An Engineering Computation Oriented Grid Project: Design and Implementation[1]

Xianqing Wang[1, 2], Qinhuai Zeng[2], Dingwu Feng[2], and Changqin Huang[2, 3]

[1] Guangdong Institute of Science and Technology, Zhuhai, 519090, P.R. China
[2] Hunan University of Arts and Science, Changde, 415000, P.R. China
[3] College of Computer Science, Zhejiang University, Hangzhou, 310027, P.R. China
xqwang_kgy@126.com, djzqh@163.com, cqhuang@zju.edu.cn

Abstract. This paper describes a Service-bAsed Grid project for Engineering computation, named SAGE. Based on the Globus toolkit, a grid-service-based architecture oriented to engineering computation is presented. To give grid users good usability, task definition and resource discovery are visually conducted. Whilst, a Quality of Service (QoS) driven user-centric scheduling strategy is proposed, two scheduling methods and steering-enabled visual interfaces are applied for different types of grid users. Result processing can be visualized in the aid of a PC-cluster and a stereopticon. The practices suggest that these mechanisms improve the convenience and QoS.

1 Introduction

Grid computing has a promising future in large-scale scientific computation. The Globus toolkit [1] is the most popular grid environment and de facto standard, and it has adopted OGSA [2] architecture to provide applications with grid service level at present. As a special type of large-scale computation, engineering computation is very hardy-solved because of requirements for many valuable computing resources or specialist instrumentations, grid computing is a suited computing infrastructure for the high capability of distributed and heterogeneous resources sharing. However, grid-based engineering computation needs to solve some important issues, such as narrowing the gap between currently deployed grid services and the would-be user community.

2 Overview of Proposed Architecture

To meet requirements for scientists in engineering computation field in grid environments, we propose the SAGE project (Service-bAsed Grid project for Engineering computation) that aims to use Grid technology to establish an enabling environment for large-scale scientific and engineering research. The overview of architecture is shown in Fig.1, it consists of four components: a) The interface component is a portal

[1] A Project Supported by Scientific Research Fund of Hunan Provincial Education Department, China (Grant No. 04A037).

V.S. Sunderam et al. (Eds.): ICCS 2005, LNCS 3516, pp. 887 – 890, 2005.

of the system, It not only provides a visual interface for development and steering of grid-based applications, but also gives these system administrator a friendly management portal. It comprises a task submission window and two types of grid scheduling interfaces. b) Task-level logic component is responsible for task management, community policy and user administration. It includes virtual organization register, user manager, task definition, task import and export, task submission, task scheduling, task monitoring, task rescheduling, file transfer, and result processing. c) Grid service logic part consists of all services and service management mechanisms. All services are divided into three types: advanced general service, engineering computation modeling and post-processing service, basic grid service. The former two types are constructed on the basis of the latter. Service management contains: service deployment, service location, service composition, service scheduling and service-level performance steering. d) Grid Security Infrastructure is the last hierarchy in our architecture. It is based on the Globus Toolkit's security mechanism, main characteristics exist: parallelized subtask-based authorization, community policy and task delegation management [3].

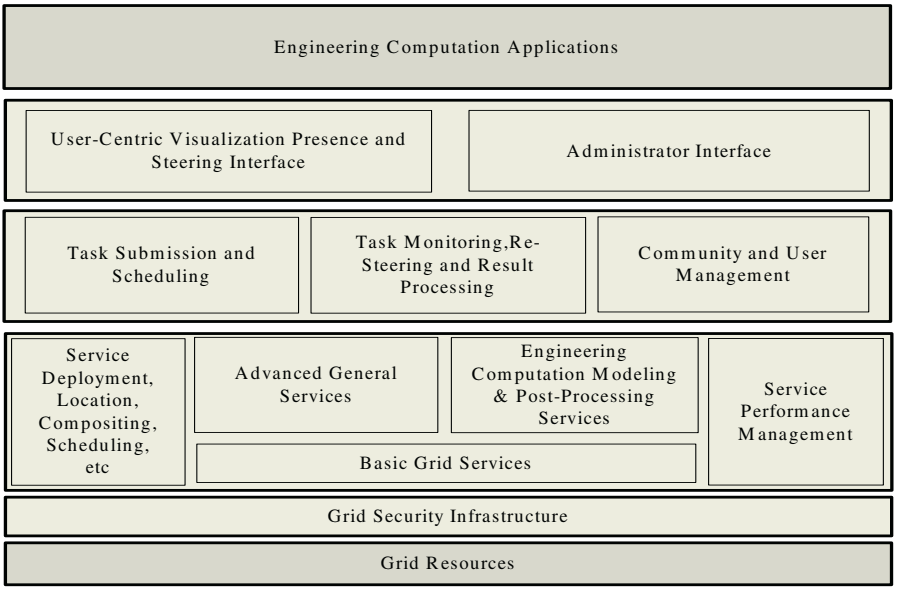

Fig. 1. Engineering computation architecture

3 Definitions of Grid Task and Resource Discovery

A session for each user is defined in the Grid. The session is created when one logs in, and destroyed after one logs out. In the session, a user defines computation task, possesses resources, monitors task execution and does post-process for the results. The Grid provides a graphical user interface to input a new task or to update an existing task. The data include executable program, resources and schedule requirements.

When the Grid accepts the data, an instance of task class is initialized and added to logical resources allocation queue. The task's state is initialized to "WAITING".

Resources information services provide two interfaces, a hierarchical resources tree, and a resources list. The content of the tree is the detailed information regarding hosts in virtual organizations, where local hosts are registered. It also shows the architecture of those virtual organizations. In the list, each item contains information of an individual host. The information on the list can be customized for display. Generally it consists of host name, node count, CPU count, speed and free percentage of CPU, free memory size, free secondary storage, network bandwidth and latency. In this project, resources are divided into two classes, computing power and visualization devices.

4 Task Scheduling and QoS Management

This architecture adopts aggregated QoS driven visual scheduling, the scheduling could be performed through two types of visual windows, scheduling and QoS sessions. These visual methods provide users with a direct awareness and a friendly interaction. A visual scheduling framework is shown in Fig. 2.

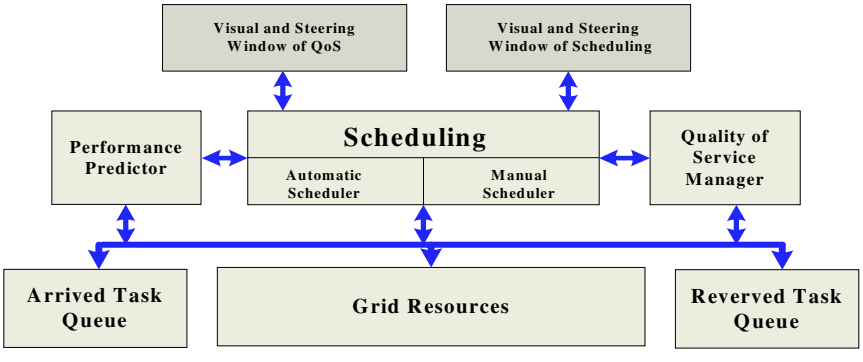

Fig. 2. A visual scheduling framework

According to different types of users, the visual scheduling is performed in both manual and automatic manner with a monitoring mechanism. In addition to the arrived task queue, a reserved task queue is used for arranging pre-scheduled tasks. Hence reservation-based scheduling can perform well. The scheduling algorithms during the automatic scheduling can be selected by users, as various conventional algorithms can be integrated into the system. The idea is based on the fact that different algorithms are suitable for respective environments and objectives.

In our architecture, a self-designed simple performance predictor serves for the scheduler by predicting and computing the previously mentioned performance metrics. It analyzes the performances of the tasks to be scheduled in advance; furthermore, it evaluates QoS parameters of scheduled tasks. QoS manager is responsible for accepting these required performance values from the input, for setting performance

threshold values that are conditions of triggering adjustment mechanisms and warning user, and for managing the events of post-scheduling and the interactions for a better performance. In the project, aggregated QoS is modeled based on four performance metrics. The relevant scheduling algorithms are based on Dual-Component and Dual-Queue Distributed Schedule Model (D^3SM), which is described in the literature [4].

5 Visualization for Results

The project supports the collaborative visualization of meshes and visual steering of CFD/CSM simulations. The visualization will be implemented by our on-going project named ViG (Visualization in Grid). ViG makes use of the DAWNING PC-cluster. The cluster has 9 rendering nodes, and each node has a dual-head FX5200 display card connected to a projector. The projectors are deployed to generate stereoscopic 3 x 3 images on the large display wall. ViG will provide input to the development of this service and will develop a fully Grid-enabled extension of standard rendering APIs to allow scientists to run Grid applications through the ViG user interface.

6 Conclusions

The Grid project for engineering computation (SAGE) supplies the users with a visual application development and deployment environment for engineering computation. The whole is characterized by grid service. Whilst, definition of grid task & resource discovery are visually conducted, QoS driven user-centric scheduling, are designed and implemented. Result processing can be visualized in the aid of a PC-cluster and a stereopticon. In practice, the convenience and QoS are significantly improved, however, there exist a great shortage in the whole system, which are yet the important aspects of our future work.

References

1. I. Foster, and C. Kesselman (eds.), The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann, 1999.
2. J. Nick, I. Foster, et al, The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Open Grid Service Infrastructure, Global Grid Forum, 2002.
3. CQ Huang, GH Song et al, An Authorization Architecture Oriented to Engineering and Scientific Computation in Grid Environments, ACSAC 2004, LNCS 3189, pp. 461–472, 2004.
4. CQ Huang, DR Chen et al, Performance-Driven Task and Data Co-scheduling Algorithms for Data-Intensive Applications, APWeb 2004, LNCS 3007, pp. 331–340, 2004.