International Journal of Computer Science and Applications,
Vol. 5, No. 3, pp 56 - 69

GRID COMPUTING WITH GLOBUS : AN OVERVIEW AND RESEARCH CHALLENGES

RAJEEV WANKAR

Department of Computer and Information Sciences, University of Hyderabad
Hyderabad, AP 500 046, India
wankarcs@uohyd.ernet.in
http://dcis.uohyd.ernet.in/~wankarcs

The Open Grid Forum (OGF) is an organization that resulted from the merger of the Global Grid Forum (GGF) and the Enterprise Grid Alliance (EGA). GGF was an international organization that started in 1999, with the focus on the development of open standards for grid software interoperability, common practices, agreements and other related issues and proposed several specifications with the help of several working groups. The grid project started with the aim of using high-end computational recourses, networks, databases and scientific instruments owned and managed by multiple organizations. Globus [5] is one of the most successful projects in grid computing to test these specifications. Although it could overcome many technological barriers, many are still remains as open questions. In this article we discuss about grid, Globus Toolkit and present some of technical challenges the grid community faces. Further, we provide future research directions in Grid Computing.

Keywords: Globus; Grid, OGF, OGSA, WS-GRAM, MDS, Meta Scheduler

1. Introduction

The concept of Grid in Computing was first envisaged by Leonard Kleinrock in 1969 when wrote: "We will probably see the spread of computer utilities, which, like present electric and telephone utilities, will service individual homes and offices across the country" [13, 14]. In 1998, Carl Kesselman and Ian Foster suggested a proper definition of grid they wrote- "A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities". In the present scenario it is define as a "type of parallel and distributed system that enables the sharing, selection, and aggregation of geographically distributed autonomous resources dynamically at runtime depending on their availability, capability, performance, cost, and users' quality-of-service requirements". Grid computing is more about collaborating and resource sharing than high performance computing. Grids enable virtual organizations share geographically distributed resources as they pursue common goals: assuming the absence of central location, central control and omniscience [3, 6, 7].

Proper definition of various components of the grid i.e. Organization, Interface, Policy, Protocol, Virtual Organization, Virtual Machine, Programming System, Abstract Machine, Programming Model, and grid programming systems has been given in [8].

These definitions are derived from the established domain of distributed systems and are based on explicitly stated axioms and assumptions. These definitions establish a basis for characterizing and classifying grid research.

Although grid computing appears quite similar to the normal distributed computing, its requirements are much more complex than distributed computing. Distributed Computing refers to managing hundreds or thousands of computer systems which individually are more limited in their memory and processing power. On the other hand, grid computing concentrates on the efficient utilization of a pool of heterogeneous systems with optimal workload management. The basic aim is to utilize an enterprise's entire computational resources (servers, networks, storage, sensors, scientific instruments and information), acting together to create one or more large pools of computing resources. In the process, it also respects the organization's security policies. There is no limitation of users, departments or originations in grid computing. Grid computing focuses on the ability to support computation across multiple administrative domains that sets it apart from traditional distributed computing.

If one wants to execute an application on the grid which connected through some network then the following questions come to the mind: 1. Where are the resources to run the applications? 2. What configuration of machines needed to run the job? 3. How to schedule the job on remote machine? 4. How secure is to run the job on remote machines? 5. How to transfer the data on the remote machines? 6. How to monitor the execution of the job and to get results? To answer these questions, in 1996 one of the most popular research and development projects focused on enabling applications of grid concepts to scientific and engineering community, known as Globus was started. This project started as a joint initiative of the Argonne National Laboratory's Mathematics and Computer Science Division, the University of Southern California's Information Sciences Institute, and the University of Chicago's Distributed Systems Laboratory. Several other projects also started along with Globus, to name few are Gridbus [49], Legion [44] NetSolve [46] and Unicore [50]; each has its own architecture and application requirements.

The answers of the questions posed in the last paragraph lies in the design of the Globus. It has Security, Data Management, Execution Management, Information Services and Common runtime components. In this paper we will talk about all of these, briefly explain the present state of Globus Toolkit, discuss the future challenges in grid computing and propose research directions. The paper is organized in the following way: In section 1, we present the evaluation of the Globus Toolkit, in section 2 we talk about the present state of research in the Execution Management. In section 3 we present the recent contributions in the area of Grid Security, in section 4 about the Information Services. In Section 5 we give brief of Data Management and discuss some research problems.

2. Globus toolkit

The Global Grid Forum [43] developed standard interfaces, behaviors, core semantics for grid applications based upon web services. GGF introduced the term Grid Service as an extended web service that conforms to the GGF OGSI standard. Standard provided for interoperability of independently developed services. Before 2004 (Pre GT4), the Globus Grid Forum (GGF) standard was divided into two parts: 1. Open Grid Services Architecture (OGSA) and 2. Open Grid Services Infrastructure (OGSI). OGSA defined standard mechanisms for creating, naming, and discovering grid service instances. It also addresses architectural issues relating to interoperable grid services. OGSI on the other hand was based upon grid service specifications. It specified a way clients interact with a grid service i.e. service invocation management, data interface, security interface etc... Later, OGSI was removed by retaining some mandatory specifications and merging them with OGSA, and new Web Service Resource Framework (WSRF) is introduced. *(Figure 1: Courtesy www.globus.org).*

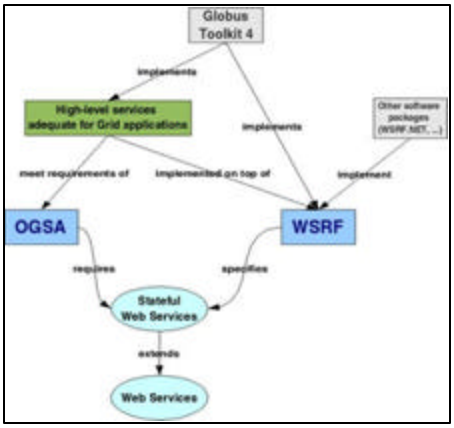

Figure 1: New Specifications for GT4 using WSRF

Globus Toolkit Version 2.4 (GT 2.4) provided a standard development framework which could be used to develop grid applications. The toolkit supports discovery, management, and monitoring of grid resources, as well as file management and security features. This was one of first popular Grid Toolkit that was used in many grid projects around the world.

Another version of Globus toolkit (GT 3) is based on the Java APIs. It is an open-source implementation of the Open Grid Services Infrastructure (OGSI) and provides a number of available OGSI services and the ability to create custom OGSI-compliant services as needed. This toolkit gave a powerful framework to the users for building grids, based on standard Web Services Architecture. The OGSI specifications which were build on grid and web service technologies defined how grid services are created and managed, and how information can be exchanged. OGSI uses the Web Services Description Language (WSDL) to define grid services that are accessible over the Internet using SOAP/HTTP protocols.

The latest version of GT4 includes software for security, information infrastructure, resource management, data management, communication, fault detection, and portability. It uses many third party software services and it is packaged as a set of components that can be used either independently or together to develop grid applications. It has both WS

and non-WS components. Common runtime component was first used with GT4 to provide both web and pre-web services with a set of libraries and tools that allows these services to be platform independent using web services stack (WSRF, WSN, etc…). The latest GT4 provides Java WS Core, C WS Core, Python WS Core APIs and tools for developing grid services and provides a run-time environment for hosting them. They provide support for management of the state, discovery of resources, naming and many more. It also provides advanced security infrastructure including different mechanism of transport and message level security.

There are four key services of GT: 1. GSI (Grid Security Infrastructure), used for grid security, 2. WS-GRAM (Grid Resource Allocation Manager), for remote job submission and control, 3. GridFTP, for secure data transfer and 4. MDS (Monitoring and Discovery Service), for service information. GT4 uses WSRF, which is a framework consisting of a number of specifications such as WS-Resource Properties, WS-Resource Lifetime, WS-Service Groups, WS-Notification, etc… WSRF specifies how to use XML to describe and access a resource's properties, how *stateful* (an instance of service is stateful if it can remember about prior actions, implies variables within service that can maintain values between accesses) resources are addressed, how a resource is created and messages to destroy resources, provides a message subscription and notification mechanism for web services etc...

3. Execution Management

Execution Management is an important component in GT. The basic job of the execution management service is the grid resource allocation and management. In GT4 remote job execution is carried out by the GRAM service, which is instrumental for providing a web service interface for submitting requests to execute jobs, defined in a job description language. It also monitors and controls the resulting job executions. GT4 includes two different GRAM services: the 'pre-WS GRAM', introduced in GT2, and the newer Web Services-based 'WS GRAM', introduced in GT4.

When a job is submitted to the Globus, GRAM sends it to different distributed resources. The challenge remains to design a Meta scheduler, which can take care of even scheduling jobs at the remote sites. When the user Client sends the job request from the local user, the request is forwarded by this Meta Scheduler to a grid manager; the grid manager in turn queries the Resource Manager (RM) for resources. RM stores information about local and remote resources. If available resources are found on a local site, job request is forwarded to the local scheduler. If the resources are not available locally, job request is sent to a remote site's Globus gate-keeper, which in turn forks the job to the job manager and is finally submitted to the remote scheduler for execution.

There are several Meta Schedulers exist in the literature some of them are Condor-G [45], DIRAC [15], EMPEROR [16], DIANA[17], GridWay [19] and GMarte (Integration of

metascheduler into Service-Oriented Architecture based on WSRF Grid services) [18]. One of the popular Meta scheduler Condor-G uses the Classad language to describe machine states, job constrains and job requirements. The Matchmaking algorithm is used in it for job scheduling in the centralized scheduling system. The extended version of Classad is used to support parallel jobs and mapping between jobs and resources with the experiential performance model.

Nimrod [48] is another resource broker for parametric computing on computational grids. It supports computational economy paradigm for grid computing and budget constraints based scheduling. Alchemi/Gridbus [1] is a .NET based toolkit for service-oriented computing. It provides services for (a) management of resources based on distributed computational economy at co-operative and competitive levels and (b) deployment of compute and data grid applications on them. EMPEROR provides a framework for implementing scheduling algorithms based on performance criteria. In implementing a particular instantiation of this framework, they have proposed a method for predicting host load and memory resources, and accordingly estimating the running time of a task. EMPEROR provides fully fledged grid scheduling functionality, which complies with OGSA standards. EMPEROR is one of the few standards based meta-schedulers makes use of dynamic scheduling information. Data Intensive and Network Aware (DIANA) is a meta-scheduling approach which takes into account the network characteristics along with computation and data when scheduling single or bulk jobs. It allows for optimized scheduling decisions as well as reduced execution times of data intensive jobs.

The latest version of the Globus Toolkit includes both pre-WS and WS GRAM services to submit, monitor, and control jobs on remote grid resources. GridWay is a meta-scheduler, which allows the simultaneous and coordinated use of pre-WS and WS-GRAM services and makes easy transition to a web service implementation of the Globus components.

GRAM services are used for several other purposes also. The MPICH-G2 [34] is the implementation of the Message Passing Interface (MPI) that uses GRAM to schedule jobs on distributed grid nodes. Ninf-G [35] uses GRAM interface to dispatch tasks on nodes. QMP: LQCD Message Passing API is another implementation of the MPI standards for the grids [47].

A high performance scheduling algorithm based on Fuzzy Neural Networks has been proposed in [28]. Fuzzy Logic technique is used to evaluate the grid system load status, and adopt the Neural Networks to automatically tune the membership functions. It has been shown how the proposed algorithm is beneficial when there are many factors that influence the system's load circumstances; as the number of factors increase, it becomes very difficult to set up the system. In another work, a predicting model of task's runtime, based on Back-Propagation neural networks considering several factors has been proposed which is the heart of any scheduling and resource allocation algorithms [29]. It

has been claimed that the method has many advantages including small network structure, task's user deadline, quick learning and use conveniently etc.

One of the major challenges for the grid computing researchers is to find out the efficient allocation of resources to the tasks submitted by users. Due to dynamic nature of resource availability and lack of the centralize control, allocation mechanism should be highly distributed and robust to tackle problems arise in the grid environment. A lot of research has been carried out to support the decision making of the Meta scheduler using well known machine learning techniques. Even simple reinforcement learning can be used to achieve load balanced resource allocation in large scale heterogeneous system [21]. Researches can use the theory of Genetic Algorithms, Rough Sets and Neural Network for the scheduling or predicting approximate time of execution of the job which is a computationally hard problem to solve otherwise.

4. Grid Security

Security is the basis for the grid and key requirements for ensuring a secure connection involves: Data Confidentiality - protection of information exchange against spying, Authentication - proof of identity, Data Integrity - protection of message modified in transit (intentionally or by accident) and Non-repudiation - guarantee that sender cannot deny that he/she sent message, similarly receiver not deny receiving message [9].

In 1976, Whitfield Diffie & Martin Hellman invented Public-Key Cryptography. It is perhaps one of the most important results in the history of cryptography. It uses application of number theoretic concepts of functions. Grid Security Infrastructure (GSI) uses public key encryption for single sign-on (defined in the next paragraph). It uses X.509 certificates for credentials and Secure Sockets Layer (SSL) communication protocol for message security [11].

In GT4's default configuration, each user and resource is assumed to have X.509 public key credentials. Certificate Authority (CA) is a program that issues certificates to processes. One can get a certificate by running certificate request command in GT. In the core GSI software provided by the Globus Toolkit user's private key is stored in a file in the local computer's storage. To prevent other users of the computer from stealing the private key, the file that contains the key is encrypted via a password (also known as a pass-phrase). To use the GSI, the user must enter the pass-phrase, which is required to decrypt the file containing their private key. Single sign-on is the process of enabling user and it's agents to acquire additional resources without repeated authentication (passwords) and it is achieved with proxies. It consists of a new certificate with new public, private keys and owner's identify. This certificate is signed by the owner and not by the CA and it is given limited lifetimes (generally 24 hours). Proxy's private key is not kept as secure as owner's private key. The research can be carried out to provide more secure grid authentication process, i.e. by keeping the private key of the user on some

external storage (may be a electronic card/hard key) and not on the user's secondary storage. Use of Visual Cryptography in conjunction with the authentication process is a challenging research area which can be explored more.

Apart from authentication, there are several other additional factors which are very important from the security perspective. Authorization is the process of deciding whether a particular identity can access a particular resource, Access control covers broader aspect of authorization and controlling specific types of access. There are three main types of access controls, namely Mandatory Access Control (MAC), Discretionary Access Control (DAC), and Role Based Access Control (RBAC) [12]. These Access control can be Static/Dynamic [4], Fine Grain or can be Coarse Grain [2].

There are several authorization modules present in the literature starting from the basic Globus based grid-map authorization systems to systems that uses XACML [10] to query an external service for the authorization. These all systems support the generation, storage and retrieval of the credentials based on the policy enforcements. Authorization systems themselves are Push and Pull Authorizations. These types of authorization systems where authorization credentials are "pushed" are called push based authorization systems. In such a model there exists a certificate generator who checks the user's credentials and generates a certificate so that the user can access the resource. The access controller allows access to the resource based on the certificate validity. In case of the pull model, where the user supplies the minimum credentials like the username, password, and the controller makes the access decision based on the user policies pulled from the database [36]. Some of the examples are MyProxy [66], VOMS [39], CAS [38], Akenti [37] PRIMA [20] and PERMIS [40]. GSI supports the notion of local policy enforced locally. To achieve this, GSI provides mechanisms for translating a user's GSI identity (i.e., the distinguished name from the user's certificate) to a local identity. Once translated, the local identity can be used to enforce local policy decisions, such as file access, disk quotas, and CPU limits.

Community Authorization Service (CAS) is a Virtual Organization level system that addresses solution of three critical authorization problems that arise in distributed virtual organizations: scalability, flexibility and expressibility, and the need for policy hierarchies. A trusted third party has been proposed in CAS that is administered by the virtual organization that performs fine-grain control of community policy while leaving ultimate control of resource access with the resource owners.

PRIMA, a system for PRIvilege Management and Authorization, provides enhanced grid security services. This system is designed based on the usage scenarios of grid users. The requirements for added flexibility, increased expressiveness, and more precise enforcement are met by a combination of three mechanisms: (1) use of secure, fine-grained privileges representing externalized access rights for grid resources that can be freely created, shared, and employed by grid users; (2) a dynamic policy generated for

each request combining the request's user provided privileges with the resource's access control policy; and (3) dynamic execution environments specially provisioned for each request that are enforced by the resource's active operating system and which support legacy applications.

Many authorization systems seldom look at the way in which they store and organize the resources' security policies to work more effectively. In other words current authorization systems do not define a data structure to store and manage the security policies intending to provide a quick response to the user. One of the structural representations to achieve set goals of the grid environment is hierarchical representations. Storing security policies in such a structural manner (hierarchical decision tree) will increase the scalability of the authorization system by several times. One of the research challenges is to design such a data structure or a storing mechanism which can implement this strategy.

5. Information Services

Information services are one of the most important requirements of the grid system. It includes Discovery and Monitoring of the resources so that the Meta-scheduler can schedule the job. Discovery is the process of finding the computational resource/service which suits the client's requirement. On the other hand Monitoring detects and diagnoses the problems when the job is executed remotely. Both require the information collection about the resources from the complete grid environment.

Monitoring and Directory Service (MDS) [24] is an important service that provides many important functions and available with GT from its inception. This service was originally called as "Metacomputing Directory Service". It is based upon Light weight Directory Access Protocol (LDAP). MDS consist of two types of services: Grid Resource Information Service (GRIS) and Grid Index Information Service (GIIS). GRIS is associated with each resource and answers queries from client/user about the particular resource. It accesses an "information provider" deployed on that resource for requested information. GIIS on the other hand is a directory service that collects ("pulls") information for GRIS's. It is a "caching" service which provides indexing and searching functions.

In GT4 these mechanisms are provided by the Web Service Resource Framework (WSRF) and WS-Notification Specification. GT4 provides a common aggregator framework for collecting information from services, thus uses MDS-Index: (Xpath queries, with caching), MDS-Trigger: (perform action on condition) and MDS-Archiver: (Xpath on historical data). GT provides a range of browser interfaces, command line tools and many web service interfaces that allow the user to query and access the gathered information. One of the example services is WebMDS. It has been observed that as the MDS4 Index grows, query rate and response time both slow downs. This response time slows due to increasing data transfer size. GT4 is also served by many third party

information providers. They collect information from some system and make it accessible as WSRF resource properties. Few of the information providers are Ganglia [61], SCMSWeb [62], CluMon [63], Nagios [64] and Gridscape[65].

For many applications, the ability to find a server (or a set of servers) that satisfies a set of functional and network properties are very valuable. In their paper An-Cheng Huang and Peter Steenkiste [22] have proposed an integrated solution called as Network-Sensitive Service Discovery (NSSD). It allows users to benefit from network-sensitive selection without having to implement their own selection techniques, and it does not require providers to expose all their server information to users.

Grid resource locations are either centralized or hierarchical and often prove inefficient as the grid systems scales rapidly. On the other hand, the Peer-to-Peer (P2P) paradigm emerged as a successful model to achieve the scalability in distributed systems and thus can be seen as an alternative for the discovery [31]. Because such information systems are built to address the requirements of organizational-based grids, they do not deal with more dynamic, large-scale distributed environments. The number of queries in such environments makes a client–server approach ineffective. The super-peer model is a novel approach that provides the convergence of P2P models and grid environments, as super-peer serves a single organization in a grid and at the same time connects to other super-peers to form a peer network at a higher level. One of the methods based on the super-peer model for handling membership management and resource discovery services in a grid appears in [23]. In this paper a new resource discovery protocol is also presented.

Resource discovery is an important function of the data management in grids. Presently majority of the solutions provided for it are either centralized discovery based (LDAP) or decentralized based on P2P. It would be interesting research problem to see how the centralized or decentralize solutions based on ontologies can be useful for resource discovery in a better way.

6. Data Management Services

Any large distributed system application requires to access, update, integrate and transfer large amount of data. This is an important requirement of many grid applications also. The designer of the Globus took this requirement into consideration from its inception and provided several mechanisms, using which one can perform these operations. GridFTP is one of the important services that are implemented in the GT. It provides libraries and tools for the reliable, secure, high-performance memory to memory and disk to disk file transfer facilities. Apart, it provides a Reliable File Transfer (RFT) service [25] which provides multiple GridFTP reliably. Replica Location Service (RLS) service provides information about the location of the replicated files [27]. Data Replication Service (DRS) is a combination of RLS and GridFTP for the data replication

management. Data Access and Integration (DAI) tools [26] provide access to the server side processing of the relational and XML data base. In OGSA-DAI, issues related to the distributed query are addresses by OGSA-Distributed Query Processing (DQP).

Transparency is the basis of any distributed system. In a traditional distributed system the transparency exist for 1. Access- hides differences in data representation and how a resource is accessed, 2. Location- hides where a resource is located? 3. Migration- hides that a resource may move to another location 4. Relocation - hides that a resource may be moved to another location while in use, 5. Replication - hides that a resource may be copied at several locations, 6. Concurrency - hides that a resource may be shared by several competitive users (consistent state of resource is important), 7. Failure- hides the failure and recovery of a resource, 8. Persistence- hides whether a (software) resource is in memory or on disk. Many of these transparencies are difficult to achieve in distributed database systems.

In a large distributed data base system the data sources are several, and many of them are typically under local control and very heterogeneous in size and complexity. A lot of research has been carried out in recent time for the design of Peer-to-Peer data management systems. Unfortunately many of the P2P systems can not be scaled to meet ever increasing requirements of the grid based distributed systems as the grids are dynamic in nature and many resources come and go. Data management in the grids context offers new research opportunities since traditional distributed database techniques need to scale up for supporting high data autonomy, heterogeneity and dynamicity.

With the increasing popularities, XML is becoming a standard for the exchange of information on the internet. Being a natural carrier of the data, and in the absentia of any true open source database management system, any system design based on the XML can be very useful especially when several supporting XML technologies exist. For example local query can be done on local schema without worrying about the global schema and later XQuery can be used for the querying on the global one. Investigations on designing grid based database systems using XML standards is an interesting research problem.

In many of the virtual grid organization the high availability of the data is very important. In Data Grids, the data is replicated in such a manner that it is always available, especially in the government sectors where the information pertaining to one state may be useful or related to the information available at other state. Replication of the information is also useful in case the network is vulnerable to natural catastrophe or we wish to reduce the fault tolerance. Techniques can be proposed for how to replicate the data optimally. In this context, extension of OGSA-DQP for handing replicated information is an interesting topic for the research.

The Globus Toolkit provides a number of components for doing data management. The components available for data management fall into two basic categories: data movement

and data replication. GridFTP, a protocol defined by Global Grid Forum, provides a secure, robust, fast and efficient transfer of bulk data. The Globus Toolkit provides the most commonly used implementation of this protocol; these implementations are a server implementation, command line client and a set of development libraries for custom clients.

High speed bulk data transfer is an important part of many data-intensive scientific applications in grids. SABUL [30] and UDT [42] are application-level data transfer protocols for data-intensive applications. It is designed for reliability, high performance, fairness and stability. SABUL uses UDP to transfer data and TCP to return control messages. A rate-based congestion control that tunes the inter packet transmission time helps achieve both efficiency and fairness. In order to remove the fairness bias between flows with different network delays, SABUL adjusts its sending rate at uniform intervals, instead of at intervals determined by round trip time. On the other hand, UDT is a protocol for emerging distributed data intensive applications over wide area high-speed networks. UDT uses UDP to transfer bulk data and it has its own reliability control and congestion control mechanisms. This protocol is not only for private or QoS-enabled links, but also for shared networks. Furthermore, UDT is also a highly configurable framework that can accommodate various congestion control algorithms. It would be interesting research problem to see how these protocols can be integrated with the GSI and be used with the data intensive applications.

Workflow management is another important activity in the data management. Using the proper workflow, it is be possible to combine and execute the complex chain of applications without the human intervention. Several workflow management systems for scientific computing have been proposed [41], such as: Askalon [33], D2K [51], Kepler [58], P-Grade [60], GridAnt [53], GridNexus [54], GridServiceBroker [55], Pegasus [52], Teutaa [57], Swift [56], and Xbaya [59]. However, processing workflows in a grid is an open issue and imposes many challenges [32]. The scheduling of a workflow focuses on mapping and managing the execution of tasks on grid shared resources that are not directly under the control of these workflow systems. Thus, choosing the best strategy for workflow execution in a grid is a challenging research area.

7. Conclusion

Grid project started with the aim of designing a system which works well in the absence of central location, central control and omniscience, which is still difficult to achieve till to date. As the grid technology is getting more mature, usage of it is also increasing. In the beginning it was used for the scientific applications but recently many industrial organizations have started to use it for their enterprise applications. In order to effectively utilize grid technology/resources, researches need to address technical and social challenges arise in this loosely coupled system. In this paper we have posed few problems and presented new research directions to overcome some of the problems.

References

1. Akshay Luther, Rajkumar Buyya, Rajiv Ranjan, and Srikumar Venugopal, "Alchemi: A .NET -Based Enterprise Grid Computing System", Proceedings of the 6th International Conference on Internet Computing (ICOMP'05), June 27-30, 2005, Las Vegas, USA.
2. Elisa Bertino, Pietro Mazzoleni, Bruno Crispo, Swaminathan Sivasubramanian Elena Ferrari, "Towards supporting fine-grained access control for Grid Resources" 10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS'04), 2004, pp. 59-65.
3. F. Berman, G. C. Fox and A. J. G. Hey (*edited),* "Grid Computing Making the Global Infrastructure a Reality", Wiley, 2003.
4. Guangsen Zhang, Manish Parashar, "Dynamic Context-aware Access Control for Grid Applications", IEEE Fourth International Workshop on Grid Computing, 2003, pp. 101-108.
5. Ian Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit", Intl J. of Supercomputer Applications, 11(2):1997, pp. 115-128.
6. Ian Foster, C. Kesselman, S. Tuecke, "The Grid: Blueprint for a New Computing Infrastructure", International Journal of Supercomputer Applications, 15(3):2001.
7. Introduction to Grid Computing with Globus, IBM Redbooks (on-line)
8. M. Parashar and J.C. Browne, "Conceptual and Implementation Models for the Grid", Proceedings of the IEEE, Special Issue on Grid Computing, IEEE Press, 93(3):2005, pp. 653-668.
9. Marty Humphrey, Mary R. Thompson, Keith R. Jackson. "Security for Grids", Proceedings of the IEEE, 93(3):2005.
10. OASIS Security Services Technical Committee XAMCL, Extendible Access Control Markup Language (XACML) committee specification 1.0, Feb. 2003.
11. Von Welch1, Ian Foster, Carl Kesselman *et al.*, "X.509 Proxy Certificates for Dynamic Delegation", 3rd Annual PKI R&D Workshop, 2004.
12. Weizhong Qiang, Hai Jin, Xuanhua Shi, Deqing Zou, and Hao Zhang M. Li *et al.* (Eds.), "RB-GACA: A RBAC Based Grid Access Control Architecture", GCC 2003, LNCS 3032, 2004, pp. 487-494.
13. Jeremy M. Norman (*edited),* From Gutenberg to the Internet: A Sourcebook on the History of Information Technology: 2005, pp. 870.
14. L. Klienrock, "UCLA press release," 1969, http://www.lk.cs.ucla.edu/LK/Bib/REPORT/press.html
15. Eddy Carona, *et al.*, "Definition, modeling and simulation of a grid computing scheduling system for high throughput computing", Future Generation Computer Systems 23:2007, pp. 968-976.
16. Lazar Adzigogov, John Soldatos and Lazaros Polymenakos, "EMPEROR: An OGSA Grid Meta-Scheduler Based on Dynamic Resource Predictions", Journal of Grid Computing 3:2005, pp. 19-37.
17. Richard McClatchey *et al.*, "Data Intensive and Network Aware (DIANA) Grid Scheduling", J Grid Computing 5:2007, pp. 43-64.
18. Molt´o,V. Hern´andez *et al.*, "A service-oriented WSRF-based architecture for metascheduling on computational Grids", Future Generation Computer Systems 24:2008, pp. 317-328.
19. Eduardo Huedo *et al.*, "A modular meta-scheduling architecture for interfacing with pre-WS and WS Grid resource management service", Future Generation Computer Systems 23:2007, pp. 252-261.
20. Markus Lorch & Dennis Kafura, "The PRIMA Grid Authorization System", Journal of Grid Computing 2:2004, pp. 279-298.
21. Aram Galstyan *et al.*, "Resource Allocation in the Grid with Learning Agents", Journal of Grid Computing 3:2005, pp. 91-100.

22. An-Cheng Huang and Peter Steenkiste, "Network-Sensitive Service Discovery", Journal of Grid Computing 1:2003, pp. 309-326.
23. Carlo Mastroianni a, *et al*., "A super-peer model for resource discovery services in large-scale Grids", Future Generation Computer Systems 21:2005, pp. 1235-1248.
24. Xuehai Zhang Jennifer M. Schopf, "Performance Analysis of the Globus Toolkit Monitoring and Discovery Service, MDS2", Proceedings of the International Workshop on Middleware Performance (MP-2004), part of the 23rd International Performance Computing and Communications Conference (IPCCC), April 2004.
25. Allock W., Foster I, Madhuri R., " Reliable data transport: A critical service for the grid", In Building Service Based Grids Workshop, 2004, Globus Grid Forum 11.
26. Atiknson M., Chervenal A. *et al*., "Data Access, Integration and Management", The Grid: Blueprint for a New Computing Infrastructure, Morgan Kauffmann, 2004.
27. David Cameron *et al.*, "Replica Management in the European DataGrid Project", Journal of Grid Computing 2:2004, pp. 341-351.
28. Kun-Ming Yu1 *et al.*, "A Fuzzy Neural Network Based Scheduling Algorithm for Job Assignment on Computational Grids", LNCS 4658: 2007.
29. Jingbo Yuan, Shunli Ding, Cuirong Wang, "Tasks scheduling based on neural networks in grid", Third International Conference on Natural Computation, ICNC 2007.
30. Yunhong Gu and Robert Grossman, "SABUL: A Transport Protocol for Grid Computing", Journal of Grid Computing 1:2003, pp. 377-386.
31. P. Trunfioa *et al.*, "Peer-to-Peer resource discovery in Grids: Models and systems", Future Generation Computer Systems 23:2007, pp. 864-878.
32. Yu, J., Buyya, R., "A Taxonomy of Scientific Workflow Systems for Grid Computing, SIGMOD Record, Special Section on Scientific Workflows 34(3):2005, pp. 44-49.
33. Wieczorek, M. *et al*., "Scheduling of scientific workflows in the ASKALON Grid environment", SIGMOD Rec. 34(3):2005, pp. 56-62.
34. N. Karonis, B. Toonen, and I. Foster, "MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface", Journal of Parallel and Distributed Computing, 2003.
35. Y. Tanaka, H. Takemiya, H. Nakada, and S. Sekiguchi, "Design, implementation and performance evaluation of gridRPC programming middleware for a large-scale computational grid", in Proc. of the 5th Intl. Workshop on Grid Computing (GRID 2004). Pittsburgh, PA, USA: IEEE Computer Society 2004, pp. 298–305.
36. Anirban Chakrabarti, Grid Computing Security, Springer-Verlag Berlin Heidelberg 2007.
37. Thompson, M., *et al.*, "Certificate-based Access Control for Widely Distributed Resources", in Proc. 8th Usenix Security Symposium. 1999.
38. Pearlman, L. Welch, V. Foster, I. Kesselman, C. Tuecke, S., A community authorization service for group collaboration", Proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks (POLICY'02) 2002, pp. 50-59.
39. R. Alfieri, R. Cecchini, V. Ciaschini, L. Agnello, A. Frohner, A. Gianoli, K. Lorentey and F. Spataro, "VOMS, an Authorization System for Virtual Organizations", in 1st European Across Grids Conference, Santiago de Compostela, Spain, February 13–14, 2003.
40. D.W. Chadwick and A. Otenko, "The PERMIS X.509 Role Based Privilege Management Infrastructure", in 7th ACM Symposium on Access Control Models and Technologies, Monterey, USA, June 3–4, 2002, pp. 135–140.
41. http://www.extreme.indana.edu/swf-survey/
42. http://udt.sourceforge.net/index.html
43. http://www-unix.globus.org/toolkit/
44. http://www.legion.virginia.edu
45. http://www.cs.wisc.edu/condor/condorg
46. http://www.cs.utk.edu/netsolve/
47. http://usqcd.jlab.org/usqcd-docs/qmp/QMP-2-0-Introduction.html
48. http://www.csse.monash.edu.au/~davida/nimrod/
49. http://www.gridbus.org
50. http://www.unicore.eu/

51. http://alg.ncsa.uiuc.edu/do/tools/d2k
52. http://pegasus.isi.edu/wms.php
53. http://www-unix.globus.org/cog/projects/gridant/
54. http://www.extreme.indiana.edu/swf-survey/GridNexus.html
55. http://www.gridbus.org/broker/
56. http://www.ci.uchicago.edu/swift/guides/quickstartguide.php
57. http://www.par.univie.ac.at/project/prophet/node4.html
58. http://kepler-project.org/
59. http://www.extreme.indiana.edu/xgws/xbaya/
60. http://www.lpds.sztaki.hu/pgrade/
61. http://ganglia.info/
62. http://www.opensce.org/components/SCMSWeb
63. http://clumon.ncsa.uiuc.edu/
64. http://www.nagios.org/
65. http://www.gridbus.org/gridscape/
66. http://grid.ncsa.uiuc.edu/myproxy/