Volume 3, Issue 8, August 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Integrate Mobile Devices with Grid by Using Efficient Local Resource Scheduling

Dr. Akash Saxena
S.J Engineering College, CSE
India

Khushboo Saxena
TIT Bhopal, IT Dept
India

Aakanksha Saxena
Suresh Gyan Vihar University, CSE Dept
India

Abstract*— Grid computing has emerged as an important field, distinguished from conventional distributed computing by its focus on large-scale resource sharing, innovative applications, and in some cases, high-performance orientation. Grid computing is defined as "flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources". In the road of making mobile devices as an entity in grid computing technology, and to bridge the gap between mobile world and grid computing world we are using Layered Architecture, which divides the complexities existed in integration, among different layered components. We give our contribution details, in the implementation of the proposed Layered Architecture. In the implementation we considered JAVA enabled mobile devices, GT (Globus Toolkit) based grid services. In Layer-2 of the architecture we need to have an efficient scheduling algorithm which addresses the problem of local scheduling. We are using ABC algorithm to address local resource scheduling problem. With this we try to use efficient use of resources using mobile devices

***Keywords*— Globus Toolkit, Grid Computing, Mobile Devices, ABC algorithm, Java.**

I. Introduction

Grid is an environment that provides the ability to share and transparently access resources across a distributed and heterogeneous environment not only requires the technology to virtualizes certain resources, but also technologies and standards in the areas of scheduling, security, accounting, systems management, and so on. Generally, Grid system combines with the resources that are more powerful, diverse and better connected as compare to desktop on the Internet. Grid Infrastructure [1] contains a wireless PDA or mobile phone persisting different roles. All over access of small wireless devices is the most congenital way to provide information such as data consumers. As such they provide some applications such as networked application or Web browsers are handy to traditional access media provided with alternatives and extensions.

The biggest limitation in Grid Computing technology which is forcing the people to stand away from this technology is accessibility to grid service is not within the reach of common people. By providing access to grid from a simple mobile device a common man can get benefit of this technology. So the integration of mobile and grid provides new opportunities, along with introduction of new challenges. Introduction of mobile devices in a distributed computing technology like grid computing requires an extension of the heterogeneity model [2]. We need to consider the fact that grid may have many different kind of technologies which needs to be incorporated into grid nicely. Due to the mobility problem the network is unstable in wireless domain, disconnections and the resulting network partitions should be handled efficiently.. However, with mobile or wearable devices, these events need to model in a general way. A mapping scheme should decide how different components of an application should be mapped, i.e., which components should be assigned to the back-end and which to the front-end (mobile device). These decisions should consider some issues like power consumption, locality for access, resources and throughput requirement.

The existing limitations in mobile devices make it beneficiary from the Grid. The integration of grid and mobile devices advantageous for mobile to mobile and mobile to desktop collaboration for resource sharing and provide better performance to users. A grid-based mobile environment would allow mobile devices to become more efficient by offloading resource-demanding work to more powerful devices or computers.

Mobile OGSI.NET project [3] is an implementation of the OGSI.NET for mobile devices. It is built on top of Microsoft .NET Compact Framework and based on the Pocket PC 2003 or Windows Mobile operating system. In [4] Yi and Livny proposed architecture for mobile Condor access, they analyzed the constraints imposed by mobile platforms when connected to Condor. In [5] Millard, Arouna presented the potential advantages of using the Grid for mobile e-learning, described their experiences with implementing a mobile e-learning Grid client using current Grid technologies. Miika Tuisku [1] surveyed the current technologies and standards, mainly concentrating on MIDP (Mobile Information Device Profile) devices and Java (J2ME) interfaces.

The scheduling can be categorized as, global and local scheduling. Various global scheduling algorithms have been operational such as data-centric scheduling [6], high-throughput scheduling [7], application-level scheduling [8], and economy-based scheduling [9]. After completion the job of global scheduler, it submitted to the local resource managers like Condor [7], LSF [10], and PBS [11]. In Layer2 of the system model we need to have an efficient local resource scheduling algorithm which tries to optimize the resource utilization and improves the global performance of the system. We proposed one such algorithm. In our algorithm we mainly considered unreliable connectivity problem of the mobile devices, battery power issues, economy issues. The rest of paper described in following sections: section 2 discusses the System Architecture; section 3 describes the implementation details of our proposed System Architecture. Section 3.1 describes in detail its components features and implementation details. Section 4 presents ABC algorithm for efficient resource scheduling which is a core component in Layer-2 of the system Architecture. In Section 5 presents conclusion.

II. SYSTEM ARCHITECTURE

The System Architecture is a Layered type (depicted in Figure 1). Due to the layered approach changes made in one layer not necessarily reflects the other layers functionality and implementations, upper layer entities will provide services to the lower layer entities, and lower layer entities will depend on upper layer services. Due to layered approach different layers can be developed independently. In Layer-1 we will have mobile devices like PDA's, mobile phones, laptops etc. Layer 2 is introduced to address the requirements of various mobile technologies. These components should have features like encapsulating heterogeneity, able to communicate with standard protocols and languages like HTTP and XML, able to handle lot many requests. In the context when mobile devices are acting as processing elements rather than just grid users, these entities required to have feature like dividing the work, assigning subtasks among the mobile devices, accumulating the processing results from different mobile devices before sending back the result. Layer2 component should have an efficient local resource scheduling algorithm. The algorithm should be able to select a set of mobile devices among the pool under its coordination. Moreover it we apply ABC algorithm for resource sharing and compare the performance of traditional and new ABC algorithm. In the coming sections we will discuss both algorithms. These components should be collocated with an *access point*. It acts as a gateway to the mobile world.

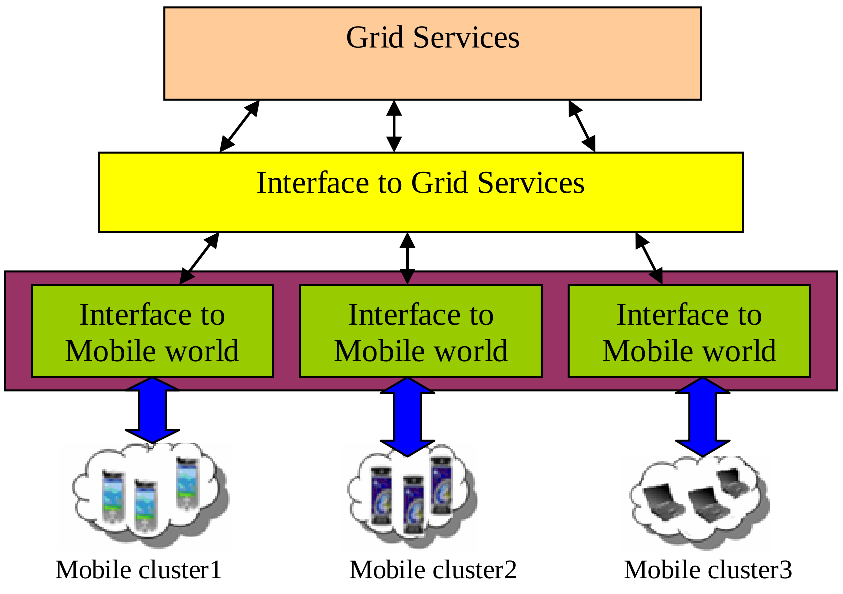
Grid Services
Interface to Grid Services
Interface to Mobile world
Interface to Mobile world
Interface to Mobile world
Mobile cluster1
Mobile cluster2
Mobile cluster3

Figure 1 System Architecture

Layer-1:
In the lower layer we will have mobile devices like PDA's, mobile phones, laptops etc. most of the mobile devices are equipped with internet features. With proper interface chord a mobile device can start communicating to the entities in the internet world.

Layer-2:
We might have mobile devices of different technologies to address this we introduced Layer2.
Layer2 component should have features like encapsulating heterogeneity, able to communicate with standard protocols and languages, able to handle lot many requests. In the context when mobile devices are acting as processing elements rather than just grid users, Layer2 entities required to have feature like dividing the work, assigning subtasks among the mobile devices, accumulating the processing results from different mobile devices before sending back the result. Due to the accumulation feature the number of messages transmitted will be reduced.

Layer3:
Due to the introduction of Layer-3 we can use different technologies to develop grid independent of Layer2 entities. Layer3 acts as interface between grid services and mobile world interfaces. It will take the parameters from lower layers based on those parameters it should locate the appropriate service and invoke the proper operation on that service. This interface should also have the ability to invoke grid services which might have been implemented in various technologies. We also need to have standards based on which the entities in Layer2 and Layer3 can communicate.

Common features include encapsulating grid services heterogeneity, understanding the parameters passed from lower layers. Here again the XML would come into picture to overcome the data representation problem between, Layer 2 and Layer 3 components.

Layer4:
Layer4 will have grid services which exploits idle resources which are spread across the internet. These services can be implemented using different technologies like condor, GT (Globus Toolkit), unicore, legion, jxta etc. Layer4 entities will serve the requests passed from its lower layers and sends back the results.

III. IMPLEMENTATION DETAILS OF OUR SYSTEM ARCHITECTURE

In the framework shown in Figure 2 we depicted our attempt, to partial implementation of the System Architecture. We demonstrated the proposed model in the context of JAVA based mobile devices and GT (Globus Toolkit) based grid services. The proposed implementation can be broadly devised into five components. Different components will serve purpose of different layers entities. Though we consider only JAVA enable mobiles, our framework is easily extensible onto other technologies.

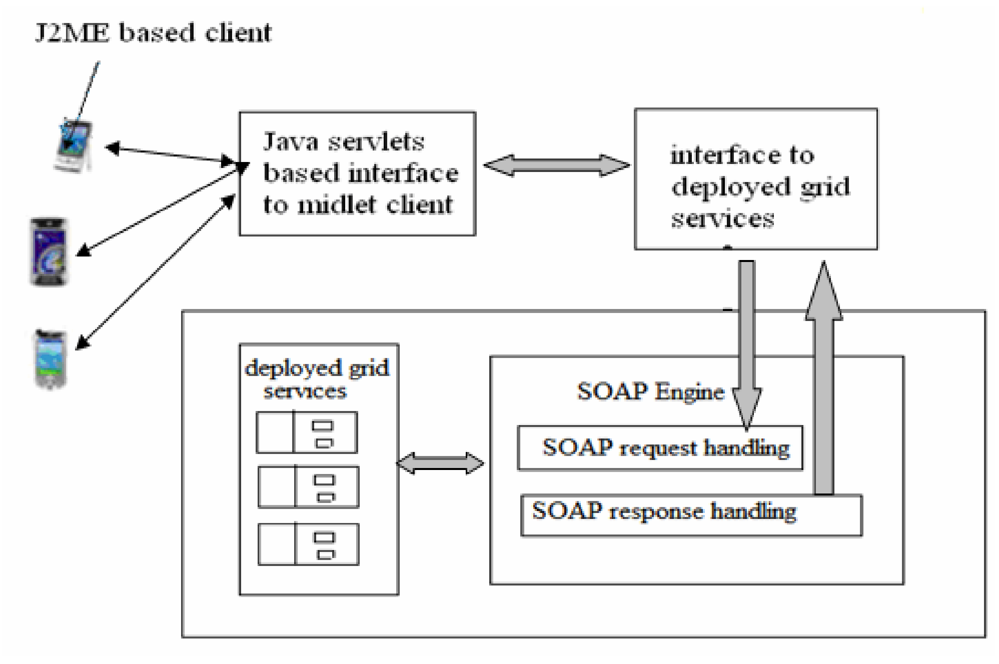

Figure 2 Architecture for Interface to GT based Grid Services through JAVA enabled mobile devices

J2ME midlet client. This component serves the purpose of Layer1 entities to some extent. It is a J2ME Midlet. It has the ability to provide nice GUI to JAVA enabled mobile devices. The mobile user will enter the data on the Midlet GUI which acts as parameters to the WSRF component functionality those data will be transmitted over the air to the appropriate servlet running on web server. In fact J2ME GUI feature used just to illustrate our implementation. Anyone who wants to use our framework can issue a request to this component to perform a particular task on the grid. Once the request issued the task invocation will be done automatically. It is basically a Midlet which is written in J2ME, it is able to create HTTP connections to an HttpServlet. This servlet sits in a HTTP web server (e.g. .apache tomcat) and serves to HTTP requests. We have used WTK2.5.2_01 (Wireless Toolkit2.5.2_01) emulator for testing our J2ME client component.

JAVA servlet based interface to midlet client. It consists of JAVA servlet. It will pack the parameters received from midlet client component and passes it to deployed grid services interface component. This module belongs to Layer2 of the System model. This component is Java based that runs in a Web server. A servlet is a Java class and it executed in Java Virtual Machine (JVM) by a service known as servlet engine. The servlet engine loads the servlet class the first time that it is requests, or optionally already when the servlet engine is started. It handles multiple requests till it will not explicitly unloaded or servlet engine got shut down.

Interface to Deployed grid services. This component meets requirements of Layer3. It will take parameters from midlet interface component using those it will invoke the appropriate methods in the specified grid service. It will return the results back to the midlet interface module. In case of error it will return the appropriate error code as a result. It is a JAVA based component. It also forwards the result of a task processed by mobile devices to some client in Internet. We took GT (Globus Toolkit) based WSRF client software and modified the code according to current requirement. We made this component to communicate both with midlet interface component and grid service component.

SOAP Engine. It is software, which deals SOAP message processing issues. It is a part of Layer3. For SOAP messaging is handled by SOAP engine. A web service is deployed into an Axis message processing node using a deployment descriptor known as a Web Service Deployment Descriptor (WSDD). The WSDD describes how the various components installed in the Axis node are combined together to process incoming and outgoing messages to the service. We have used the apache tomcat SOAP processing capabilities. Apache Axis is a standard implementation of SOAP engine. This component can understand SOAP request/response messages.

Deployed Grid Services. We have developed these services using GT. It belongs to Layer4 of the proposed model. These are some toy web services like counter service whose value can be accessed and modified by invoking operations like add, subtract, get value etc, a factory service which creates service instance, a resource lifetime illustration service which illustrates features like resource destroy, scheduling resource termination time etc. These services basically illustrates WSRF specifications like WS-Resource properties, WS-Resource lifetime, WS-Base faults, WS-Resource addressing. They can exploit the available resources on the grid in an efficient, secured manner. It utilizes the GT inbuilt features like resource management, execution, and data management, fault detection and security. We could also enhance the described framework to accommodate various other technologies by developing appropriate components in different layers.

J2ME Phone Access. We have used Sun Java Wireless Toolkit 2.5.2_01 for CLDC **(formerly known as J2ME Wireless Toolkit)** emulator for testing our J2ME client. They show how our J2ME client software would work on JAVA enabled devices in accessing the WSRF component features of GT. We have deployed the GT based grid services in apache tomcat web server using Globus service build tool. Features like Login Screen, Options Screen, Create Service instance, Update Property, Get Property, Immediate Destroy, and Scheduled Termination of a resource etc. We have also implemented features like getting multiple properties, updating, and inserting/deleting property into/from a resource property document.

We have developed simple GT (Globus Toolkit) based toy grid services and deployed in Apache tomcat web server. Toy services developed by us will illustrate WSRF specifications like WS-Addressing, WS-Resource WS- Resource Lifetime, WS- Resource Properties, WS-Notification, and WS- Base faults. Using JAVA, WSRF client software technologies we have developed interfaces required to connect midlet clients, grid services. We have developed GUI for Java mobiles, using that GUI a mobile user can issue a request and access the grid services.

In our framework we only concentrated on JAVA enabled devices since mobile space dominated by JAVA most of the mobile device manufacturers are bringing JAVA supported devices into market. A WSRF (Web Services Resource Framework) specification is a core component in GT (Globus Toolkit). It includes resource lifetime, resource properties, notifications, addressing specifications. The entire GT (Globus Toolkit) has been built on top of the WSRF component of GT (Globus Toolkit). So by giving the interface to WSRF component through JAVA enabled mobiles, we can easily extend the interface to remaining components of the GT(Globus Toolkit). Java Platform, Micro Edition (Java ME)[12] is a combination of technologies and specifications designed to create a platform for devices, from mobile phones to consumer products to embedded devices.. Here we are giving interface to WSRF component features of GT (Globus Toolkit) through mobile devices.

IV. JOB SCHEDULING ALGORITHMS

Layer-2 component of our proposed architecture should have an efficient local resource scheduling algorithm. The designing of local resource scheduling algorithm involves issues like energy sensitivity and weak security etc. Among the issues we concentrate on unreliable connectivity of mobile environment, battery power of the devices, cost to be paid for using the device. Job scheduling is perform based on the expected execution time on various mobile nodes. The job in the queue is assigned to the mobile node with related input data, when it finds any mobile node free or available. The output of the job is returned to mobile device interface, after completion of job execution at mobile node. Then mobile node is assigned to next job in the queue. Figure 3 shows the job execution process which is performed on the mobile node. For data transfer either input or output, mobile nodes should be in connected mode. If the mobile node is disconnected then it will wait for mobile node to be connecting again. Mobile nodes state transition diagram is shown in figure 4. State C and D denote the connection state (C) and disconnection state (D), respectively, *d* is the disconnection rate, and *c* is the reconnection rate of the mobile node, which are governed by Poisson process.

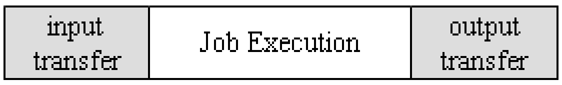

Figure 3 Job Execution Process

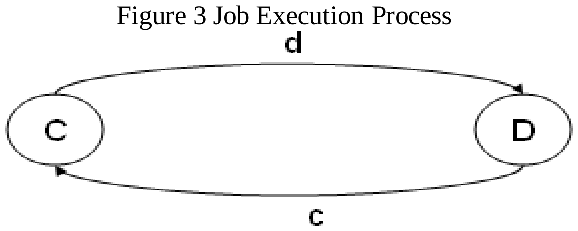

Figure 4 State transition Diagram

When a number of jobs arrive at the JQS (Job Queuing Server) of mobile device interface, or a number of workloads are decomposed from a single job, the JQS selects certain mobile nodes. In a static grid environment, we achieve improving performance as the number of participating computing nodes grows. However, the rule changes when the mobile devices participate in the computing. Links of some mobile nodes are unreliable, that is, they are likely to be in the disconnection state for a long time while others are not. The unreliable mobile nodes may decrease the overall job execution performance because the JQS should wait for the reconnection of any disconnected node in order to complete the jobs. We should determine whether it is better to include a mobile node in the computing or not. We might get different options of mobile devices sets, which reduce response time. We will try to increase the average battery power and also decrease the average cost of the selected devices.

Existing Scheduling Algorithm:

The algorithm to select the participating nodes in the computing is as follows:

```
Input:
N: number of nodes. Nodes list R=<$n_1$, $n_2$, …., $n_N$>
Each $n_i$ have properties resource ID, connection rate, disconnection rate, usage cost, battery power, time to transfer input data, time to transfer output.
T: task to be scheduled has some execution time.
Output: finding a subset of resources among N devices to execute task T.
    1. find the expected response time $g_i$ of all mobile nodes using equation 1
    2. Sort the mobile nodes in increasing order according to the response time.
       Sorted list:   SR=< $n_a$, $n_b$… $n_j$…>
    3. find the accumulated average battery power for each node $n_i$ in SR
       $b_i$=($n_1$.batterypower+…+ $n_i$.batterypower)/i
    4. find the accumulated average cost for each node $n_i$ in SR
          $c_i$=($n_1$.usagecost+…+ $n_i$.usagecost)/i
    5. for (i=N;i>1;i--)
       {w= $W_{total}$/(i-1)- $W_{total}$/i) ; g=$g_i$ - $g_{i-1;}$
                if( w-g>0)
                  break; }
    6. remove nodes 1 to i/2 and nodes i+1 to N from list
    7. remove the nodes whose w-g value less than 0
    8. Sort the remaining nodes of the list in increasing order based on pre calculated accumulated average battery power ($b_i$)
    9. find the index of the node with maximum battery power i.e 'k'(last node of the list)
    10. remove nodes 1 to k/2 from list
    11. sort the remaining nodes in decreasing order based on pre calculated accumulated average cost ($c_i$)
    12. find the index of node with minimum average cost i.e the index of the last node let it be j
    13. select the nodes $n_a$, $n_b$,……, $n_j$ for execution
    14. Distribute the subtasks of T onto the selected nodes.
```

Artificial Bee Colony Algorithm

Artificial Bee Colony (ABC) is a relatively new member of swarm intelligence. ABC tries to model natural behaviour of real honey bees in food foraging. Employee bees, onlooker bees and scouts bees are members of colony of artificial bees. The number of the employed bees or the onlooker bees is equivalent to the number of solutions. An employed bee produces changes on its solution depends on the local information and test the fitness of the new solution.

The elementary idea of ABC is to simulate the foraging behaviours of bee colonies. Social insect colonies can be considered as dynamical system gathering information from environment and adjusting its behaviour in accordance to it. While gathering information and adjustment processes, individual insects don't perform all the tasks because of their specializations. Generally, all social insect colonies behave according to their own division labours related to their morphology [13].

Algorithm:

Input: N number of nodes.

1. Initialization of algorithm: These are two variables by which it evaluates the fitness of the different nodes searching for resource.
2. Construction of bees: Evaluate employed bees are by fitness value. Fitness is value to select the working nodes searching for resources.
3. Single Shift Neighbourhood: Apply Single Shift Neighbourhood for each employed bee according fitness value of bees. This operation is performed over a matrix which contains the distance of individual nodes, remove nodes those not fit.
4. Double Shift Neighbourhood: Apply Double Shift Neighbourhood for each employed bee based on fitness value of bees. Find the best fit nodes and apply the double shift to achieve best selection for the resources.
5. Solution Construction: Calculate probability and assign resource according to probability.

6. Ejection chain Neighbourhood: Apply Ejection Chain Neighbourhood for each employed bee based on probability value of the node.
7. Terminal test: Select the best fitted optimal solution from the available solutions.

V. Conclusions

Grid computing is the latest technology which is still under the research in academia. It has not gained enough significance in the enterprise world yet. In the direction of letting the common people access to grid service we have used a System Architecture. We use ABC algorithm as local scheduling algorithm. We primarily focused on issues like intermittent disconnection problem of mobile devices, battery power details, economy issues. Moreover it requires to check the authenticity by comparing the with the ABC resource scheduling algorithm. We also simulated our proposed algorithm and studied the characteristics under different environments. So as to create an environment where the mobile devices and traditional networked desktops are likewise entities we have to extend the proposed System Architecture implementation for all kinds of mobile devices like PDAs, Laptops, Cell Phones, sensors etc. We should give innovative approaches to encourage the mobile gadget users, to keep their gadgets into grid. The job scheduling algorithm could be improved more considering issues like devices heterogeneity, security issues etc. We need to have a unified model to value the resources. We have to study, how to motivate resource owners to contribute their resources to the Grid.

References

[1] Miika Tuisku, “Wireless Java-enabled MIDP devices as peers in Grid infrastructure”, Annual cross grid project work shop &1st European acrossgrid conference, 2003.

[2] M. Migliardi, M. Maheswaran, B. Maniymaran, P Card, and F Azzedin, “Mobile Interfaces to Computational Data and Service Grid Systems”, ACM SIGMOBILE Mobile Computing and Communications Review, Oct 2002, vol.6, Issue4.

[3] D. Chu, M. Humphrey, “Mobile OGSI.NET: Grid Computing on Mobile Devices”, 5th IEEE/ACM International Workshop on Grid Computing - GRID2004, Pittsburgh, PA, Nov 8 2004.

[4] M.Livny, and S.Yi, “Extending the Condor Distributed System for Mobile Clients”, ACM Mobile computing and Communications Review, 1999, vol. 3(1), pp. 38-46.

[5] Feng Tao, Arouna Woukeu, Hugh C. Davis, and David E. Millard, “Experiences with Writing Grid Clients for Mobile devices”. 1st International ELeGI Conference on Advanced Technology for Enhanced Learning, Vico Equense (Naples), Italy, March 2005.

[6] Sang-Min Park and Jai-Hoon Kim. "Chameleon: A Resource Scheduler in a Data Grid Environment," 2003 IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'2003), Tokyo, Japan, May 2003.

[7] The Condor Project. http://www.cs.wisc.edu/condor .

[8] F. Berman and R. Wolski. “The AppLes project: A status report,” Proceedings of the 8th NEC Research Symposium, Berlin, Germany, May 1997.

[9] D. Abramson, J. Giddy, I. Foster, and L. Kotler. “High Performance Parametric Modeling with Nimrod/G: Killer Application for the Global Grid?” In Proceedings of the International Parallel and Distributed Processing Symposium, May 2000.

[10] LSF. http://www.platform.com/products/LSF/.

[11] OpenPBS. http://www.openpbs.org.

[12] Oracle Java for ME. http://www.oracle.com/us/technologies/java/mobile/overview/index.html

[13] Dr. K. Vivekanandan, D. Ramyachitra, B. Anbu “Artificial Bee Colony Algorithm For Grid Scheduling”, 2011 Journal of Convergence Information Technology, Volume6, Number7, July 2011