A Workflow-based Grid Portal for Problem Solving Environment *

Yong-Won Kwon, So-Hyun Ryu, Jin-Sung Park, and Chang-Sung Jeong

Department of Electronics Engineering Graduate School, Korea University
5-ka, Anam-dong, Sungbuk-ku, Seoul, Korea
{luco|messias|honong13}@snoopy.korea.ac.kr, csjeong@charlie.korea.ac.kr
Tel: 82-2-3290-3229, Fax: 82-2-926-7621

Abstract. In this paper, we present a Workflow-based grId portal for problem Solving Environment(WISE) which has been developed by integrating workflow, Grid and web technology to provide an enhanced powerful approach for problem solving environment. Workflow technology supports coordinated execution of multiple application tasks on Grid resources by enabling users to describe a workflow by composing many existing applications and new functions, and provides an easy powerful tool to create new Grid applications. We propose new Grid portal to allow us to use Grid resources with improved workflow patterns to represent various parallelisms inherent in parallel and distributed Grid applications and present Grid Workflow Description Language(GWDL) to specify our new workflow patterns. Also, We shall show that the MVC(Model View Control) design pattern and multi-layer architecture provides modularity and extensibility to WISE by separating the application engine control and presentation from the application logic for Grid services, and the Grid portal service from Grid service interface.

1 Introduction

For the success of Grid computing, an easy powerful PSE, which provides computing resources and high quality apparatus to solve complex problems of science and engineering technology, are needed. In internet and distributed computing, there are useful technologies for PSE, which have evolved in parallel. Web technology has emerged with revolutionary effects on how we access and process information. Grid computing enables us to use a large or nationwide network of resource as a single unified computing resource[1]. So, clear steps must be taken to integrate Grid and Web technologies to develop a enhanced powerful tool for PSE. Workflow technology is very useful because it enables us to describe a process of work by composing of multiple application tasks on multiple distributed Grid resources. It allows users to easily develop new applications by

* This work has been supported by a Korea University Grant, KIPA-Information Technology Research Center, University research program by Ministry of Information & Communication, and Brain Korea 21 projects in 2004.

composing services and expressing their interaction. In [11, 12, 15, 13, 16, 14, 17–19], several researches about previous workflow management systems for Grid computing were published, and have been studied. However, their workflow models have too simple workflow patterns like sequence, parallel constructs like AND-split/merge, conditional constructs like XOR-split, and iteration constructs to implement parallel and distributed Grid applications efficiently.

In this paper, we present a Workflow-based grId portal for problem Solving Environment(WISE) which has been developed by integrating workflow, Grid and web technology to provide an enhanced powerful approach for problem solving environment. Workflow technology supports coordinated execution of multiple application tasks on Grid resources by enabling users to describe a workflow by composing many existing applications or new functions, and provides an easy powerful tool to create new grid applications. we show our advanced workflow pattern and description language. Our new Grid portal allows us to use Grid resources with improved workflow patterns to represent various parallelisms inherent in parallel and distributed Grid applications. Also, we are concerned about the design and implementation of Grid portal architecture enhanced with softwares to allows users to transparently access remote heterogeneous resources through Grid services. We shall show that the MVC(Model View Control) design pattern and multi-layer architecture provides modularity and extensibility by separating the application engine control and presentation from the application logic, and the Grid portal service from Grid service interface.

The outline of our paper is as follows: In section 2, we describe the basic concepts of Grid and workflow technology, together with their related works. In section 3, we present our new workflow pattern and workflow description language. In section 4, we illustrate the architecture of WISE, and describe the detailed services. In section 5, we explain the implementation of WISE. In section 6, we give conclusion.

2 Related Work

2.1 Grid User, PSE and Grid Portal

A Grid user does not want to be bothered with details of its underlying infrastructure but is really only interested in execution of application and acquisition of correct results in a timely fashion. Therefore, a Grid environment should provide access to the available resources in a seamless manner such that the differences between platforms, network protocols, and administrative boundaries become completely transparent, thus providing one virtual homogeneous environment. Grid requires several design features: a wide range of services on heterogeneous systems, information-rich environment on dynamic Grid, single sign-on, and use of standards and the existing applications. Globus toolkit establishes a software framework for common services of Grid infrastructure by providing a meta computer toolkit such as Meta Directory Service, Globus Security Infrastructure, and Resource Allocation Manager[2, 3]. However, it is responsibility of application users to devise methods and approaches for utilizing Grid services.

PSE is a useful tool for solving problems from a specific domain. Traditionally, it was developed as client-side tools. Recently, a web-based Grid portals have been developed to launch and manage jobs on the Grid, via Web, and allow users to program and execute distributed Grid applications by a conventional Web browser. Webflow is a pioneering computing web portal work, where http server is used as computing server proxy using CGI technology[4]. GridPort[5] allows developers to connect Web-based interfaces with the computational Grid behind the scenes through Globus Toolkit[2] and Web technologies such as CGI and Perl. Hotpage user portal is designed to be a single point-of-access to all Grid resources with informational and interactive services by using GridPort. Astrophysics Simulation Collaboratory (ASC) portal is designed for the study of physically complex astrophysical phenomena[6]. It use Globus and Cactus as a core computational tool.

2.2 Workflow Patterns

In Grid computing, workflow is a process that consists of activities and interactions between them. An activity is a basic unit of work: a grid service or an application executed on Grid. In [11–13], the existing workflow patterns are introduced for describing the only control flow. Most of them are basic patterns such as sequence, simple parallel constructs like AND-Split/Join, conditional constructs like XOR-Split/Join, OR-Split/Join, iteration construct. Other patterns such as N out of M join, deferred choice, and arbitrary cycle are presented in [11, 15]. These are insufficient to express parallel applications. Triana [13, 14] presents link elements for data flow and simple control flow patterns such as a pair of AND-Split and AND-Join, a pair of IF-Split and If-Join, and Count Loop and While Loop. The Grid Services Flow Language(GSFL) [16] is an XML based language that allows the specification of workflow descriptions in the OGSA framework. It also use simple link elements. GridAnt [18] is a client side workflow tool based on java Apache Ant. It describes parallelism by specifying dependencies between tasks. The myGrid workflow [19] provides a graphical tool and workflow enactor, but in terms of the parallel control flow, they support only simple parallelism like the above workflow models and languages.

3 Grid Workflow Description

3.1 New Advanced Workflow Patterns

We need new advanced workflow patterns for Grid applications to describe various parallelism such as pipeline, data parallelism, and many synchronizing constructs and to prevent incomprehensible workflow description. Complex workflow can be made by simple link patterns, but it is difficult. Moreover, any control flow produced by composing sequence and arbitrary cycle may generate ambiguity which is a state to be too complex and difficult to comprehend correct meaning. Therefore, to describe a precise workflow easily and fast, the structured patterns with clear context is more efficient than the non-structured ones

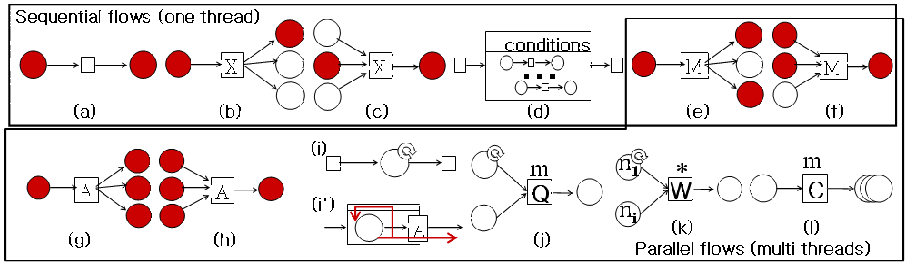

Fig. 1. Advanced basic control patterns: (a) Sequence (b) XOR-Split (c) XOR-Join (d) Loop (e) Multi-Split (f) Multi-Join (g) AND-Split (h) AND-Join (i) AND-Loop symbol (i') AND-Loop description (j) Queue (k) Wait (l) Node copy

made by any compositions of sequences and arbitrary cycles. The details of our workflow model are published in [20]

In figure 1 we show our basic patterns with three groups. In sequential flow, there are four types: sequence for sequential control flow, XOR-Split for conditional execution, XOR-Join for conditional selection among many executed activities, and Loop for repetition. In mixed flow, there are two patterns: Multi-Split for multiple conditional execution, and Multi-Join for multiple conditional selection. Parallel flow includes AND-Split for parallel execution, AND-Join for blocked synchronization, AND-Loop, Queue, Wait, and Node copy. Whenever an iteration of AND-Loop is complete, two control flows occur to two directions like figure 1 (i'): the one for repetition and the other for next sequential activities. The circular-arrow in figure 1 (i) is the graphic notation of AND-Loop. AND-Loop can send many flows to a next node N continuously. If N is bottleneck, activities that send flows to N may stop or be processed slowly. A pattern is needed to prevent this situation. In queue pattern, all input control flows are stored in queue and transferred whenever the next node is idle. In node copy pattern, a node is copied up to the limited number of times. An idle node is selected and executed in parallel whenever a input control flow occurs. This pattern can increases computing power and may solve the above bottleneck problem. In wait pattern, wait node blocks until some or all input control flows are received. For example, in figure 1 (k) wait node blocks until all control flows generated by AND-Loop n_1 and node n_i are received. The symbol '*' means all.

3.2 Grid Workflow Description Language

We define the Grid Workflow Description Language (GWDL) which is an XML based language that specifies our workflow model using XML Schemas. The GWDL architecture consists of dataDefine, resourceDefine, activityDefine, and flowDefine elements. DataDefine lists user data types that are used to describe input/ouput data of an activity node. ResourceDefine describes host address or resource specification for executing an activity. ActivityDefine lists activities which are executions of executable files or requests to running services on

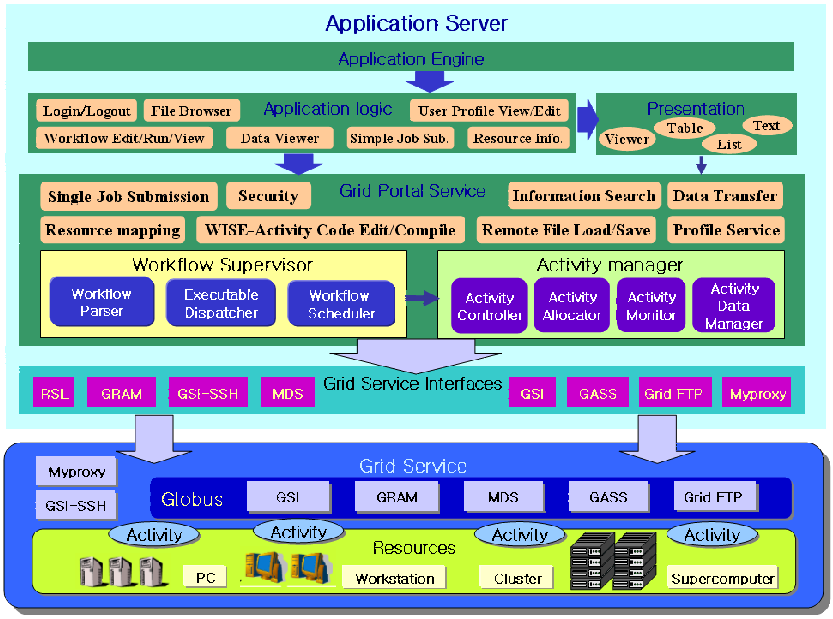

Fig. 2. The Overall Architecture of WISE

Grid resources. It describes function, names and types of input/output data. FlowDefine defines activity node elements which have an activity and a resource specification, and both of control and data flows which describe the interaction of activity nodes. It has basic control flow elements for representing our new workflow patterns in section 3.1. We also define three elements for control flow: <sequence>, <parallel>, and <loop>. Activity nodes in sequence element and parallel element are connected sequentially and executed concurrently respectively. Loop element iterates the sub-workflow. It has a 'pipeline' attribute which indicates AND-Loop. Also, in flowDefine elements, there are data elements for describing data flow, which has source, destination, and variable name.

4 WISE Architecture

4.1 Overall Architecture

Our Grid portal has a 3-tier architecture which consists of clients, web application server, and a network of computing resources like figure 2. Web application server in the middle tier is augmented with Grid-enabling software to provide accesses to Grid services and resources. It is designed as a multilayered structure which exploits MVC(Model-View-Controller) design pattern and CoG(Commodity Grid) technology to construct a highly modular and flexible software environment. Application engine in web server controls and organizes

the overall portal by executing the proper application logic component, according to the client request, which in turn carry out Grid services through Grid service interface, and then activating the presentation module which generates application specific display to be transmitted to the client's browser. MVC design pattern provides modularity and extensibility of Grid portal. Grid service interface components are implemented by using CoG technology which maps Grid functionality into a commodity tool.

4.2 WISE User Interfaces

Our Grid portal provides the following functions which allow users to easily access Grid resources and make new Grid applications efficiently by using our workflow-based Grid portal.

User Authentication and Profile: This is basic components for Grid portal. A user is authenticated only once and provided all functions of our portal. WISE has user profile function which manages his/her information such as Grid certificate creation, update and management, list of available Grid resources, management of environment variables on many distributed host, request of resource authorization to its manager and email address.

Remote File Browser: This is file management functions such as file directory browsing of remote hosts, creation, delete, editing, upload from local file system, download to the local, file transfer between two remote hosts by GridFTP, and edit/save of remote file. By this basic and powerful function for remote file management, User can modify and move remote files, configuration/parameter files of application, and so on.

Graphic Workflow Editor: This editor is for creating a GWDL file. User can describe a workflow by graphical tool or text-based XML editor. First activities and its input/output data are defined. Second, User describes the interaction between them by making a workflow with them. User can specify the host to execute an activity, or yield the choice of host to workflow execution function.

Our workflow system provides not only an executable file execution but also WISE-activity which interacts workflow supervisor tightly and can send data to another WISE-activity with socket. Workflow supervisor monitors the state of WISE-activity and controls its execution and data transfer through socket. User can program and compile WISE-activity codes with this workflow editor after define its name and input/output data in GWDL. User will use WISE-activities to implement new Grid applications or monitoring programs to report input, output, and state of an executed legacy application efficiently.

Workflow Execution: After edit a GWDL file, we run it with this function. This shows the state of executed workflow graphically. The execution of an activity, state of input/output data transfer, values of some variables in workflow, and standard output/error of an running activity are displayed. If user specifies the requirement for the host to run an activity, workflow supervisor will selects an idle host by using resource mapping portal service.

Data Viewer: User can select a data file through file browser and see it with a data viewer which is associated with the data type. User can add new data viewer that is implemented by java or Microsoft COM to our Grid portal.

Resource Information: Resource Information function enables us to find data and status about Grid resources such as CPU, memory, file system, network, os, and software through MDS.

Simple Job Submission: User can send a request to run single executable file to a remote host and see the text type standard output of it.

4.3 Grid Portal Service

The Grid portal services forms the foundation of WISE, and are used directly by the application logic and presentation activated by the application engine. Each application logic and presentation has one-to-one correspondence with a user request from GUI on client side for solving a specific application problem on remote resources. In this subsection, we describe how various Grid portal services complete their missions by executing Grid services through the Grid service interface components.

Single Job Submission: Job submission service can be executed in two modes. In user mode, user can prepare a job specification using RSL component, and job submission service executes the jobs in the remote resources as specified in RSL by using GRAM component. In automatic mode, job submission service finds a computing resources from resource mapping service.

Information Service: Information service provides static and dynamic information on resources by querying and formatting results obtained from the MDS component in Grid service interface layer. It supports querying the MDS for hardware information such as CPU type, number of CPUs, CPU load and queue information that can be used by the user to make more efficient job scheduling decisions.

Data Transfer Service: Data transfer provides file transfer capabilities between client and target machine, or between third-party resources as well as file browsing to facilitate the transfer of programs or data files for remote execution and data retrieval by using GridFTP in Grid service interface layer.

Security: Security service simplifies the authentication task by exploiting GSI of Globus which provides a secure method of accessing remote resources by enabling a secure, single sign-on capability, while preserving site control over access control policies and local security infrastructure.

User Profile Service: User profile allows user to keep track of past jobs submitted and results obtained by maintaining the user history, and in addition enables the customization of a particular set of resources by users to provide additional application specific information for a particular class of users. Also sending an email, and management of user certificate are provided.

Workflow Supervisor: Workflow supervisor consists of three parts: parser, scheduler, and dispatcher. First, a GWDL file is inserted into workflow parser.

The parser parses it and sends the activity node information into scheduler. Second, despatcher acquires data about executable files of activities and distributes the executables to Grid resources. Third, Scheduler runs the input GWDL file with the activity node information. It sends requests of activity execution to activity manager, monitors the information of activities, and controls activities. If no host name for an activity node, it get a host from resource mapping service.

Activity Manager: Activity manager controls an execution of activities with Globus GRAM service, acquires the state of activities, and manages input/output data of activities with GridFTP service. Also it controls and monitors WISE-activity directly with socket for state acquisition and data transfer.

WISE-Activity Editing/Complile: First, user describe the name, and input/output data of WISE-activity in a GWDL file, and then workflow editor generates the template codes for WISE-activity from content of the GWDL. Second, User can program logic of WISE-activity and compile the codes. This service provides two functions: making template codes and compiling user codes.

Resource Mapping: This service finds computing resources to which a user can submit jobs by using MDS component, and return idle resources according to the user requirements such as memory size, CPU performance, and etc.

4.4 Grid Service Interface

Grid service interface defines classes that provide access to basic Grid services and enhanced services suitable for PSE, and encapsulates the functionality for Grid services offered in Globus toolkit by using CoG.

RSL component provides methods for specifying resource requirements in Resource Specification Language(RSL) expressions. GRAM component provides methods which allows users to submit jobs on remote machines as specified in RSL, monitor job status and cancel jobs. GridFTP[8] component provides file transfer on Grid. GASS component supports the access of remote file system on Grid. MDS component allows users to easily access MDS service in Globus. MyProxy is a secure online certificate repository for allowing users to store certificates and then to retrieve them at a later time. User can acquire Globus proxy with ID and password from MyProxy server.

5 Implementation

WISE deploys Apache web server, a free, open source web server that support SSL. WISE was developed under the open source Tomcat java servlet container which is freely and widely available and implemented by usig java language, JSP, Java Beans, and java servlet. Grid service interfaces are implemented in Java packages that provide the interface to the low level Grid services by using Java CoG kit[10]. WISE provides users with various interactive operations : login/out, job submission, MDS query, GridFTP, file browsing, user profile, workflow editing and running. They support transparent access to heterogeneous resources by

providing a unified and consistent window to Grid by allowing users to allocate the target resources needed to solve a specific problem, edit and transfer programs or data files to the target machines for remote execution and data retrieval, select and submit the application jobs automatically, query the information on the dynamic state of hardware and software on Grid for the more efficient job scheduling decisions as well as private user profile. In addition, WISE also provides a easy-to-use user interface for using workflow-based parallel programming environment on Grid, by supporting graphical workflow editor, resource finding, authentication, execution, monitoring, and steering. Therefore, Grid portal provides a transparent mapping between user interface and remote resources, hiding the complexity of the heterogeneous backend underlying systems.

6 Conclusion

Commodity distributed computing technology in Web enables the rapid construction of sophisticated client-server applications, while Grid technology provides advanced network services for large-scale, wide area, multi-institutional environments and applications that require the coordinated use of multiple resource. In this paper, we present a web-based Grid Portal which bridge these two worlds on internet in order to enable the development of advanced applications that can benefit from both Grid services and commodity web environments. Also We provide new workflow patterns and GWDL which can overcome the limitations of the previous approaches by providing several powerful workflow patterns used efficiently to represent parallelisms inherent in parallel and distributed applications. To describe a workflow of grid application without ambiguity, we have proposed formally new advanced basic patterns such as And-Loop, queue, wait, node copy and etc by classifying them into three categories; sequential, parallel, and mixed flow. Our workflow-based Grid portal have been designed to provide a powerful problem solving environment by supporting a unified and consistent window to Grid which enables a substantial increases in user ability to solve problems that depend on use of large-scale heterogeneous resources. It provides users with a uniform and easy to use GUI for various interactive operations for PSE such as login/out, job submission, information search, file browsing, file transfer, and user profile, and especially supports interfaces for using workflow-based parallel programming environment on Grid, by supporting graphical workflow editor, resource finding, authentication, execution, monitoring, and steering. We have proposed a multi-layer architecture which can provide modularity and extensibility by each layer interacting with each other using the uniform interfaces. Also, we have shown that MVC design pattern provides flexibility and modularity by separating the application engine control and presentation from the application logic for Grid services, and that commodity-to-Grid technology for Grid service interface supports various platforms and environments by mapping Grid functionality into a commodity distributed computing components. As a future work, we are extending our Grid portal which enables the automatic conversion

of a given problem into optimized parallel programming models by supporting more specific coarse-grained parallel programming models in our system.

References

1. I. Foster, C. Kesselman, S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," International J. Supercomputer Applications, 15(3), 2001.
2. Globus Toolkit, http://www.globus.org
3. I. Foster, and C. Kesselman, "The Globus Project: A Status Report," Heterogeneous Computing Workshop, pp. 4-18, 1998.
4. W. F. Erol Akarsn, G. C. Fox and T. H. Haupt, "Webflow High-level Programming Environment and Visual Authoring Toolkit for High Performance distributed Computing, In Proceedings of Supercomputing '98, 1998.
5. S. M. Mary thomas, and J. Boisseau, "Development of Web Toolkits for computational Science Portals: The NPACI Hot page," In Proceedings of HPDC 9, pp. 308-309, Aug. 2000.
6. Astrophysics Simulation collaboratory: ASC Grid Portal, http://www. ascportal.org.
7. K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman, "Grid Information Services for Distributed Resource Sharing," Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press, August 2001.
8. W. Allcock, J. Bester, J. Bresnahan, A. Chervenak, L. Liming, S. Meder, S. Tuecke, "GridFTP Protocol Specification," GGF GridFTP Working Group Document, September 2002.
9. K. Czajkowski, I. Foster, and C. Kesselman, "Resource Co-Allocation in Computational Grids," Proceedings of the Eighth IEEE International Symposium on High Performance Distributed Computing (HPDC-8), pp. 219-228, 1999.
10. G. V. Laszewski, I. foster, J. Gawor and P. Lane,"A Java Commodity Grid Kit." concurrency and computation: Practice and Experience, pp. 645-662.
11. B. Kiepuszewski, Expressiveness and suitability of languages for control flow modelling in workflows, http://tmitwww.tm.tue.nl/research/patterns/download/phd_bartek.pdf
12. W.M.P. van der Aalst, A.H.M, Hofstede, B. Kiepusziewski, A.P. Barros. (2003). Workflow Patterns, Distributed and Parallel Databases, Jule 2003, pp. 5-51
13. Junwei C., Stephen A. J., Subhash S., and Grahan R. N., GridFlow: Workflow Management for Grid Computing, Proc. 3rd IEEE/ACM Int. symp. on cluster Computing and the Grid, 2003
14. Triana Workflow, http://www.gridlab.org/WorkPackages/wp-3/D3.3.pdf
15. Dan C. M., A Grid Workflow Management Architecture, http://www.cs.ucf.edu/ dcm/GWfA.pdf
16. Sriram K., Patrick W., Gregor von L., GSFL:A Workflow Framework for Grid Services http://www-unix.globus.org/cog/projects/workflow/gsfl-paper.pdf
17. Hugh P. B., Grid Workflow http://vir.sandia.gov/ hpbiven/ggf/draft-bivens-grid-workflow.pdf
18. GridAnt, 2003, http://www-unix.globus.org/cog/projects/gridant/
19. myGrid workflow, 2003, http://www.mygrid.org.uk/myGrid/
20. Kwon Y. W., Ryu S. H., Jeong C. S. and Park H. W., XML-Based Workflow Description Language for Grid Applications, LNCS 3043, ICCSA 2004. pp. 319-327