Service Oriented Distributed Manager for Grid System

Entisar S. Alkayal
Faculty of Computing and Information Technology
King Abdul Aziz University
Jeddah, Saudi Arabia
entisar_alkayal@hotmail.com

Prof.Dr. Fathy A. Essa
Faculty of Computing and Information Technology
King Abdul Aziz University
Jeddah, Saudi Arabia
fathy55@yahoo.com

Abstract— **Current problems in science and engineering become more complicated and need more computing power to tackle and analyze. The processing power of a single computer system has become inadequate for that problem and use of a supercomputer is not always an optimal solution. Nowadays institutions as universities or companies have lots of Desktop PCs in their infrastructures. With advanced in hardware, software and network gird computing are developed to benefits from these futures in idle computers in an efficient way to solve complex problems that need powerful computing requirements. Grid technology, which connects a number of personal computers, can achieve the same computing power as a supercomputer does, also with a lower cost. The goal of grid computing is to aggregate idle resources on the Internet or Intranet such as CPU cycles, storage spaces, etc. to facilitate utilization. In this research we present computational grid framework that used to utilize idle computers in Intranet to execute jobs in an efficient way than a single computer. We design and implement Service Oriented Distributed Grid Manager (SODGM) framework based on web services technology to manage resources and jobs in the system. SODGM balance the load among available resources to increase computers utilization in the system and minimize jobs response time. To evaluate the SODGM system, we performed tests on the Computers Laboratory in the Faculty of Computing and Information Technology in King Abdul Aziz University. The tests show that SODGM system provides good performance compared with using single computer in terms of execution time, resource utilization and system throughput.**

Keywords- Grid;Web Services;management; job scheduling

I. INTRODUCTION

The growing popularity of the Internet and the availability of powerful computers and high-speed networks as low-cost commodity components are changing the way that we do computing and use computers today. The interest in coupling geographically distributed computational resources is also growing for solving large-scale problems, leading to what is popularly known as grid computing. In grid computing , a wide variety of computational resources, storage systems and databases, special class of scientific instruments (such as radio telescopes), computational kernels, and so on are logically coupled together and presented as a single integrated resource to the user [1].

Grid management system is more complicated. Grid development involves the efficient management of heterogeneous, geographically distributed, and dynamically available resources. In this environment, the resource scheduler become one of the most critical components of the Grid management middleware, since it has the responsibility of selecting resources and scheduling jobs in such a way that the user/application requirements are met, in terms of overall execution time (performance) and resources utilized. Resource management in computational Grids provides the ability to determine whether or not resources are available, and if so, map the submitted jobs to specific resources [1]. Therefore, a resource manager needs to be able to perform the following services in an efficient way: authentication services, information services discovery, deployment, monitoring, and scheduling services. Grid computing has adopted Web services technology to deal with environmental heterogeneity and to enhance service and application interoperability. However, it is a challenge to realize web service applications with high performance, reliability and availability to meet the requirements of grid communities [2].

In this research we study, design and implement prototype of manager framework that used to manage resources in computational grid based on service oriented technology. SODGM is a management and scheduling system which provides an easy way to manage distributed computational resources and an efficient way to process a large number of user requests for computing. It's designed depending on Web Service to adapt robust, scalable and interoperability requirements by the grid system.

The research is organized as follows: **section 1** introduces the research subject and it also addresses the objectives and motivation of the research. **Section 2** presents briefs background about the Grid system, its definition and characteristics. It also provides brief summary about web service technology and its core standards. **Section 3** presents the related work to this research and literature review of computation grid systems and management middleware**. Section 4** discusses the design of proposed grid manager middleware based on web services architecture and provides a model for service oriented distributed grid manager (SODGM). In addition, **Section 4** discusses the results that are

obtained from testing SODGM and evaluate the system. **Section 5** presents the conclusion and direction for future works.

II. BACKGROUND

A. *Grid overview*

The concept of Grid computing started as a project to link geographically dispersed supercomputers, but now it has grown far beyond its original intent. Grids cannot be considered as a revolutionary technology; rather they have evolved from existing technologies such as distributed computing, the Internet, web services, various cryptography providing security features and virtualization technology [8]. The grid technology takes features from these technologies to develop a system that can provide computational resources for some specific tasks. Grid computing offers a solution to the computationally intensive nature problems. The grid computing paradigm is an emerging field of computing science that aims to offer a seamless, integrated computational and collaborative environment [3]. Ian Foster defines a computational Grid as "a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to high end computational capabilities" [10].

B. *Web services overview*

A service is any entity that provides some capability to its client by exchanging messages. A service is defined by identifying sequence specific messages exchanges that cause the service to perform some operations. A web service is any service that is available over the Internet, uses a standardized XML messaging system, and is not tied to any one operating system or programming language [38] (See Figure 1)

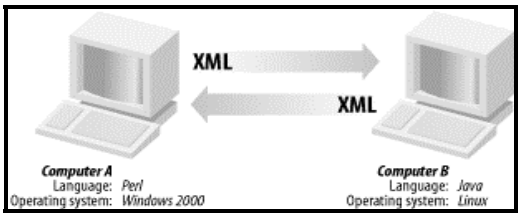

Figure 1: A basic web service

Web services are emerging as a promising infrastructure for building distributed applications. Web services are based on a Service- Oriented Architecture (SOA) in which clients are service requestors and servers are service providers. The W3C defines a Web Service as a software system designed to support interoperable machine to machine interaction over a network [11].

Web Services standards are important for building distributed applications, which are typically constructed from a set of services that are independently designed, deployed and managed. The widespread adoption of standards offers advantages including: interoperability, stability, tool support and implementation re-use. This is particularly important as Web Services technologies can be acquired from a variety of sources, including major vendors and the open source community. Developing a grid system that uses Web services therefore has a number of distinct benefits, including [12] : increased compatibility , increased flexibility ,Cross-platform-enabled development by eliminating the complexities of exchanging data , easy deployment using an existing Web server and easy communication and accessibility by making it simple to contact the grid components over an Intranet or Internet.

III. RELATED WORK

There have been many projects focused on grid computing that have designed and implemented resource management systems with a variety of architectures and services. In this section, we present related resource management services and their functions in some of grid systems.

A. *Condor*

Condor [6, 13] is a resource management system designed to support high-throughput computations by discovering idle resources on a network and allocating those resources to application tasks. The main function of condor is to allow utilization of machines that otherwise would be idle thus solving the wait-while-idle problem. Jobs submitted by the users are queued by Condor and scheduled on available machines transparently to the user. Condor resource requests are specified in Classified Ads resource specification language. Condor selects available machine to run a user's job, it can also migrate a running job from one machine to another until it is completed. Condor has a centralized scheduling model. A machine is the condor system (Central Manager) is dedicated to scheduling. Each condor work station submits the jobs in its local queue to the central scheduler which is responsible for finding suitable resources for the job execution. The information about suitable available resources to run the job (execution machine information) is returned to the job submission machine.

B. *Resource Managemetn in Legion*

Legion [7] is an object-based meta-system developed at the University of Virginia. Legion provides the software infrastructure so that a system of heterogeneous, geographically distributed, high-performance machines can interact seamlessly. Legion attempts to provide users, at their workstations, with a single, coherent, virtual machine. Legion provides a framework for scheduling which can accommodate different placement strategies for different classes of applications. Scheduler in Legion has a hierarchical structure. Users or active objects in the system invoke scheduling to run jobs, higher level scheduler schedules the job on cluster or resource group while the local resource manager for that domain schedules the job on local resources.

C. *Resource mangeemtn in Globus*

Globus [4] mainly works on grid infrastructure technologies. The core of Globus Grid is the toolset Globus Toolkit (GT). The current version GT4 has been released. GT comprises a set of layered grid tools realizing the basic services for security, resource location, resource management, communication, etc. They efficiently support the application grid infrastructure. The combination of Globus Toolkit and web service brings the future of a standardized grid research product .Globus can be viewed as a Grid computing framework based on a set of APIs to the underlying services [5]. Globus provides application developers with a pragmatic means of implementing a range of services to provide a wide-area application execution environment.

IV. SERVICE ORIENTED DISTRIBUTED GRID MANAGER (SODGM)

In this section, we introduce the architecture and services of our service oriented grid manager (SODGM) that is built based on web services technology to manage grid systems. SODGM offers different functions; each function has been designed and implemented based on web services technology. Using web services for resources management in grid system enhance interoperability and provide platform independent. Different resources in grid are varying in operating system, CPU, etc. This difference can result in a very heterogeneous character of some Grid environments and lead to complex management for these resources. A Services Oriented Architecture (SOA) is well suited to addressing some of the issues that arise from such heterogeneous, locally controlled but globally shared system.

A. *SODGM Architectures*

Grid resource management is the core component of grid, the general structure of SODGM that is designed in this research is shown in Figure 2. The system is built based on service oriented technology with a centralized manager structure. SODGM provides functions such as resource discovery, job submission, job management and monitoring. The general architecture model of SODGM is shown in Figure 3. It consists of 3 types of machines, one Manager module in server machine connected with database, many Executers (computing) machines that used to execute the jobs of users and Users machine that can be registered and connected to the system to submit their jobs and retrieve their results. Manager machine provides the management middleware that managing SODGM system, while executer machine is any machine that registered or connected to SODGM system through executer interface. Any user can register to the system and connect to the manager and send jobs to the manager . System Database contains static and dynamic information about Executers machines and jobs in the system .

In SODGM model, there is a central resource management unit to which each Executer connects and users send their jobs to that central manager. The manager is responsible for scheduling jobs among different executers. In general, each executer may have single or multiple processors and the processors at different executers can be either homogeneous or heterogeneous. Upon arrival, jobs must be assigned to exactly one executer for processing immediately by instantaneous scheduling or wait to be scheduled by the scheduler. We assume that jobs can be executed on any executer and no parallelism and migration is allowed among executers. The load monitor is responsible for probing the current state of each executer. The arrived jobs will be placed in a waiting queue at the manager. As the system workload grows heavy, there are more and more jobs waiting in the queue, the scheduler perform load balancing over the jobs. The load balancing objective now is to minimize the total execution time of those waiting jobs as well as a well-balanced load across all executer machines.

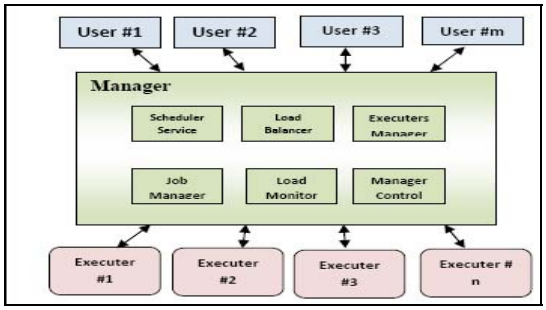

Figure 2: The General Structure of SODGM Model

A. *Manager Services in SODGM*

SODGM Manager manages the machines in the system and it responsible for naming the machine and registering it in the system. It also accepts the jobs from a user and schedules them depending on load balance algorithm then it migrates job to selected machine to execute it and getting the results. SODGM Manager consists of numbers of services each one of them performs specific tasks for the manager and cooperates with each other to satisfy manager functions. These services are Executer Registration Service, Users Registration Service, Job Manager Service, Job Scheduler Services, Load Balancer Service, Manager Control Service, Load Monitor Service and IDS. The descriptions of these services are as follow:

1) *Executers Registration Service :*

This service manages the requests of executers, it is used to register the new executer machine to the manager Database, connect user to the system after verifying authentication and disconnect user from the system. The user submits information about the executer machine to the manager machine. After the function of registration complete successfully then the manager generates unique ID to that executer to identify the machine in the system.

2) Users Registration Service :

This service manages the user's requests to register them to the system, it is used to register the new user machine to the manager Database, connect user to the system after verifying authentication and disconnect user from the system. The user submits information about the machine to the manager machine. After the function of registration complete successfully then the manager sends the generated unique ID to the user to identify the machine in the system.

3) Job Manager Service :

The Job manager service collects the information about jobs in the SODGM and stores them in the system database. Job manager service takes a job request from a user and stores job files in the manager machine and also stores the job into waiting queue. After executing a job, the job manager service also takes the responsibility of collecting results and sends it back to the user. When the job failed, the job manager restores the job to the waiting queue for new scheduling. The job manager service is responsible for executing jobs during its life time until it finish or stop. User's job may fail to execute, the role of job manager is monitoring job status information, in order to deal with a fault-tolerant in time monitoring the failure job. Basic status of grid job including: Submitted (submitted), Waiting (ready in ready queue before assign to resource), Ready (ready in scheduling queue), Running (running), Done (complete successfully), Failed (failure). Various conversions between the jobs states are: once job has been submitted, status shows as Pending, if execution starts up, the job will be deleted from Pending table and added to Running table, means the job is running.

4) Load Balancer Service

The role of load balancer service is responsible for choosing specific executer depending on executer availability and load. The load balancer selects executer information from information database services that satisfy load balancing strategies to maximizing the executer utilization and minimize the total job execution time. Then allocate the job to one of the available executer machine. Load balancer select job form waiting queue and assign it with appropriate executer then queue the job into ready queue and change the job state from Waiting to Ready .

5) Job Scheduler Service

Job scheduler service retrieves a ready job from the job ready queue and dispatches the job to the selected executer to be executed**.** The job scheduler schedules the job in the ready queue depending on executer availability. Job scheduler is responsible for dispatching the job to specific resource. Job scheduling retrieves jobs ready for execution from the Ready Job Queue sends a job to appropriate executer resource according to scheduling strategy and generates results.

6) Load Monitor Service

This Service monitors the Load of Executer by calling Executer Status service in each Executer in the system. The service stores the performance metrics for each executer machine in the database. The performance information are CPU utilization, Available Memory size and Available hard disk size.

7) Information database Service (IDS)

Information service provides services to enable resource registration while keeping track of a list of available executers in the SODGM system. The manager can query this entity for executers contact, configuration, characteristics and status information. Information contains static information and dynamic information about executer machines and jobs in SODGM. Executer information may be static which does not change with time, such as hardware type, memory size, the type and version of the operating system, the information is obtained once at the time by sampling, Or dynamic information which is sampled with a fixed time interval, such as CPU utilization, memory utilization, and length of job queue. As dynamic information plays a more important role for the grid resource management and scheduling, so it needs to ensure the real-time dynamic information.

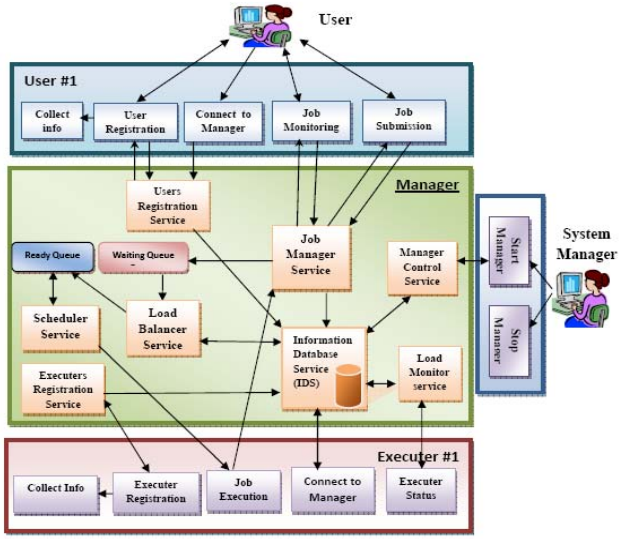

Figure 3: The Architecture of SODGM Model

B. *Executers Services in SODGM*

The Executer in SODGM system consists of numbers of services, each one of them perform specific tasks. The Executer functions are : Request Register machine to SODGM , Connect and disconnect executer from SODGM , Monitor Executer Status and send information to the load monitor in manager and Execute jobs for users and send back the result to manager. The Executer Services are:

1) Executer Registration Service

Executer Registration collect information about the executer machine then send this information to executers manager service to register this executer into the system. In addition to collected information about the executer, the user can specify

some information such as user name, password, operating system, port number, etc.

2) Job Execution Service

The Job execution service is used to execute the job in the executer machine and then return results and output files to the job manager service in manager machine.

3) Information Collector Service

This service used to retrieve static information about each executer such as operating system type, memory size, CPU speed, hard disk size, IP address .etc. it called locally from executer registration to collect information about executer to send this information to the manager.

4) Executer Status Service

This service used to provide the manager with the information about status of the executer . It retrieve information such as executer CPU load , memory load and available hard disk from the machine. Load monitor service ask this service in periodic time about executer status

C. *User Services in SODGM*

The User module in SODGM system consists of numbers of services, each one of them perform specific tasks. The user interface functions are: Request Register user machine to SODGM, Connect and disconnect to the SODGM, Submit jobs and it required input file to SODGM and Managing and monitoring submitted jobs. The user services are:

1) User Registration Service

User Registration collect information about the user machine then send this information to executers manager service in the manager to register this machine into the system. In addition to collected information about the machine, the user can specify some information such as user name, password, operating system, port number, etc.

2) Job Submission Service

Job submission is a user-oriented interface in the user machine. It allow user to submit the job to the manager. The user needs to provide job name, file location, necessary parameters for execution such as required input file, name and path of the output file. Users are interfacing with the SODGM system through this service. It receives a job request from user, then submits the job to manager service in the manager and migrate the required job files to the manager.

V. SODGM IMPLEMENTATION AND EVALUATION

In order to validate our approach, we implemented a services oriented manager by using web service technologies. Our prototype is a Microsoft .NET application developed in C Sharp (C#) and Windows Foundation Communication (WCF), which implement web services with Visual Studio 2005. Also, we implement the system Database tables and their relationships using Microsoft SQL Server 2005. It is a relational database management system (RDBMS) produced by Microsoft.

To evaluate the SODGM system, we made various experiments to measure the efficiency of SODGM scheduling factors. We tested that our system performs its functions by managing jobs over available executer machines to balance the load over them. We preformed experiments test on desktop computers in the Computer Laboratory in Computing and Information Faculty in king Abdul Aziz University. The computers that were used in the test are 16 computers connected with each other in a local area network (LAN).

The jobs in SODGM denote applications executable files that the SDGM system schedules and runs them. Jobs are computational jobs that need high performance requirement in processing. In this research, we focus on jobs that need high CPU speed and high CPU frequency for running. These jobs take a lot of time to run in one computer. We want to reduce the execution time by distributing the jobs over available executer nodes. Jobs in our experiments are chosen from scientific and research problems such as security algorithms, numeric algorithms and cryptography. The jobs are hash function, Compute prime numbers, Sorting algorithms, Random numbers, Encryption algorithms.

To test the efficiency of the distributed manager over the centralized manager, we compared the load of the system when the numbers of jobs and executers nodes in the centralized manager and in distributed manager are increased. In addition, we compared the average waiting time and the average execution time.

Table 1: System Load in centralized and distributed managers in minutes

# Jobs	Number of Managers	
	1M and 6E	2M and 6E
2	36.5	36.3
3	42.6	41.5
4	43.7	42.9
5	45.25	43.2
6	48.5	46
7	49.9	46.8
8	52.3	48.7

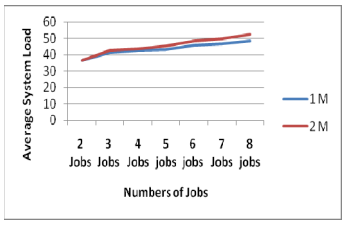

Figure 4: System load in Centralized and distributed manager

We perform experiment using two managers with six executer nodes and compute the execution time for jobs. The results of experiments are shown in figure 4. Comparing the results from

these experiments with result when using one manager, we find that using distributed manager gives less execution time comparing with using one manager (centralized manager) especially when numbers of jobs increased.

Table 2: Execution Time in Centralized and Distributed Manager

# Jobs	Number of Managers	
	1M and 6E	2M and 6E
2	7.05	7.94
3	11.18	11.14
4	12.37	12.06
5	16.5	16.31
6	31.17	30
7	39.4	37.9
8	40.34	38.13

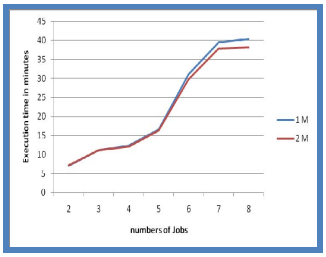

Figure 5: Execution time in Centralized and Distributed manager

The waiting time is decreased when using more than one manager (a distributed manager) because the jobs are distributed over managers and this makes management of jobs is easier than in a centralized manager.

Table 3: Waiting Time in centralized and distributed Managers

# Jobs	Number of Managers	
	1M and 6E	2M and 6E
2	0.35	0.33
3	0.4	038
4	1.39	1.25
5	5.5	4.1
6	9.31	7.90
7	15.25	12.18
8	24.26	18.37

As shown in Table 3, using a distributed manager in a large number of jobs makes the system to run jobs with a smaller execution time i.e. it gives good performance in minimizing execution time compared to a centralized manager.

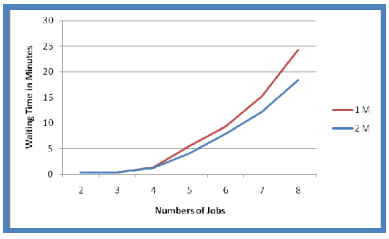

Figure 6: Waiting time in Centralized and Distributed managers

VI. CONCLOUSION

In this research, we introduced a new service oriented grid manager (SODGM) that has been built based on web services technology for managing grid systems. SODGM has been designed and implemented. The benefits of SODGM are: increasing available computing power, providing a flexible accessing to resources, maximizing the use of new/existing resources, running and managing jobs in a trustable, and providing simple startup functionality. In addition, SODGM provides flexible, easy-to-use and simple interfaces for registering to the system and submitting jobs. Finally, SODGM hides the complexity of the grid to users.

SODGM can be installed on a single machine or each web service of the manager can be deployed on a different machine to increase SODGM performance and scalability if the numbers of both users and jobs are increased. This means that the performance of SODGM is not affected (decreased) by jobs scalability. SODGM can be used as centralized manager architecture or hierarchical manager architecture. This means that SDGM has decentralized manager advantages, which include scalability, fault tolerance and peer-to peer functionality. SODGM manages jobs and executer nodes with good performance by minimizing execution time and waiting time. It also increases system throughput. It utilizes available executer nodes in an efficient way.

REFERENCES

[1].Foster, I., Kesselman, C., and Tuecke, S. (2001) "The Anatomy of the Grid: Enabling " , Scalable Virtual Organizations. *Int. J. Supercomp. App.*, 15(3): PP.200–222 .

[2] Mowbray and Miranda (2007) "How web community organization can help grid computing", International Journal of Web Based Communities, v 3, n 1, PP. 44-54

[3] Baker, M. Buyya, R. and Laforenza, D. (2002) "Grid and grid technologies for wide-area distributed computing," *Software: Practice and Experience*, vol. 32, no. 15, pp. 1437 – 1466.

[4] Globus Alliance. Globus. Web Published, 2007. Available online at: http://www.globus.org/ (accessed December 1st, 2009).

[5] Foster I, Kesselman C. (1997) "Globus: A metacomputing infrastructure toolkit", *International Journal of Supercomputer Applications* 1997; **11**(2): PP. 115–128.

[6] Condor. The Condor project. Web Published, 2007. Available online at: http://www.cs.wisc.edu/condor/ (accessed December 1st, 2009).

[7] Grimshaw, A. S. and Wulf, W. A. (1997) "The legion vision of a worldwide virtual computer ". *Communications of the ACM*, **40**(1), PP.39–45.

[8] Buyya, R. Chapin, S. and DiNucci, D. (2000) "Architectural Models for Resource Management in the Grid", Grid 2000, Bangalore, India.

[9] Qadir, k. et al . (2008) " System Architecture for Efficient Grid Resource Management" , 2008 4th International IEEE Conference "Intelligent Systems".

[10] Foster, I. and Kesselman, C. (2003) " *The grid: blueprint for a new computing infrastructure* ", chapter 2, pages 15–52. Series in Computer Architecture and Design. Morgan Kaufmann, 2nd edition, December 2003.

[11] Web Service activity web site in W3C , http://www.w3.org/2002/ws/

[12] Kertész, A . and Kacsuk, P. (2009) " Grid Interoperability Solutions in Grid Resource Management " , IEEE SYSTEMS JOURNAL, VOL. 3, NO. 1, MARCH 2009 , PP. 131-141.

[13] Chapman, C., et al. (2004) "Condor Services for the Global Grid ", National Environment Research Council, 2004 .