All executors connected to the Alchemi Manager is as dedicated or non-dedicated.

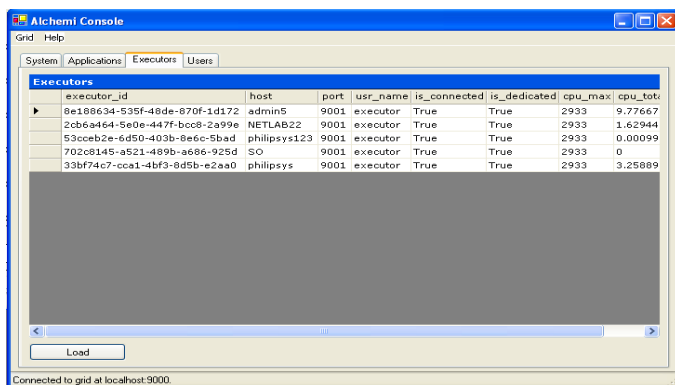


Figure 6. Alchemi Console with executors

The grid bus broker using the DISCOVER algorithm finds the suitable resources and submits job to the grid environment. The Alchemi manager sends the job to the executor to complete the task. Depends upon the job the power usage of the node will vary, figure 7 shows the CPU power availability and usage.

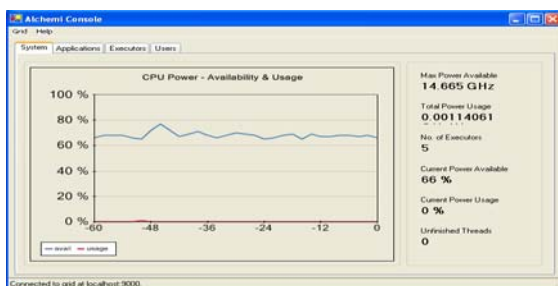


Figure 7. CPU Power – Availability and Usage

The eXtensible Parametric Modelling Language (XPML) is an XML-based language, which is used by broker to create parametric applications. The XPML supports description of parameter sweep application execution model in which the same application is run for different values of input parameters often expressed as ranges [9]. Another XML-based Grid Resource Language is a service description file that can

be used by the broker and the properties of their services has to be described. They are different services are described in the grid environment like compute service, data service, information service and application service. JSDL is a Job Submission Description Language used to describe the requirement of computational job for submission to resources. In the xml based JSDL job description consists of the Job identification, Application, Resources and Data Staging information. The job is submitted to the alchemi manager through grid bus broker workbench.

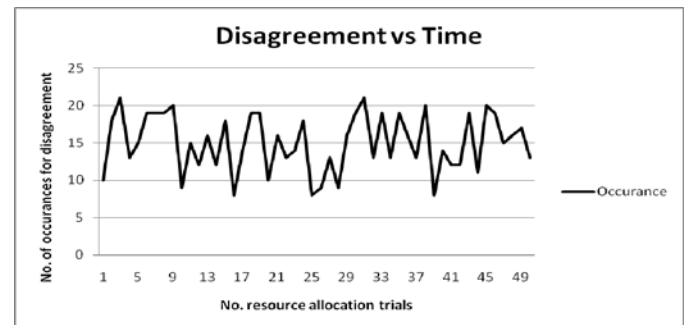


Figure 8. Disagreement Vs Time

Let us consider 50 samples for a grid session time T and for each sample, 50 trails of tasks are submitted. In each and every trail there are 'n' number of disagreements for 'N' number of job submitted. The disagreement ratio can be calculated from the number of disagreements and the total number of jobs submitted as shown in equation (1). If there is any disagreement then it can be solved by the dispute solver based upon the severity, process and criticality of the task. The number of occurrences for disagreement vs time is as show in figure 8. The dispute solver can restart or resubmit or cancel the job. In this disagreement mode, a negotiation process can also be initiated and if the negotiation fails then through the agreement, jobs can be resubmitted as per the affidavit.

VI. CONCLUSION

The resource discovery model under disputes is proposed using grid bus broker and dispute solving mechanism in a grid environment. The DISCOVER algorithm helps to minimize the time taken to separate or detach the disputed resources and the tasks from the grid and solve the conflict through an affidavit process of agreement. The proposed XRCA technique with its template will strengthen the process of agreement and negotiation in run time since it is based on the specification accepted by the user as well as the grid broker. The evaluations of the proposed DISCOVER and XRCA technique is experimented and the performance metrics are derived. The work is having a limitation on the number of managers and executors due to load balancing counter measure that is arising if the number disagreement is high. The future work focuses on the identification faulty resources and incorporation of proactive mechanism to alert the auditor and the manager before resource dispute occurs.

REFERENCES

- [1] G.Murugasan, C.Chellapan, "An Economic-based Resource Management and Scheduling for Grid Computing Applications" International Journal of Computer Science Issues, Volume 7, Issue 2, pp 20-25, March 2010
- [2] Hojjat Baghban, Amir Masoud Rahmani, "A Heuristic on Job Scheduling in Grid Computing Environment," Seventh International Conference on Grid and Cooperative Computing, pp.141-146, 2008
- [3] Jing Hu, Mingchu Li, Weifeng Sun, Yuanfang Chen, "An Ant Colony Optimization for Grid Task Scheduling with Multiple QoS Dimensions," Eighth International Conference on Grid and Cooperative Computing, pp.415-419, 2009
- [4] Yiqun Zhu; Minglu Li; Chuliang Weng; , "Ant Algorithm with Execution Quality Based Prediction in Grid Scheduling," *ChinaGrid Annual Conference, 2009. ChinaGrid '09. Fourth*, pp.77-83, 21-22 Aug. 2009
- [5] Cho-Chin Lin; Chun-Wei Shih; , "An efficient scheduling algorithm for grid computing with periodical resource reallocation," *Computer and Information Technology, 2008. CIT 2008. 8th IEEE International Conference on* , pp.295-300, 8-11 July 2008
- [6] Cardosa, M.; Chandra, A.; , "Resource Bundles: Using Aggregation for Statistical Wide-Area Resource Discovery and Allocation," *Distributed Computing Systems, 2008. ICDCS '08. The 28th International Conference on* , pp.760-768, 17-20 June 2008
- [7] Massimiliano Caramia, Stefano Giordani,"Resource allocation in grid computing: an economic model", WSEAS Transactions on Computer Research, Volume 3, Issue 1, ISSN:1991-8755, pp 19-27, Jan 2008
- [8] Jong-Kook Kim, Sameer Shivle, Howard Jay Siegel, Anthony A. Maciejewski, Tracy D. Braun, Myron Schneider, Sonja Tideman, Ramakrishna Chitta, Raheleh B. Dilmaghani, Rohit Joshi, Aditya Kaul, Ashish Sharma, Siddhartha Sripada, Praveen Vangari, Siva Sankar Yellampalli, "Dynamically mapping tasks with priorities and multiple deadlines in a heterogeneous environment Source" Journal of Parallel and Distributed Computing, Volume 67 , Issue 2, pp 154-169, February 2007
- [9] <http://www.cloudbus.org/broker/>