

Semantic Enhanced Self-Configuring Grid Framework

Hao Li
School of Software
University of Yunnan
Kunming, China

Gehao Lu
School of Software
University of Yunnan
Kunming, China

Joan Lu
School of Computing &
Engineering University of
Huddersfield Queensgate,
Huddersfield HD1 3DH

Shaowen Yao
School of Software
University of Yunnan
Kunming, China

Abstract

This paper reviews the advantages and disadvantages of traditional 5-level Grid architecture and new OGSA standards. It also summarizes the current popular Grid resource schedule algorithms and frameworks. The paper points out the problems and trends faced by the Grid communities. It emphasizes that there must be a self-configuring Grid to support the dynamic changing complex systems. The paper proposed a new semantic enhanced Grid resource allocation framework which is highly flexible and autonomous. Some supporting prototype verification and formal methods are discussed as well. As a PhD proposal, the paper gives work plan and possible outcome at the end.

Keywords: SOA, Autonomic Computing, resource allocation

1. Introduction

Nowadays, we need information anywhere and anytime. With the development of the Internet, we urgently need a new way to reach the target which can make everyone get any necessary information at anyplace. How can we share the resource which is located at anywhere in the world and how can we get what service we really want through the Internet? This proposal focused on the very hot point in the IT area, the grid computing and SOA (Service-Oriented Architecture) [3]. How can we put them working together? According to researches of many

international organizations (such as OGSA,); there are some mature ways to do that. But still now how can we make the grid service (grid computing + SOA) autonomic work, such as self-configuring, self-optimizing, self-healing and self-protecting. There are still some jobs need to do. This article aims the self-configuring and what is the challenge of self-configuring. Grid is “A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to the high-end computational capabilities” (Foster & Kesselman) in 1998. The evolution of Grid Computing is broadly classified into three generations. At the beginning of the concept grid computing is the U.S.A. government want to union distribute high performance computers and databases and other devices can work together. It makes the whole system looks like a super computer.

1.1 Some of the basic concepts of the grid computing

“Grid computing equates to the world’s largest computer” [1]. Actually the grid computing is the structure which is distribute, sharing resource and cooperation that is working together access the network. Even somebody said: “The grid is next generation Internet.” In the early definition of grid is “A computational grid is a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to the high-end computational capabilities” (Foster & Kesselman) in 1998. The evolution of Grid Computing is broadly classified into three generations. At the beginning of

the concept grid computing is the U.S.A. government want to union distribute high performance computers and databases and other devices can work together. It makes the whole system looks like a super computer.

1.2 Autonomic Computing

Human faces an embarrassing dilemma when more technologies invented to improve their lives, that is, the capacity of human brain is limited, compared to the endless scientific and technical innovations. The growing complexity of the computer system is a good example to illustrate this point of view. When a system grows to the size that spans the boundaries of many organizations, the task of learning the system becomes an impossible mission because even skilled people cannot understand every details of every part of the huge system. In this sense, we can see that the trend of computing must evolve from the initial calculation centric to intelligent network centric. Autonomic Computing is an initiative started by IBM in 2001. Its ultimate aim is to create self-managing computer systems to overcome their rapidly growing complexity and to enable their further growth. [2] In IBM's vision, the Autonomic Computing is composed of three major areas: self-optimizing, self-protecting, self-healing and self-configuring. The ultimate goal is to create and deploy self-managing infrastructure technologies, to reduce complexity, lower cost of ownership and increase reliability, to establish an architectural framework in Autonomic Computing. Providing technologies reduces the cost of managing systems that is automating. The architecture of autonomic computing is under construction. Currently architecture should contain the following capabilities: solution knowledge, common system administration, problem determination, autonomic monitoring, complex analysis, strategies for autonomic managers and transaction measurements. To support those capabilities, the architecture can modeled as seven conceptual elements: decision-making contexts, controlled-loop structure, managed elements, autonomic manager, autonomic manager

collaboration, autonomic manager knowledge, self-managing systems.

For example, Océano Project which makes system track performance metrics, aggregate and correlate metrics to SLA violations, and orchestrate reconfiguration. ABLE Autonomic Components use Java-based agent framework and AI component library. It makes the use of agent builder, test and debug tools, multi-agent platform. The system can add adaptively through on-line machine learning and policy-based behavior using rules-based knowledge representation. The system also adds flexible, reactive, and deliberative goal-seeking behaviors and distributed hierarchical communication and feedback control [4].

In the real world, Grid is the actual complex system based on large scale network and varieties of devices. How to apply the Autonomic Computing concepts to the Grid application is the successful sign for both Autonomic Computing and Grid Computing. With the feature of autonomic computing, Grid can greatly reduce the work and complexity associated with a large system. It also can respond better to sudden changes in the system and adjust setting appropriately.

2. Existing Grid architecture

2.1 5-layer architecture

“The 5-layer Grid Architecture” [5] based on the protocol was important at the beginning of the grid. “The 5-layer Grid Architecture” emphasizes the protocol is the key issue. Based on this protocol can make the virtual organization sharing the resource and cooperation each other. At the same time, it can develop and manage the relationship among the organization. This protocol can make the grid develop to the flexible, cooperation, standard and improve sharing resource.

As the Web launches the HTTP and HTML to communicate, the protocol as the core of the grid, it makes sharing the resource come true. The protocol

in the 5-level Grid Architecture, it is not the collection of the all protocols; actually, it is according to the different functions to divide into different levels. Every level supports the upper level. The lowest of the structure is Fabric which is face to the real physical resources. The fabric offers the management and control interface to the upper level. The second level is Connectivity which is responsible to the security and communication for the physical resources. The upper level is Resource level, which reflects the abstract characteristic of the resource. The fourth level is Collective level, which is responsible to collect all the individual resource to solve heterogeneous resource problems.

The top level is Application which just cares what kind of the resource available. The application level is far away from the real resources. Actually, all the resources are offered by the Virtual Organization. From the view of the structure, the 5-level Grid Architecture looks like the TCP/IP protocol. Each level does different jobs. The architecture of the 5-level Grid looks like an hourglass which means each level has different amount of protocols. For these core protocols, they are implementing all upper levels to reflect for themselves. At the same time, it has to implement the reflection to all the lower levels. The core protocol must support by any network computing environment. Therefore, the amount of core protocol must not be too much. That is the reason why the 5-level Grid Architecture looks like an hourglass. In the real model, the core protocol consists of Connectivity and Resource.

The 5-level Grid Architecture emphasizes the protocol as a core of the structure. It implements resource sharing at higher level. It is not only simply exchange the files, but also emphasizes directly to access the computer, software, data and other resources. This is a dynamic sharing which crosses different organization and management even different physical locations. This sharing is deeper, wider, dynamic level which is under different kinds of controls. From the view of the 5-level Grid Architecture, it emphasizes how to construct the grid physically from the protocol and standard. Though

they are mentioned sharing resources, it is also very difficult to realize the service over the grid. Therefore, after the 5-level Grid Architecture, from the IBM and Foster has a new generation grid concept: OGSA (Open Grid Service Architecture)

2.2 Open Grid Service Architecture (OGSA)

Because the dynamic coordinated resource and the application of Grid Computing is complex. It is difficult to schedule all the resource properly and make them cooperate to do some job. The introduction of Open Grid Service Architecture (OGSA) is to support creation, maintenance, ensembles of services maintained by virtual organizations (VO). OGSA builds a platform which is composed of Open Grid Service Infrastructure (OGSI), OGSA platform services, and some other specialized domain-specific services. The goal of OGSA is to define the core components and platform specific binding so that Grid Computing can deliver better QoS. There are five layers in the current OGSA: native platform services and transport mechanisms, OGSA hosting environment, OGSA transport and security, OGSA infrastructure (OGSI) and OGSA basic services (meta-OS and domain services). OGSI defines essential web service specification and description mechanism. It provides support to the basic behavior like service discovery, service instance creation and service management in order to achieve maximum interoperability. The Grid service is a stateful web service which contains state and service data compared to the traditional web service. The Grid service is described using GWSDDL instead of WSDL.

In addition to OGSI, the OGSA basic services provide OGSA meta-OS services and OGSA domain services. The Open Grid System Architecture (OGSA) Common Management Model represents the IT resources such as disks, file systems and operating systems. Service domain components provide registration, collection, routing, selection, interoperation, transformation, composition and

automatic orchestration of services. This domain concept is especially useful when single service is from a totally different heterogeneous hardware, systems and applications. The OGSA policy service provides rules or policies to manage and control access to any grid service. Those policies cover security policies, workload balance policies, business process policies and so on. Security issues are also major challenges faced by OGSA. The security architecture of OGSA builds some security mechanism to support common security models and protocols through Grid security services. In the real world, the Grid user has to consider issues like cost allocation, capacity analysis and service pricing. OGSA defines interfaces and behaviors for those metering and accounting requirements by providing metering services interface, rating services interface, accounting services interface and billing services interface. Logging is generation and management of log message in most of the systems. The OGSA logging is based on a distributed architecture and it separates the OGSA logging implementations to a logging service. One of the bottlenecks of distributed computing is distributed form of data. In OGSA, the complexity can be made transparent to grid applications through grid data virtualization services. This virtualization service helps distributed data hide their location, naming, distribution, and ownership.

3. Research Trends and State of Art

Even OGSA is focus on the service; there are still some weak points. The SOA is suitable to work on the grid. In the next generation, how to build an Autonomic computing is becoming a more important thing. According to the Joshy Joseph, Craig Fellenstein from IBM, the next generation grid will be work on the following areas: “

- Autonomic computing
- Business On Demand and infrastructure virtualization
- Service-oriented architecture and grid
- Semantic grids” [1]

In the autonomic computing area, Self-configuring (which is able to adapt to the changes in the system), Self-optimizing (which is able to improve performance), Self-healing (which is able to recover from mistakes), and Self-protecting (which is able to anticipate and cure intrusions) are the most important. Self-configure is an autonomic application system [2], it should be able to configure and reconfigure itself under varying and unpredictable conditions. With the development of the grid computing infrastructure, it is important to concentrate on how to realize the infrastructure self-configuring, such as server, memory, licenses and band wide. An automatic way to do that job is needed.

[6] proposes an integrated management and scheduling scheme for computational grid. It solves some pivotal and important questions such as resources heterogeneous and information dynamic. It proposes a mixed scheduling algorithm with the goal of the least number of missed real-time deadlines and load balance of grid resources, called LMLB and a real-time task selection algorithm called Weighted Priority Schedule Algorithm (WPSA) is adopted in real-time scheduling. It affords transparent support for high-level software and grid applications, enhancing the performance, expansibility and usability of computers, and providing incorporate environment and information service.

[7] Presents a general-purpose resource selection framework that addresses resource configuring problems by defining a resource selection service for locating Grid resources that match application requirements. It is called set matching, which extends the Condor matchmaking framework to support both single resource and multiple resource selection. Newer framework based its architecture upon Web services and SOAP. For example, the EU Data Grid Management Services [8] are a framework which is composed by components like Replica Location Service, Replica Metadata Service, Replica Optimization Service, Replica Subscription and high-level replica management. The following figure shows how the components interact to finish the

basic Grid tasks.

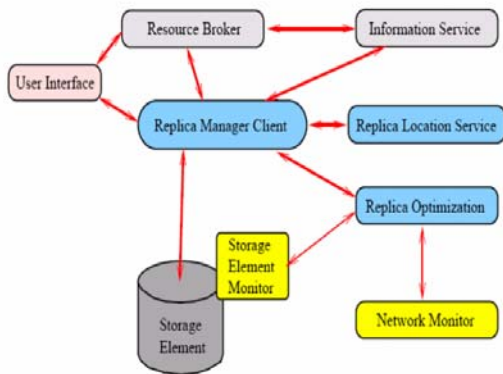


Figure 1: EU Data Grid Management Services [8]

4. The grid resource allocation

This paper focuses on the self-configuration of service-oriented grid architecture. Self-configuration aims at the ever-changing resource. It uses middleware platform of the grid service to offer the transparency and standard services. Self-configuration from different ways which are resource description, monitors, discovery and schedule to realize automatic.

Therefore, this paper will cover following study areas.

From the view of the mechanism, which is based on the current grid computing service system, the study will try to support self-configuration framework structure. The current resource manager is short of the flexibility and extension with the changing of the resources. Therefore, for the dynamic resource, the job scheduler, resource monitor, resource adapter and resource register need extend to satisfy the request of grid self-configuration.

The following figure shows the improved frame work which is based on the original one.

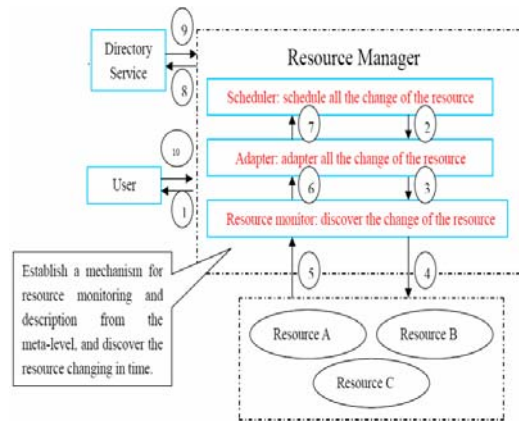


Figure 2: The grid resource allocation

To use the resource monitor discover change of the resource. To use adapter adapt resource changing. To use scheduler arrange all the available resources. Under the condition of interfacing the upper service and adapting the Resource describes language, it is necessary to build a mechanism which can satisfy a variety of basic resources, and that is the purpose of offering the best opportunities for the users.

5. Conclusion and Future work

1. The ontology description of resource and adaptation: in order to describe the dynamic resource changing and resource registering and management, an ontology descriptive mechanism is to be built for dynamic resources. As a meta-meta level abstract description, the ontology of resource is extended based on the current resource description language and it can be adapted to varieties of resources (such as Clustering, SMP etc). The mechanism is aim to form a solid foundation for monitoring, discovering, registering, allocating and scheduling the dynamic resource. The mechanism of the adaptive encapsulation using resource ontology supplies a unified infrastructure of resource description for the other part of the resource manager.

2. Resource monitoring and discovering: the purpose of studying resource monitoring and discovering in the environment dynamic resource is to offer the support of the resource management, job scheduler and resource of the RM

in the grid environment. Based on the ontology description of resources, from the Meta level to set up the resource monitoring description and monitor mechanism. It is important to discover the change of the resource on time and to offer the evidence for the job scheduler and dynamic resource arrangement. It is useful to set up a formal method modeling way to validate the study algorithm or mechanism, using some of the ideas from distributed computing and P2P.

3. The job scheduling and dynamic resource allocation: according to the feature of dynamical changing, improve the job scheduling and resource allocation algorithm for the current resource manager. Make the resource manager fit for the self-configuration of Grid Computing. The algorithm for Grid job scheduling and Grid resource allocation can adopt DHT (Dynamic Hash Table) technology in P2P computing or use the existing methods for job scheduling algorithm and resource allocation algorithm in other types of computing. Also, a formal method can be used to model this new resource manager and to verify the completeness of the targeted algorithms or mechanisms. The formal method is a good approach to analyze the performance of algorithms and mechanisms as well.

Reference

- [1] Joshy Joseph, Craig Fellenstein "Grid computing" Publish by Prentice Hall PTR. December 30, 2003. ISBN: 0-13-145660-1
- [2] IBM "what is autonomic computing" [URL] http://en.wikipedia.org/wiki/Autonomic_Computing, Last access time is 17/02/2006
- [3] Hao He "What Is Service-Oriented Architecture", <http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html>, Last access time is 07/02/2006.
- [4] Martin C Brown "Build grid application based on SOA" <http://www-128.ibm.com/developerworks/grid/library/gr-soa> Last access time is 07/02/2006
- [5] Sun Jian, Yin Xiaofeng, Qin Guangwei. Technology of Grid Computing and information sharing. Railway Computer Application. Volume 13, 4. No: : 1005-8451 (2004) 04-0005-04
- [6] Ran Zheng, Hai Jin. An Integrated Management and Scheduling Scheme for Computational Grid. Journal of Computer Science and Technology, Vol.18, No.4, 2003.
- [7] Chuang Liu, Lingyun Yang, Ian Foster, Dave Angulo. Design and Evaluation of a Resource Selection Framework for Grid Applications. Department of Computer Science, University of Chicago, Chicago, IL 60637, USA
- [8] Diana Bosio, James Casey, Akos Frohner, Leanne Guy, Peter Kunszt, Erwin Laure, Sophie Lemaitre, Levi Lucio, Heinz Stockinger, Kurt Stockinger. Next-Generation EU DataGrid Data Management Services. Computing in High Energy and Nuclear Physics, 24-28 March 2003, La Jolla, Californ