ITU-GRAM+: A WEB Based Grid Middleware

G. Bengi TURKEL
Department of Computer Engineering
Istanbul Technical University
Istanbul, Turkey
bengi.sendur@avea.com.tr

D. Turgay ALTILAR
Department of Computer Engineering
Istanbul Technical University
Istanbul, Turkey
altilar@itu.edu.tr

Abstract— **This paper describes the Grid middleware (ITU-GRAM+) which is implemented by using only open source software based on trend web technologies. The ITU-GRAM+ allows users to access computational and data resources, submit their jobs and track job status via standard interfaces. Main goal of ITU-GRAM+ is mapping the job resource requests to the resources in a way that will satisfy both the application users and resource owners. In order to satisfy this requirement ITU-GRAM+ is designed to have Resource Discovery, Resource Allocation, Job Submission, Job Execution and Job Monitoring components. Some of these components in ITU-GRAM+ work with SOA architecture to facilitate a communication between resources and resource manager which also gain flexibility to the system in terms of adding new resources and new users. For this communication ITU-GRAM+ exposes different web services to run both on resources and resource manager. In this paper all details about components are described with their additional functionality comparing to existing middleware.**

Keywords-Grid Computing, Grid Middleware, Resource Manager, Web Service

I. INTRODUCTION

Grid middleware are large software which provide a set of basic functionalities, each one implemented by a separate component. Such functionalities include data storage, authentication and authorization, resource monitoring, resource management and job management.[1]

One of the most challenging areas of Grid application development is the construction of portals to allow computational scientists, researchers and high performance computer/application users to access the resources via an easy to use web page interface. The aim is to develop common components that can be used by portal developers to allow them to build a web application that can securely authenticate users to remote resources and help them make better decisions for scheduling jobs by allowing them to view pertinent resource information.

In this paper, a new Grid Middleware (ITU-GRAM+) system that contains these components is designed and implemented using standard interfaces and protocols through new technologies which are open source to all people. The ITU-GRAM+ responds both data and computational intensive requirements of users. In order to meet these requirements in such environment, Service Oriented Architecture (SOA) principles have gained great importance. Service-Oriented Architecture (SOA) is a set of principles and methodologies for designing and developing software in the form of interoperable services [2]. For ITU-GRAM+, SOA is the main architecture to provide a communication between resources. Due to allow users to reach these resources, a web application is designed and developed. Getting job specific information by users and allow them to reserve and allocate the resources according to these requirements is main goal of this web application. But resources in grid system are not only belongs to system, they have to maintain their daily work issues. For this reason, to keep update information about resources in such system, Grid Information System is implemented.[3] The aim of this component to retrieve resource information such as *available cpu, available storage, available memory* which could be changed dynamically. The challenging question is that how the system informs the resources about releasing job after reservation of resource is completed. The Job Execution Component is developed to send job information to resources via web services which is exposed by resource. So far the components of ITU-GRAM+ are described, now we will have a look state of todays grid middleware.

Currently, the Globus Toolkit is the de facto standard for grid computing because of its wide acceptance and deployment worldwide, even though several alternatives do exist, like Legion and Condor[4] . Web services-based GT4 which is one of the latest deliveries of Globus team provides significant improvements over previous releases in terms of robustness, performance, usability, documentation, standards compliance and functionality. A set of service implementations provide useful infrastructure services. These services address such concerns as execution management (GRAM), data access and movement (GridFTP , RFT, OGSA-DAI), replica management (RLS , DRS), monitoring and discovery (Index,Trigger, WebMDS), credential management (MyProxy, Delegation ,SimpleCA), and instrument management (GTCP). Most are Java Web services but some are implemented in other languages and use other protocols. Grid Resource Allocation and Management (GRAM) service addresses these issues, providing a Web services interface for initiating, monitoring, and managing the execution of arbitrary computations on remote computers.[5] During ITU-GRAM+ design phase, all Globus components are investigated and analyzed. Most of Globus components have corresponding component in ITU-GRAM+ such as GRAM, GridFTP , Index that’s why we called the with + suffix. It have additional components plus to GRAM.

Condor is the one of alternative resource management system which is designed to support high-throughput computations by discovering idle resources on a network and allocating those resources to application tasks. Legion is the other alternative to existing resource management system which is a reflective, object-based operating system for the Grid. It offers the infrastructure for grid computing. Legion provides a framework for scheduling which can accommodate different placement strategies for different classes of applications.

Limitations of the existing Grid middleware which does not take into account the needs of everyday scientific and business users. Day-to-day computer users and small to medium sized organizations often do not use clusters, and thus for them setting up Grids using the existing middleware is complex [6]. Furthermore, Grid enabling their applications is nearly impossible, as they are not easily supported, and this poses a massive barrier to the pervasive adoption of Grid computing by these communities. Grid Middleware is also complex to setup and necessitates a steep learning curve. But ITU-GRAM+ has basic interface to use and it is not necessary to adapt your application for grid system. You can submit your jobs as it is.

The paper is organized as fallowing.Section2 will describe the reasons that motivate me to do this work.. Section 3 describes main functionalities of ITU-GRAM+ and details of the components are explained in Section 4. Last section includes future works and conclusion.

II. MOTIVATION

Many Grid Portals can provide a customizable interface allowing scientists and researchers to perform Grid operations such as remote submission of their own programs, staging input and output files, and querying resources and queues information. Some of them is widely used by most of people and contains all necessary components to support grid infrastructure. According to my best knowledge there is no such a grid middleware is implemented is only using open source software .From the perspective of developers , it is easy to create their own middleware which provides basic infrastructure to add new functionalities and improve the system themselves. From the perspective of day-to-day computer users, scientists and business users it is easy to start work on it without having computational knowledge. These are the key points of my work to motivate me.

III. ITU-GRAM+ OVERVIEW

ITU-GRAM+ main functionalities are Resource Discovery, Resource Allocation, Job Submission, Job Execution and Job Monitoring. In this section it will be given brief description of these components respectively.

Computational or data-intensive applications use the resources according to the requests from the user in order to achieve results quickly and efficiently. The resource management system must take into account not only computational resources such as the amount of available CPU, memory, data storage capacity but also starting time of job, the financial value of the resource and the efficiency of the resource in order to find these resources. Source information on the resource can change dynamically, so to make the right choices these information should be kept constantly up to date. Grid Information Service component is implemented on ITU-GRAM+ using web-service and batch job technologies in order to satisfy this update requirement of grid nature.

Job management component is used to submit, cancel and monitor jobs for execution on available resources. Users can submit their jobs using job description language (jdl) which is a high-level, user oriented language [7]. It has lot of information about content of job. It is based on XML structure which is a standard to exchange data between systems. After submitting jobs, it is possible to monitor job steps. The resource will update the grid resource manager on critical points of the process with unique id which is provided by Grid Resource Manager in the beginning of job submission.

Resource management is used to allocate suitable resources to jobs. As in the economy, if we think that there are unlimited demands and limited number of resources, scheduling of resources is the one of the challenge problem on Grid systems. In this work resources selection is done not only the physical properties of the resources but also the distances from each other is considered. At the same time this distance varies depending on the type of job is relatively. All of these components will be analyzed one by one in next section.

IV. ITU-GRAM+ ARCHITECTURE AND IMPLEMENTATION

The above components which are given a brief description will be more detailed in this section. In Figure1 describes the architecture of ITU-GRAM+. ITU-GRAM+ includes a web application which allow users to use a standard interfaces in order to submit their jobs. The application container is the Tomcat for both web services and web application. Apache Tomcat is an open source webserver and servlet container developed by the Apache Software Foundation [8] .Java is the language used to develop these components. Java is currently one of the most popular programming languages in use, particularly for client-server web applications [9]. MySQL officially, but also commonly the world's most used relational and open source database management system [10]. ITU-GRAM+ choose the MYSQL to store all the information about users, resources, jobs and reservations. Hibernate is one of the free software that facilitated the storage and retrieval of Java domain objects via Object/Relational Mapping [11]. Java Server Pages (JSP) is a technology that helps software developers to serve dynamically generated web pages based on HTML , XML, or other document types [12]. Apache Struts 2 is an elegant, extensible framework for creating enterprise-ready Java web applications [13]. Apache Axis2 is a Web Services / SOAP / WSDL engine, the successor to the widely used Apache Axis SOAP stack. [14]

Figure1 shows that all the relations between these technologies and which of them are used for which component of ITU-GRAM+. From the Grid Middleware point of view there are three different part of system. The Web Application which interacts with end user, the web service which is exposed to get update status of job and the batch jobs which are running at background to retrieve update resource information and submit jobs.

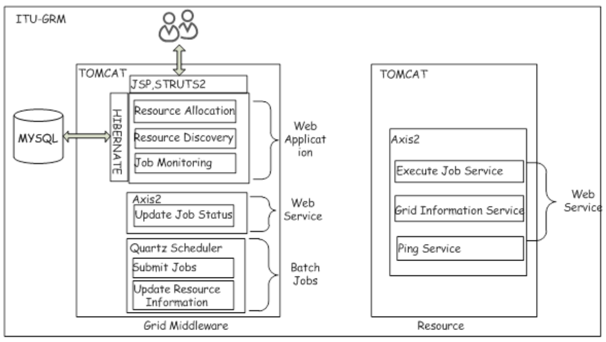


Figure 1. ITU-GRAM+ Architecture

From the resource point of view, only one web service is exposed includes three different methods inside.

Figure 2 shows a sample grid job life beginning from user submission to the completion of job in grid environment as an overview. The resources showed in the figure are small subset of in all resources. Client (user) gives the requirements inside job description language file (jdl) and user interface. Resource manager shows all available resource list and allows to user select one of them. System allocate resources during job execution time (Step1).A batch job waits for starting time of the job and invoke a web service to send job to computing resource (Step2). If the input file is stored in remote resource then it retrieves the file from File resources (Step3). After execution of job in computing resource, if output file is produced and if it is needed to keep in remote resources then the computing resource will send the file to storage resources.-(Step 4). The computing resource will inform resource manager for each step of job execution (Step5). Users could track their jobs via user interface (Step6).

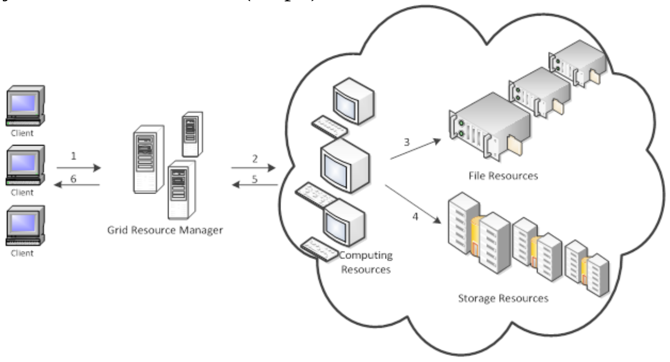


Figure 2.Grid Job Life

A. *Grid Information Service*

The resource information on Grid system can be changed dynamically because of grid nature. The nodes are not just a computational or data resource for Grid they also have their own work. So grid system should be aware of the changes on the resources. In order keep track these changes a web service is implemented which is running on the resources. The batch job on grid resource manager runs in a periodic time to retrieve the number of available processors, free memory and free disk space.

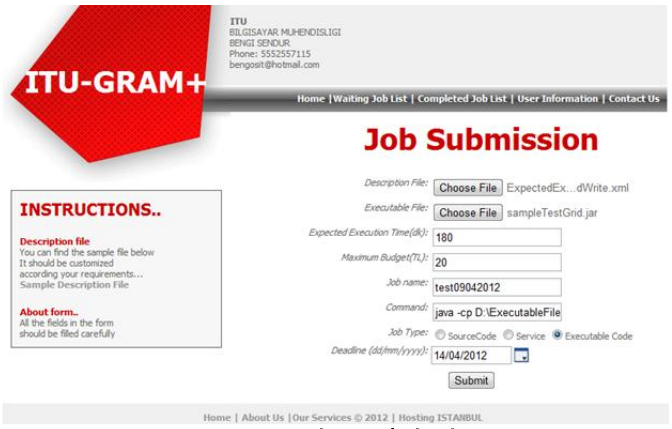
B. *Get User Requirements*

The users enter their requirements using both web user interface and job description language (jdl) file. The sample jdl file is showed in Figure 3. Some of them are clear to understand the meaning considering the tags but some of them are not. *gridType* could be *DataIntensive* or *Computing-Intensive* based on which resource type is needed more comparing to other types. *ioType* could be *Read* or *Write* based on input file size is greater or smaller than output file size relatively. Considering this comparison between output and input files, computing resource will be chosen to be closer to (as distances) file resource or storage resource in order to decrease the time during network transfer. *inputFileType* or *outputFileType* are the tags which are valuable to understand where input file will be found and where the output file will be kept. These tags values could be *Local*, *Remote* or *Expected*. *Local* means that input or output file exists on local system of user that submits jobs. *Remote* means that the user knows where input file could be found or output file could be kept and they are not on local. *Expected* means that user does not know where input file could be found and asks to grid system to find it. According to *gridType*, *ioType*, *inputFileType*, *output-FileType* values, some tags become mandatory. Based on combination of these tags the algorithm and resource types could differ.

As a result, user provides the details of the job in this sample xml file. Deadline, execution time of job, budget, execution file and other related information will be taken via user interface as shown in Figure 4.

```
<?xml version="1.0" encoding="UTF-8"?>
<RequirementList>
    <gridType>DataIntensive</gridType>
    <ioType>Read</ioType>
    <inputFileType>Expected File</inputFileType>
    <inputFileName>earthquake.txt</inputFileName>
    <inputfilePath>D:/Data/Inputs</inputfilePath>
    <remoteMachine>192.168.1.1</remoteMachine>
    <isWrited>No</isWrited>
    <os>windows</os>
    <cpu>1</cpu>
    <memory>3</memory>
    <storage>200</storage>
    <outputFileType>Local</outputFileType>
    <outputFileName>output.txt</outputFileName>
    <outputfilePath>D:/Data/Outputs</outputfilePath>
    <remoteOutputMachine>192.168.1.1</remoteOutputMachine>
</RequirementList>
```

Figure 3.JDL File


ITU-GRAM+

ITU
BILGISAYAR MUHENDISLIGI
BENGI SENDUR
Phone: 5552557115
bengosit@hotmail.com

Home | Waiting Job List | Completed Job List | User Information | Contact Us

Job Submission

INSTRUCTIONS..

Description file
You can find the sample file below
It should be customized
according your requirements...
Sample Description File

About form..
All the fields in the form
should be filled carefully

Description File: Choose File ExpectedEx...dWrite.xml
Executable File: Choose File sampleTestGrid.jar
Expected Execution Time(dk): 180
Maximum Budget(TL): 20
Job name: test09042012
Command: java -cp D:\ExecutableFile
Job Type: SourceCode Service [✓ Executable Code]
Deadline (dd/mm/yyyy): 14/04/2012
Submit

Home | About Us |Our Services © 2012 | Hosting ISTANBUL

Figure 4.Webpage of job submission

C. *List Available Resources*

As soon as the users pass their requirements to web application, the system checks firstly what kind of resources (such as computational, file or storage) are required and which of them meet the physical requirements of job. Expected execution time of job is important to find available interval time of resources. Figure 4 shows the one of sample scenario which contains the all resource types defined in ITU-GRAM+. For each resource type the system finds time interval which is blank and add to list. Each minute in time table is represented with “1” and “0”. For the example in Figure 5, the time table list is constructed as following.

Deadline-Job Submission Time=25 m

CR1={1,1,1,1,1,0,0,0,0,0,**0,0,0,0,0,**0,0,0,0,1,1,1,1,1,1,1}
SR1={0,0,1,1,1,1,1,0,0,0,**0,0,0,0,0**,0,1,1,1,1,1,1,1,1,1,1}
FR1={0,0,0,0,0,1,1,1,1,1,**0,0,0,0,0**,0,0,0,0,0,1,1,1,1,1,1}

List of ResourceList = { CR1SR1FR1, CR1SR2FR1,
CR1SR2FR2…. CRx SRy FRz }

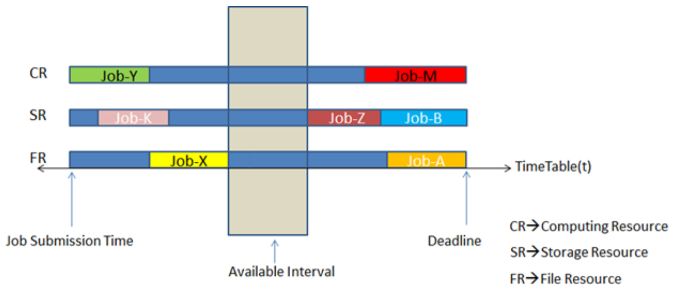
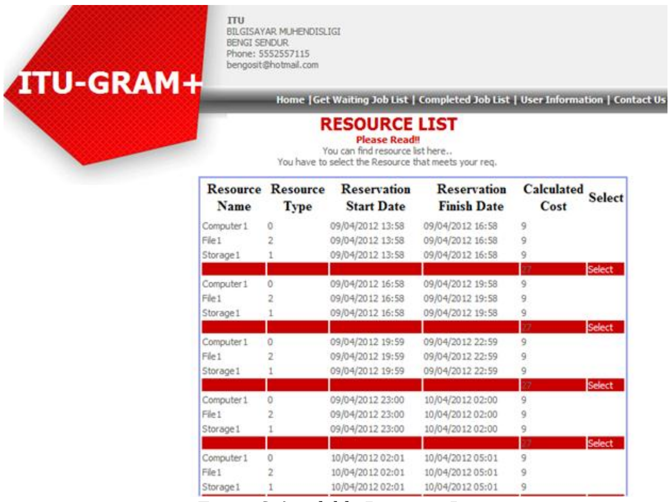


Figure5-List Available Intervals

But this list could be so much to show the user. So when the list is populated it will be filtered in terms of budget which is one of the input passing by user. Every resource has own cost which is calculated by the system based on their physical properties. A discount will be applied to calculated cost according to some conditions such as total load on the system during job execution or how long before the deadline the job is submitted or how many times the user submit a job to check loyalty of user. All these different situations cause a change on calculated cost. Despite of removing some triples from list based on budget there are still some triples are inefficient comparing the other triples. For example the input file of the job is larger than output file this means that CR should be closer to FR instead of SR in order to decrease time during transfer of the file through network.

If Distances(CR1-SR1) > Distances(CR1-SR2) then CR1SR1 should be removed from the list. The distances is not a physical distances between resources. As showed in Figure 1 every resources has *pingService* which takes the *ip* of other resource as an input in order to send a ping request and waiting for response. It calculates duration and send back to resource manager. It compares these results and filters triples which will not worth to select them. As a result, maximum 10 different options should be presented to the customer. A sample resource list is showed in figure 6.


ITU-GRAM+
Home | Get Waiting Job List | Completed Job List | User Information | Contact Us

ITU
BILGISAYAR MUHENDISLIGI
BENGI SENDUR
Phone: 5552557115
bengosit@hotmail.com

RESOURCE LIST
Please Read!!
You can find resource list here..
You have to select the Resource that meets your req.

Resource Name	Resource Type	Reservation Start Date	Reservation Finish Date	Calculated Cost	Select
Computer1	0	09/04/2012 13:58	09/04/2012 16:58	9	
File1	2	09/04/2012 13:58	09/04/2012 16:58	9	
Storage1	1	09/04/2012 13:58	09/04/2012 16:58	9	
				27	Select
Computer1	0	09/04/2012 16:58	09/04/2012 19:58	9	
File1	2	09/04/2012 16:58	09/04/2012 19:58	9	
Storage1	1	09/04/2012 16:58	09/04/2012 19:58	9	
				27	Select
Computer1	0	09/04/2012 19:59	09/04/2012 22:59	9	
File1	2	09/04/2012 19:59	09/04/2012 22:59	9	
Storage1	1	09/04/2012 19:59	09/04/2012 22:59	9	
				27	Select
Computer1	0	09/04/2012 23:00	10/04/2012 02:00	9	
File1	2	09/04/2012 23:00	10/04/2012 02:00	9	
Storage1	1	09/04/2012 23:00	10/04/2012 02:00	9	
				27	Select
Computer1	0	10/04/2012 02:01	10/04/2012 05:01	9	
File1	2	10/04/2012 02:01	10/04/2012 05:01	9	
Storage1	1	10/04/2012 02:01	10/04/2012 05:01	9	

Figure 6-Available Resource List

D. *Allocate the Resource*

Best 10 resource triples at maximum are displayed to the user. The system asks users to choose only one of them. As showed in Figure 6, users can select resources for specified time interval. But the system actually does not reserve all resources in same period. If Expected Input File Size or Expected Output File Size is defined in jdl file, in order to increase the efficiency of resources, the system allocates the resources in different period. Execution time of job could be retrieved from job information but the user could not estimate the time during transferring the files through network. While allocating the resources, this information also should be considered. In figure 7, T and Z zones shows actual usage of resources. In available interval, there are some idle time for SR and FR so new jobs can be scheduled on this interval. Ratio of X to Z and T regions vary depending on the type of grid. If the grid is data intensive grid the large part of the work will be on SR or FR but if it is computational intensive grid then the large part of the work will be on CR. Since these calculations are done approximately Z and T zones should be considered as calculated with buffered time to prevent conflict with other jobs.

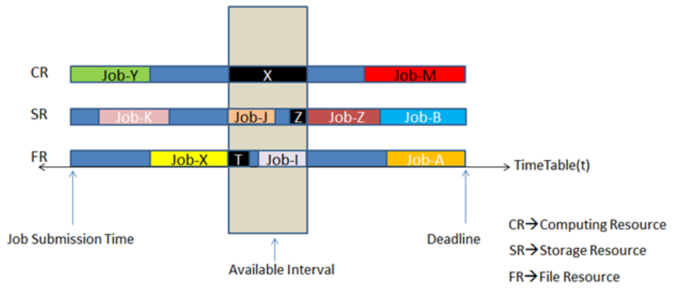


Figure7-Allocate Resources

E. *Submit Jobs to Resources*

This section describes how the system will submit jobs to resources. This component of ITU-GRAM+ works as a batch job in background to listen database for reservation jobs. Before submitting the job, execution file of the job should be transferred to CR. For this purpose, another batch job is defined to check the database and retrieve jobs which will start immediately. This batch job works in a periodic time with multithread to send files using secure FTP protocol. Submitting jobs to resources is done by using web service technologies. All necessary parameters to execute job will be passed to resources via web service. In figure 8 shows interactions between resources and resource manager during execution of job. In first step job object of *sendJob* method includes parameters listed below. These parameters will be used by resource in different steps of job execution as showed following.

Job→ {*resIdList, remoteInputIp, remoteInputPath, inputFileName, userId, jobName, command, remoteOutputIp, remoteOutputPath, outputFileName, executableFile, ioType, inputType, outputType, gridType*}

After this submission, ITU-GRAM + waits update from the resources, after that time, control is on resources. If we go over the same example is described previous sections the steps on execution of job is showed in Figure 8. These states have been defined in the system in order to track jobs in every step. *UpdateJobStatus* is an another web service which is exposed by Grid Resource Manager to retrieve the status of job from resources. *resIdList* parameter is passed to resource in step1 while submitting the job so the resources have this information. Since the web service is a stateless protocol it is important for grid system to restore the parameter on database and when any update comes from resources it matches the job with this *resIdList* parameter.

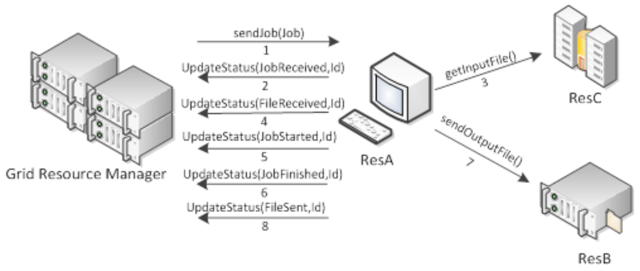

sendJob(Job)
1
UpdateStatus(JobReceived,Id)
2
UpdateStatus(FileReceived,Id)
4
UpdateStatus(JobStarted,Id)
5
UpdateStatus(JobFinished,Id)
6
UpdateStatus(FileSent,Id)
8
getInputFile()
3
sendOutputFile()
7
Grid Resource Manager
ResA
ResB
ResC

Figure 8-Submit Job

F. *Monitoring Job Status*

This section describes how to user can track their jobs on ITU-GRAM+. For this purpose a new web page is implemented to show the jobs status to users. ITU-GRAM+ has capability to allow users to cancel their jobs before starting job execution. But in this case penalty will be charged on users in order to keep stable the performance of the system. Figure 9 shows the all states are described in ITU-GRAM+ and which transactions are possible between these states. These states of job are stored on the database table which is called "ReservationJob". This table keeps the information which resources will be matched which jobs in which time period. All possible transition between these states are showed with letters below.

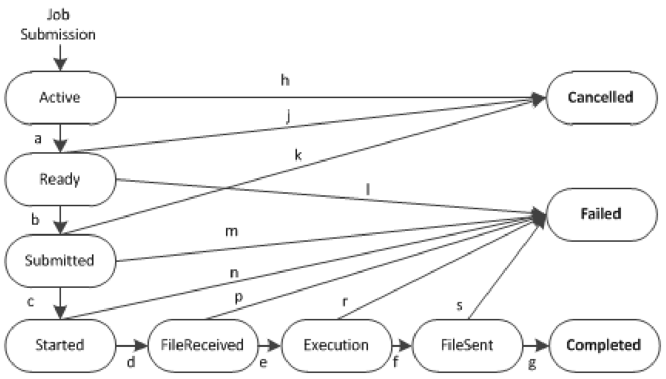


Figure9-State Diagram of Job

a→The status is changed *Active* to *Ready* when execution file transferred to CR.

b→The status is changed from *Ready* to *Submitted* when web service of *sendJob* is invoked.

c→The status is changed from *Submitted* to *Started* when resource received job information.

d→ The status is changed from *Started* to *FileReceived* when input file is retrieved.

e→ The status is changed from *FileReceived* to *Execution* when resource started to run job.

f→ The status is changed from *Execution* to *FileSent* when output file is sent.

g→ The status is changed from *FileSent* to *Completed* when job finished successfully.

h,j,k→The status is changed from *Active, Ready or Submitted to Cancelled* when the users want to cancel job.

l,m,n,p,r,s→ The status is changed from *Ready, Submitted, Started , FileReceived, Execution , FileSent to Failed* when any error occurs during job execution

Resource states are also kept in ITU-GRAM+ as showed in Figure10. ITU-GRAM+ works with reservation method so these states are valuable for the system in order to prevent conflicts.

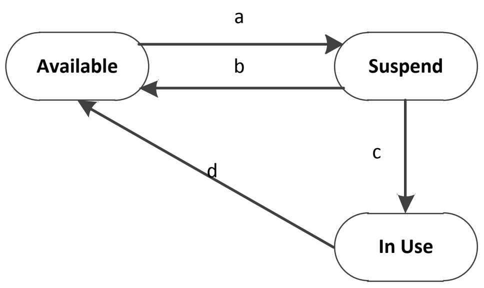


Figure 10. State Diagram of Resource

a→ The status is changed from *Available* to *Suspend* when resources are displayed to user.

b→ The status is changed from *Suspend* to *Available* when user releases or timeout is reached.

c→ The status is changed from *Suspend* to *In Use* when user selects resources that meets requirements

d→ The status is changed from *In Use* to *Available* when job is completed.

To display completed and waiting jobs to user, specific web pages are implemented. It is also possible to cancel or monitor jobs using these pages.

V. CONCLUSION AND FUTURE WORK

ITU-GRAM+ is a grid middleware which allows to users lot of capabilities to execute their jobs. The important feature of the system is that it is implemented using all open source software. In addition to this the other functionalities on the components are valuable for future work such as allocation resources considering not only availability of all resources but also the resource utilization. Besides using web service technology to ensure communication between resources and middleware is added value for Grid system. ITU-GRAM+ is not only support data or computational grids it works with fine both of them. As a result ITU-GRAM+ helps developers to create their own middleware based on open source software and provide standard and basic interfaces to users to submit their jobs. Behind of these functionalities basic user access protocols has been developed but other security related issues are kept out of the scope of this paper. In addition to this recover mechanism for failure states is handled with basic implementation. Since the ITU-GRAM+ is free software it is open any improvement for future works.

REFERENCES

[1] Job submission and management through web services: the experience with the CREAM service C Aiftimiei, P Andreetto, S Bertocco, D Cesini, M Corvo,S Dalla Fina, S Da Ronco, D Dongiovanni, A Dorigo, A Gianelle,C Grandi, M Marzolla, M Mazzucato, V Miccio, A Sciaba',M Sgaravatto, M Verlato and L Zangrando, 2008

[2] Service-Oriented Architecture: Concepts, Technology, and Design, Thomas Erl Prentice Hall PTR Upper Saddle River, NJ, USA ©2005 ISBN:013185858

[3] Ten actions when Grid scheduling: the user as a Grid scheduler, Kluwer AcademicPublishers Norwell,MA,USA ©2004 table of contents ISBN:1 -4020-7575-8

[4] A Taxonomy and Survey of Grid Resource Management Systems, Chaitanya Kandagatla,2003

[5] Globus Toolkit Version 4:Software for Service-Oriented Systems,Ian Foster,2006

[6] Job Submission Description Language (JSDL) Specification, Version1.0, 2005 http://www.gridforum.org/documents/GFD.56.pdf

[7] From Grid Middleware to a Grid Operating System, Arshad Ali, Richard McClatchey, Ashiq Anjum, Irfan Habib, Kamran Soomro, Mohammed Asif, Ali Adil, Athar Mohsin ,2006

[8] Available on 2012, http://tomcat.apache.org/

[9] Available on 2012, http://www.oracle.com/technetwork/java/

[10] Available on 2012, http://www.mysql.com/

[11] Available on 2012, http://www.hibernate.org/

[12] Available on 2012, http://www.oracle.com/technetwork/java/javaee/jsp/index.html

[13] Available on 2012, http://struts.apache.org/2.x/

[14] Available on 2012, http://axis.apache.org/axis2/java/core/