

Multi-users Coordinated Sharing Grid Resource Management System

Zheng Wang¹ Mei Dong²

¹School of Computer Science Xi'an Shiyou University Xi'an, Shaanxi 710065, China

²School of Foreign Languages Xi'an Shiyou University Xi'an, Shaanxi 710065, China

Abstract. The fundamental task of grid resource management system is to supply on-demand resources service for various users and applications by aggregate heterogeneous, dynamic and increase resource. Most current grid resource management system is exclusive which brings two sides of deficiency distinctively. We bring forward and realize a multi-user coordinated sharing resource management system, which is a utility oriented resource management system to support resources sharing by multi users or multi workload schedulers, thus extendable resource serving capability can be improved. When the system is used in an "intelligent building" and "intelligent community", the IT cost of the community enterprises can be reduced significantly.

1 Introduction

Grid is an open distributed system deployed on the Internet, with an objective to establish dynamic virtual organizations in a distributed, heterogeneous and autonomous environments, thus to achieve coordinated resource sharing and collaboration cross multi-autonomous domains, and to provide grid users or applications with aggregated resources as required in limited time, so as to solve diverse user application problems^[1]. The key question concerned by a grid resource management system has turned from "how to run a (or a batch of) task on all resources in a rapid way?" to "how to provide services to more users as required in order to improve customer satisfaction and then enhance resource revenue?". To this end, grid resource management systems shall firstly be capable of aggregating all resources cross multi-management domain on the Internet, coordinating and using aggregated resources, to provide lots of users with consistent, dynamically deployed and efficiently coordinated resource services.

When it comes to review some existing grid resource management systems, such as Globus^[2], Legion^[3], Nimrod-G^[4], AppLes^[5], Netsolve^[6], etc, it can be found that all of these systems are usually resource oriented their problems mainly lie in the following aspects: (1) Coupling load management and resource management tightly, leading a grid contained resources could only serve for a certain single application load; (2) When allocating resources, it is seldom considered to allocate the resources to users and its applications that needs the resources most and brings benefits most, usually resulting in unreasonable and unfair resource allocations, and leading to the reduction of performance of user application and

satisfaction; (3) Provision of resources is considered voluntary and unconditional. The application of resources is as required and for free, such resource sharing manner is reasonable in early distributed systems, as owners and users usually belong to a same organization, however, it doesn't apply to grids. Dedicated resources utilization is liable to leading to demand inflation of users resources and insufficient enthusiasm of providers of the resources, thus causing reduction of practical value and application prospect.

Then a multi-user coordinated sharing grid resource management system MCSGRMS is put forward to and implemented in the paper, with an aim of solving coordinating and sharing problems of various users application load to resources, achieving a utility-oriented resources management system^[7]. The raised grid resource management system MCSGRMS, along with its design philosophy, systematic structure and system implementation is elaborated in Section 1. And a SOA application example of MCSGRMS is given in Section 2, as well as test result of the application on the prototype system, verifying the validity of the system. Finally, a summary is given at the end of the paper.

2 Multi-user Coordinated sharing grid resource management system MCSGRMS

2.1 Design philosophy

(1) MCSGRMS (Multi-users coordinated sharing grid resource management system) is a customer sharing - oriented grid resource management platform, which a relative universal infrastructure for computer resource

Corresponding author: wangzheng@xsyu.edu.cn

management, and allows diverse users to run on the platform simultaneously, sharing resources in grids, all kinds of load dispatchers or load management systems could also run on which, high-performance applications, batching application, SOA application, J2EE/.NET and other various of applications are supported. Core philosophy and characteristics of which is as follows:

(2) Decoupling resource management and load management make functional interface of the resource management layer much clearer and more explicit, which is no longer like the traditional resource management system that couples resource management and load management tightly.

(3) Diverse user applications can be compared to resource consumers, to support coordinated sharing between resource consumers and resource providers ;

(4) Customers' resource sharing and allocation are strategy based, which may be restrained by centre coordinator in accordance with certain strategy, providing customers with necessary resources .

(5) Resource discovery and monitoring cross multi-organization are supported; resources could be found and obtained by using command line interface provided by MCSGRMS or API access interface.

(6) Life cycle management is simplified; each kind of load management system could co-exist on MCSGRMS, sharing and using MCSGRMS provided resources.

(7) A group of extensible services are provided, various user applications could use resources on the group of services, while heterogeneous and autonomous resources are virtualization and aggregated under the group of services.

As shown in figure 1, decoupled MCSGRMS orients to resource management exclusively, to implement the strategy of resource provider, which mainly concerned optimization and usage of resources and other problems are, e.g. when and which resource could be used, how much resources could be gained by which user, how to configure extra resources when it is unavailable. While all kinds of application load managers operated on the MCSGRMS obtain resources from MCSGRMS directly, dispatching problems of application load are mainly concerned, e.g. which working unit is executed firstly, which working unit will operate on which eligible resource, how to protect QoS objective (response time, handling capacity, delay time) of user application, etc.

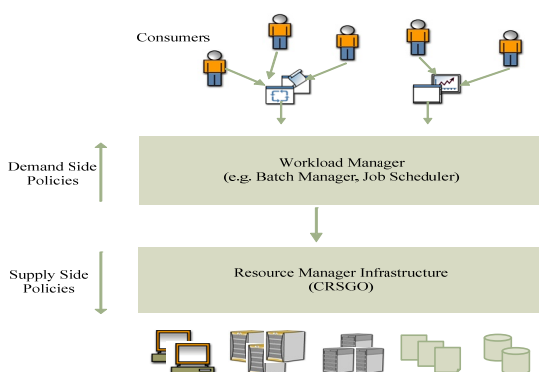


Fig.1 The design philosophy of decoupling resource management and workload management

2.2 Systematic structure

The systematic structure of a grid resource management system is shown in Figure 2; MCSGRMS lies between local resource manager layer and application load manager layer, and crosses three hierarchies of systematic structure: management service layer, extended service layer and API/SDK layer. Management service layer includes provider Agent and coordinator Agent; extended service layer include consumer Agent, event service, portal service, security service, accounting service, fault-tolerant service and a group of software components (with part of software components shown in the figure); API/SDK layer provides functional interfaces API of corresponding service. Two layers are placed below MCSGRMS. They are respectively grid device resource layer and local resource manager layer. There are also two other layers above MCSGRMS, which are respectively application manager layer and user application layer. A lot of applications can be supported by MCSGRMS. Each application manager operates on MCSGRMS, acquiring resources in grids via API or service interfaces and are used by upper users via self-dispatch.

MCSGRMS has 3 core functions, they are respectively discovering, allocating and executing, which correspond to 3 phases of grid resource management respectively. The first phase is discovery phase, which is to look for available resources for users; the second phase is allocating phase, which is to allocate resources for users according to sharing strategy; the third phase is execution phase, which is to execute user applications on the allocated resources, recording resource usage status and releasing resources.

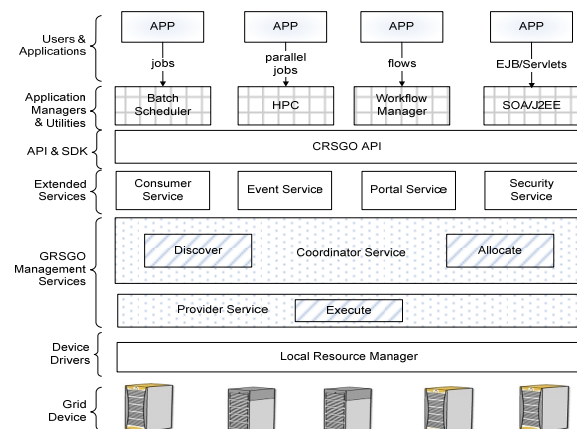


Fig.2 The system structure of MCSGRMS

MCSGRMS employs an extensible resource management systematic structure, and a plug-in method is used for virtualization of resources, i.e. virtualization function of each kind of resource corresponds to one plug-in add-in, each kind of resource is managed by the system through loading of all kinds of plug-ins, discovery, allocation and management tasks of each kind of resource is compatible with and completed by core functional components. At present, only plug-in for computing resources is implemented in MCSGRMS, the plug-in is in charge of accomplishing vitalization task of the computing resources, by abstracting computing resources

into slots that could be used in sharing, that SOA/J2EE oriented transactional applications can be supported, that is to say user application can be modeled into a service, and a service can have several service instances operating on grid computing resources.

2.3 System implementation

Core software components of MCSGRMS is consisted of 3 parts, which are coordinator agent, provider agent and consumer agent, among which, although consumer agent lies in extended service layer, due to its great convenience for upper users to use grids, it is incorporated into core components. Core framework of MCSGRMS is shown in Figure 3.

CoNsumer agent (CNA) takes charge of requiring resources for users and executing user application, which mainly includes application requirement planner and service controller, two sub-components, application requirement planner is in charge of generating corresponding resource demand according to characteristic attribute and executing requirement of user application, and sending the resource request to coordinator agent. SOA/J2EE and other application load managers are supported, such as Websphere^[8] and Symphony^[9], that is to say each user application corresponds to one service, and the service possesses a related configuration file, which employs XML representation, to fully configure attributive character, controlling strategy and allocation constrain of services. Application requirement planner will load configuration files of a service, generate relevant resource request according to model parameter of the service and then submit to coordinator agent. Service controller takes charges of execution of controlling service. Some basic controlling operation such as registration, startup, termination and update to services and service instance could be made.

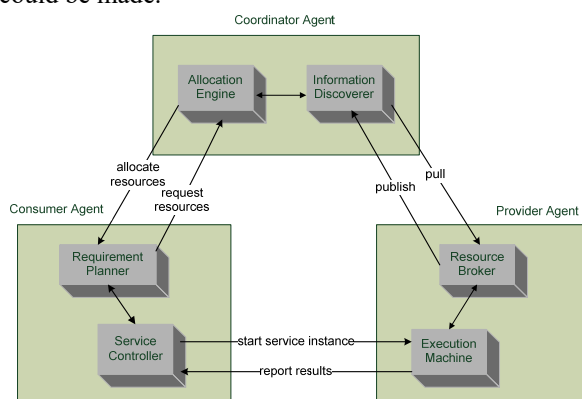


Fig.3 The structure of the core framework of MCSGRMS

CoorDinator agent (CDA) is in charge of allocating resources for user applications. Available resources in grids should be found prior to allocation, CDA includes two sub-components, and they are respectively resource allocation engine and resource information discoverer. Coordinator agent operates in virtual organizations of grids, which may aggregate resource providers in the grids and allocate resources for users according to constrain of sharing strategy in general. A resource allocation engine receives resource allocation request sent

by consumer agent, allocates resources periodically, and informs consumer agents the allocation results. Some consumers' resources will be retrieved in the allocation process. Notifications for resources retrieve will also be send to consumer agent correspondently, submitting allocation request and receiving allocation (retrieving result) notification is a asynchronous process. Resource information discoverer takes charge of aggregating and finding available resources in grids, receiving resource finding requests sent by consumer agents or internal allocation engine, and interacting with internal resource allocation engine, so as to choose suitable resources for allocation engines, and help allocation engines to make allocation decisions.

PProvider agent (PRA) is in charge of managing possessed resources and executing user application, which includes two functional sub-components, which are respectively resource broker and execution machine. There are several provider agents in a virtual organization; and each provider agent provides virtual shared resource capacity externally that meets certain sharing strategy for consumer using. Resource broker is in charge of monitoring resources status, and sending resource information to coordinator agent. Resource broker can perform 3 main tasks: resource virtualization, monitoring resource status, sending resource information to coordinator agent. At present, only one type of plug-in for computing resource is implemented in MCSGRMS. The plug-in provides access monitoring interface to the computing resource, virtualization of resources and monitoring tasks are all accomplished by firstly loading plug-in, and then making use of functional interfaces provided by plug-in. Resource broker takes charge for sending possessed available resource information to coordinator agent, and also receiving query request from coordinator agent reporting the current resource status to coordinator agent.

Execution machine receives executing request sent by consumer agent and builds executing environment for service instances after authenticated security check is completed, and then encapsulates it into a whole life cycle process that are active, operated on computing resource slots being allocated and controlling activities. Main functions of an execution machine include controlling activity, monitoring activity and reporting activity status to consumer agents, as well as gathering resources using status when activities are being operated.

3 SOA application examples of MCSGRMS

MCSGRMS uses decoupling resource management and load management design philosophy, thus achieves a relatively universal grid resource management platform and enables coordinated sharing of resources in grids across multi-load users, to meet their demands. The SOA described in this section is a service-oriented systematic structure, and an application instance with scheduler symphony deployed on MCSGRMS. Performance test results under different configurations are also given. SOA is a service-oriented application which accomplish user application. by encapsulating enterprise logic into one or

more services . SOA scheduler symphony, such as Symphony for example, is in charge of mapping and dispatching of a message to service.

In comparison with traditional scheduler symphonies that deployed on clusters, the MCSGRMS deployed Symphony is obviously of the three typical advantages: 1) Acquiring corresponding resources in grids according to application need, thus to start service instances as demanded, achieving quick response of services to user request; 2) By using MCSGRMS provided service instances restart function, high availability of services could be achieved; 3) By using resource reallocation function of MCSGRM, expandability of services could be achieved.

Symphony will start service after it gets the allocated resources from MCSGRMS, and a session will be started after service being started, a session includes a group of task, input of the task is a request message, and an output will be generated after the task is finished. Execution of a SOA user application is indicated in Figure 4, including the status of closed sessions, aborted sessions, done task and error task.

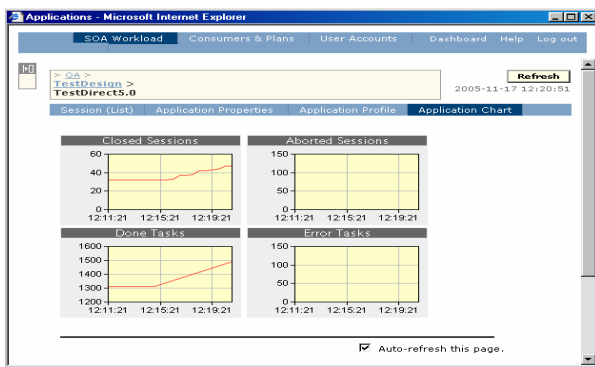


Fig.4 The load operation situations of SOA application based on MCSGRMS

We conducted system integration test to SOA application that operates on MCSGRMS. By remote installation, MCSGRMSs are installed on 200 workstations at a time and each one is equipped with a Pentium 2.5GHz processor and uses a Windows or Linux OS. As backup/recoverable SOA task influences application performance seriously, such parameter is carefully considered when test are being conducted. We examined execution status of SOA applications from two aspects: task run time and task sending throughput, and examined resources utilization rate of MCSGRMS from CPU efficiency. Performance result of system integration test are shown in Table 1

Table. 1 The performance test result of 200 CPU

OS	Recoverable?	Task RunTime	CPU Efficiency	Task Sending Throughput
Windows	Yes	1 sec	99.56%	1047
Windows	No	1 sec	99.54%	1775
Linux	Yes	1 sec	99.75%	2000
Linux	No	1 sec	99.72%	2914

It can be seen from the test that resource utilization rate of MCSGRMS is always kept at a high level. Besides average resource utilization rate under a 200 CPUs testing condition reaches 99.64%, and task sending throughput as well as task run time of SOA application have been improved to a great extent.

The system can be now used in intelligent buildings integrated Cloud computing and IoT technologies, thus makes Cloud and IoT surrounded intelligent communities. With the increasing integration of Cloud computing and IoT technologies and intelligent building, sharing of Cloud and IoT intelligent community is bound to increase year by year. It can be predicted that China's development in intelligent building and intelligent community in future will be great. This system will help IT staffs of enterprises in communities to response to ever-changing business need immediately, making real-time allocation and re-allocation of resources; thus improve allocation efficiency of IT resources and extensible service capacity of resources greatly beyond imagination and eventually reducing total IT cost of the whole intelligent communal enterprises.

4 Summaries

A multi-user coordinated sharing grid resource management system MCSGRMS is built and implemented in the paper, which can provide virtualization and automation functions, offering all kinds of application a grid platform that may share all of IT resources. It is a relatively universal infrastructure for grid resource management. In addition it is indicated by practical application that when compared with former grid management system, the system supports more user application, with high-performance application, batching application, SOA application and numerous application patterns can be operated on MCSGRMS simultaneously, and resource allocation efficiency is higher, thus user satisfaction is obviously improved. Apply the system to intelligent buildings and intelligent communities, total IT cost of the whole intelligent communal enterprises will be reduced greatly.

References

1. CHAPIN S, CLEMENT M, and SNELL Q. A Grid Resource Management Architecture, Strawman 1[M]. Manchester : Grid Forum Scheduling Working Group, 2009.
2. FOSTER D, KESSELMAN C. Globus: A metacomputing infrastructure toolkit[J]. International Journal of Supercomputer Applications, 2007, 11(2): 115-128.
3. GRIMSHAW F, WULF A. The Legion vision of a worldwide virtual computer[C]. Communications of the ACM, 40(1), 39-45, Paris, 2007. ACM Press
4. BUYYA R, ABRAMSON D, and GIDDY J. Nimrod-G: Architecture for a Resource Management and Scheduling Systems in a Global Computational Grid[C]. The 4th International Conference on High

- Performance Computing in Asia-Pacific Region,11(5), 331-335, Singapore, 2010.
5. BERMAN F, WOLSKI R, FIGUEIRA S, et al. Application level scheduling on distributed heterogeneous networks[C]. Proceedings of Supercomputing, 2(3), 111-116, Scotland, 2012.
 6. CASANOVA H,DONGARRA J. Net Solve: A networks server for solving computational science problems[J].International Journal of Supercomputing Applications and High Performance Computing, 2007, 11(3): 212 - 223.
 7. PADALA P, Kang G. Adaptive Control of Virtualized Resources in Utility Computing Environments[C], EuroSys'07, 6(5), 89-95, Belgium, 2012.
 8. Websphere,<http://www.ibm.com/developerworks/cn/websphere/geoconst.html> .
 9. Platform Symphony: Enterprise Grid Solutions for Financial Services, <http://www.platform.com>