# Grid Workflow Management System

J. Song[1], J. Singh[2], C.K. Koh[1], Y.S. Ong[2] and C.W. See[1]

[1]Asia Asia Pacific Science and Technology Center
Sun Microsystems Inc.
[2]Nanyang Technological University, School of
Computer Engineering,
50 Nanyang Avenue, Singapore 639798

Jie.Song@sun.com
Jasleensingh@pmail.ntu.edu.sg
Chee-Kian.Koh@sun.com
ASYSOng@ntu.edu.sg
Simon.See@sun.com

## Abstract

A Grid Workflow is critical to grid computing for its ability of creating complex grid computation by connecting different grid jobs logically. User can easily define and reuse the workflow for their applications that are loosely coupled. In this paper, we proposed a Grid Workflow Management System (GWMS) to provide the interface of processing workflow with grid schedulers, e.g., Sun Grid Engine clusters. The client can be a portal, an application or command line. We present the architecture of grid workflow engine, the design of the extended API for DRMAA to compile and execute workflow in SGE, and also show the flexibility, interoperability and extensibility of the GWMS.

**Keyword**: Workflow, Grid Computing, DRMAA

## I. Introduction

Computational grids have become an important emerging platform for high-performance and resource consuming scientific computing. As technology improves, scientific research has also evolved, requiring more sophisticated and complex computations. Therefore workflow is critical to grid computing for its ability of connecting different grid jobs together to generate a complex computation.

Although there are many workflow tools [2, 3], not all of them support grid directly. Of these, GridAnt [6] and Geodise project has developed workflow tools that support Globus, and TENT [7] software tool defines workflow using a component architecture and components can be access using Sun Grid Engine [4] or LSF. Gagman [8] is the meta-scheduler over Condor. It's obviously that none of them can interoperate between the heterogeneous grid

environments. Tirana [9] supports the service-oriented and grid –oriented bindings for distributed workflow over grid, but they use Grid Lab GAT and P2PS services to implement and does not support DRMAA.

In this paper, our focus is to propose a grid workflow management system especially the design and implementation of workflow engine and the DRMAA [5] based API for workflow management framework. The grid workflow editor is also improved from the earlier implementation of GriDE [1]. These three components consists the flexible and extensible grid workflow management system.

The rest of paper is organized as follows: Section 2 describes the architecture of the proposed grid workflow management system and the details design for each components. Section 3 summarizes the conclusions.

## II. Architecture of Grid Workflow Management System

As shown in the Figure 1, the grid workflow management system (GWMS) has three major components:

- Workflow Editor
- Workflow Engine
- Workflow Management Framework (WMF)

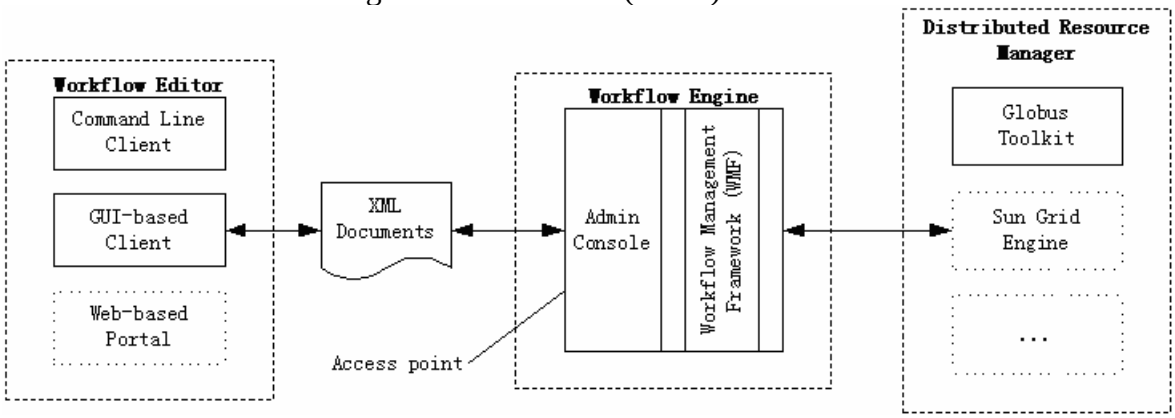



Figure 1 Architecture of Grid Workflow Management System

Workflow editor is the front-end of the system. The interface of the grid workflow editor can be a web-based portal, a GUI or even a command line interpreter. It allows the users to submit a grid workflow by providing information like job parameters, resource requirements in a grid workflow description. It will then compile the representation and generate an XML workflow description document to be submitted over the access point.

The access point can either be a web-service (for transmission over HTTP), or a direct stream transfer using sockets to the workflow engine. The Admin console is the interface to the workflow engine, which is the core component of the grid workflow management system. The engine is responsible for controlling the submission of the grid jobs according to the sequence that is described by the grid workflow description language. As shown in Figure 1, the engine uses this description to execute the job via the workflow management framework (WMF). The engine is also responsible for maintaining context information for each user, listening for requests from the user, notifying the success or failure information of the workflow and directing the output/error to the location specified by the user.

The workflow management framework (WMF) is a set of API that manages the execution of grid workflow tasks at the back-end distributed resource manger (DRM) using distributed resource management application API (DRMAA). Therefore, the WMF is extensible to support for different DRM systems, e.g., Sun Grid Engine or Globus Toolkit.

The design of the Workflow Management System has following features:

- Flexibility: The component pattern is applied in the design to minimize the coupling between major components. This will facilitate changing of one component without affecting the others.
- Interoperability: The Design of core components is based on open standards and specifications. For example, DRMAA is used, which is expected to be most adopted API for high-level interface for submission and monitoring of jobs to various DRMs. XML is used for data exchange over the grid.
- Extensibility: The system is extensible by a layered structure where each layer uses the abstraction the layer below it provides. Each layer adds more functionality to the system.

In the following sections, we will describe the details for each component.

### *A. Grid Workflow Editor*

Grid Workflow Editor is improved from the earlier implementation for GriDE. It provides functions to allow user easily create and modify their grid workflows as a graph. The node represents the computation activities of a particular grid job. The directed lines represent either control flow or data flow. It supports four types of workflows: sequence, parallel, loop and conditional. Each of them can be combined in different ways to build a complex grid workflow. The patterns are illustrated in Figure 2.

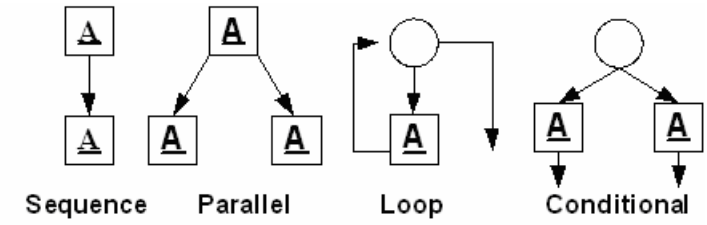



Figure 2 Grid workflow patterns

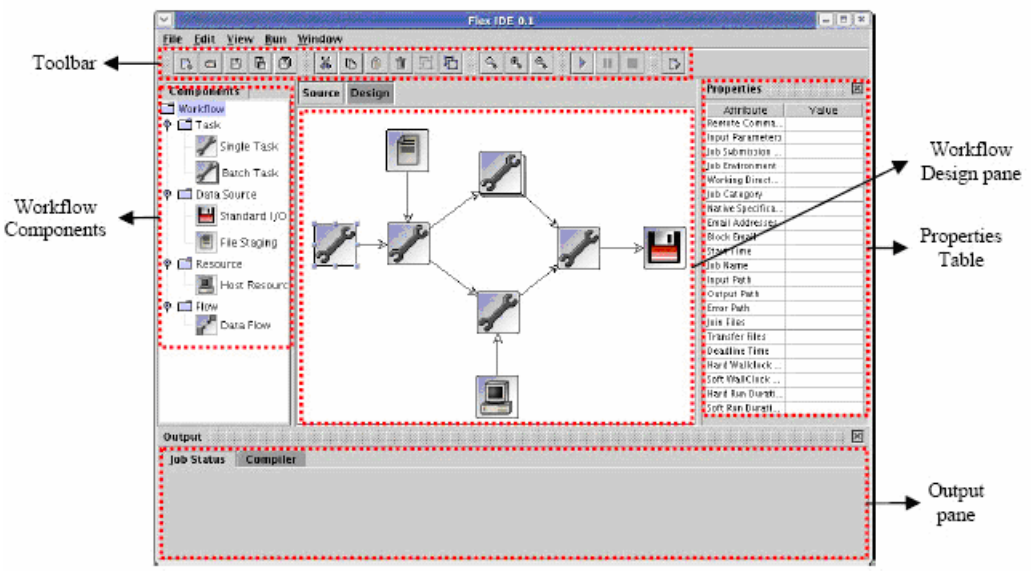



Figure 3 Workflow Editor - Design Mode

The grid workflow editor has two different edit modes: Design Mode and Source Mode. In the design mode (Figure 3), the user can express workflow of grid job execution graphically, change the attributes of individual jobs, submit it to workflow engine, view and control the execution, etc. The common edit operations such as cut, copy and paste, etc are provided. It also support usual file operations as create, open and save a file. In the source mode Figure 4), the user can compile the graphical workflow into XML file and manually modify the portion as required. The workflow editor will automatically synchronize between the design and source mode.

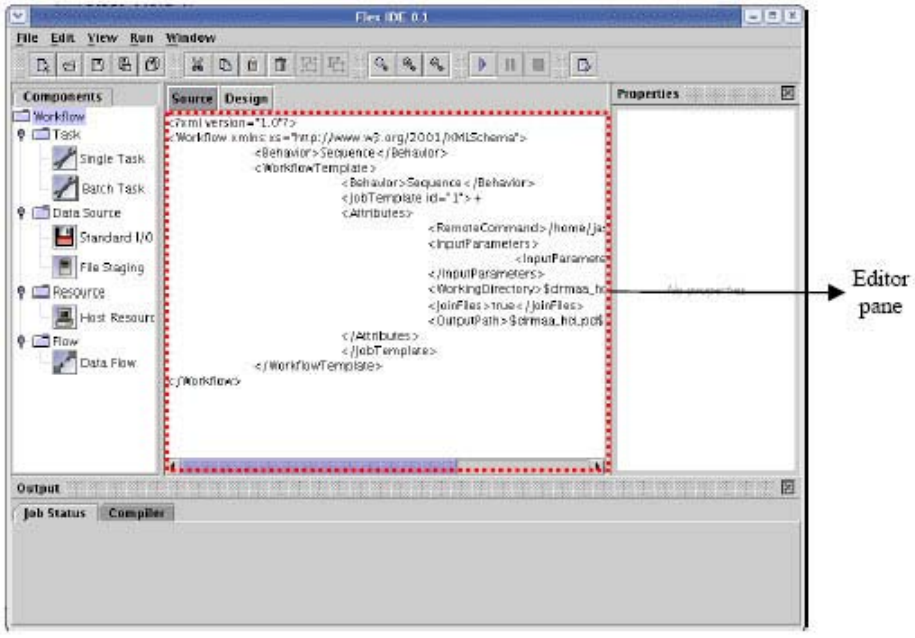



Figure 4 Workflow Editor - Source Mode

### *B. Grid Workflow Engine*

The Grid Workflow Engine is the execution module of the grid workflow management system. The objective of this component is to design an engine, which could receives and process workflow requests from clients, which are then executed

over execution grid nodes which may be controlled by a scheduler like the Sun Grid Engine or GRAM of Globus Toolkit. The grid workflow engine is essentially a daemon that accepts incoming connections from clients. Each new connection spawns off a new thread – the Job Manager, which then handles all communication with that client. Figure 5 shows the sequence diagram for the grid job submission and execution of a grid workflow on the engine.

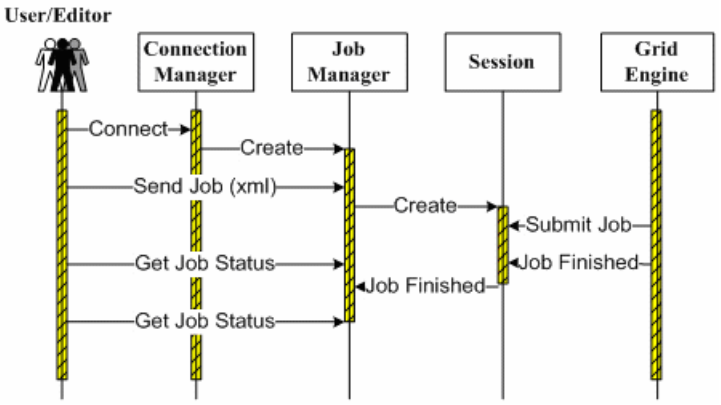



Figure 5 Sequence diagram for grid job submission and execution

The engine also has a front-end in the form of a graphical interface, called the Admin console as shown in Figure 6. It provides the functions to monitor the grid job list, change the priorities of pending jobs, view and edit the grid job request files in XML format, to run, pause and terminate the job execution, and also configure the execution policies.

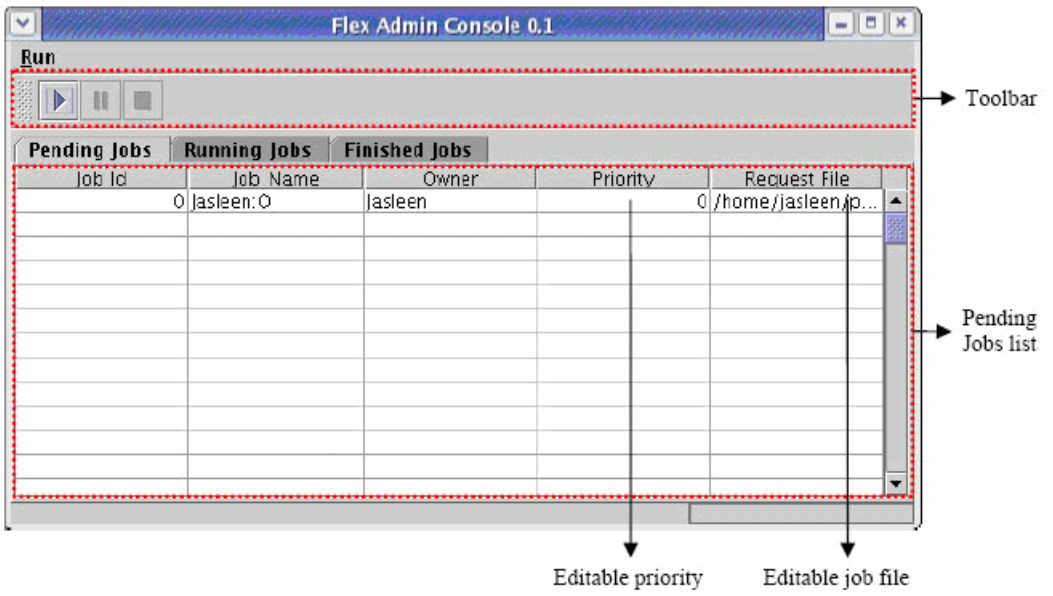



Figure 6 Admin Console for grid workflow Engine

### *C. Grid Workflow Management Framework*

The Workflow Management Framework (WMF) is a set of API that provides common and useful functions for workflow management. With this objective, a subset of the API was designed and implemented. These APIs provide functions that allow a user to

- Initiate a session
- Exit a session
- Create Workflow Templates: include Unit Workflow Templates and Composite Workflow Templates
- Execute Workflow Templates

Figure 7 shows the design of WMF. It is an abstract factory to isolate clients from implementation classes. Thus it enhances flexibility of multiple bindings to support a heterogeneous grid environment.

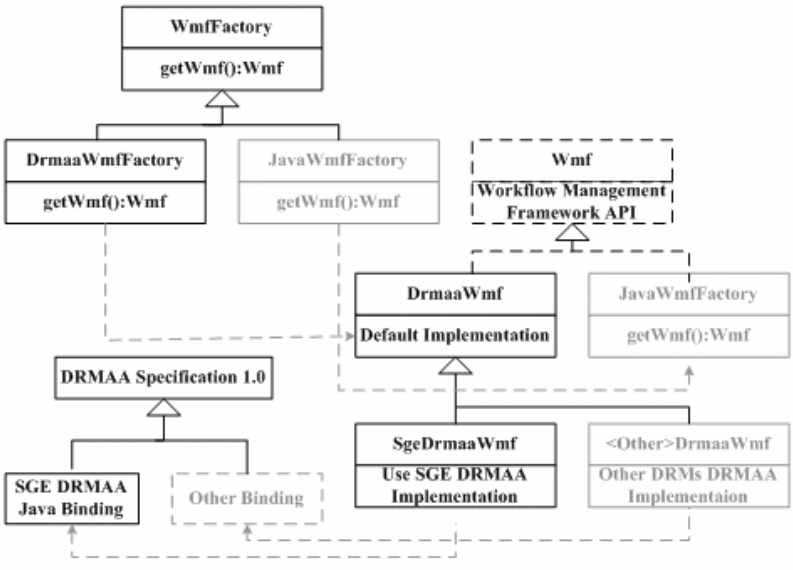



Figure 7 Grid Workflow Management Framework Design

These WMF APIs have been modeled on the DRMAA analogous to the *JobTemplate* class in DRMAA, which allows one to specify the attributes of a single job. The major classes in WMF are described below:

- *WorkflowTemplate* class: It allows one to specify the attributes of a single grid workflow element e.g. a sequence of single jobs (*JobTemplates*). It uses the Composite pattern to compose larger workflow elements from smaller ones.
- *SequenceUnitWorkflowTemplate* class: It contains a single job
- *ParallelUnitWorkflowTemplate* class: It containing the two parallel jobs
- *DrmaaWmfSession* class: It is analogous to the DRMAA *Session* class and allows a user to execute a *WorkflowTemplate* using its runWorkflow() function.

Other functions like wait() and synchronize() can block program flow until the *WorkflowTemplate* does not complete execution. These functions are analogous to the runJob(), wait() and synchronize() functions in DRMAA.

## III. Conclusions

In this paper, we design and implemented a Grid Workflow Management System (GWMS) to provide the interface of processing workflow with grid schedulers, e.g., Sun Grid Engine cluster. Three major components are described:

1) Grid workflow editor, a GUI-based client for describing execution grid workflow;
2) Grid workflow engine, an execution and job-control system
3) Grid Workflow Management Framework, a set of APIs for workflow elements.

Based on the above components, we provide a java-based flexible, interoperable, and extensible solution for expressing and executing a grid workflow system.

## References

[1] Simon See, J. Song, L. Peng, A. Stoelwinder, H.K. Neo, "GriDE: A Grid-Enabled Development Environment", in *Proceedings of the Second GCC, Shanghai*, China, 7-12 December 2003.

[2] Ewa Deelman, James Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Sonal Patil, Mei-Hui Su, Karan Vahi, Miron Livny, **"**Pegasus : Mapping Scientific Workflows onto the Grid**",** *Across Grids Conference*, 2004

[3] Grid Physics Network, http://www.griphyn.org

[4] Sun Grid Engine, http://gridengine.sunsource.net

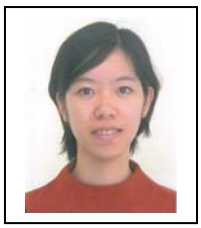
[5] Hrabri Rajic et al. , "Distributed Resource Management Application API Specification 1.0", GlobalGrid Forum Recommendation, 2004

[6] GridAnt, http://wwwunix.globus.org/cog/projects/gridant.

[7] TENT, http://www.sistec.dlr.de/tent/index.shtml.

[8] DAGMan, Directed Acyclic Graph Manager, http://www.cs.wisc.edu/condor/dagman/

[9] Hrabri Rajic et al., http://www.trianacode.org/Triana



Dr. Jie Song received the Bachelor and Master degrees in Computer Engineering from Xi'an Jiaotong University, P.R. China in 1995 and 1998 respectively. In 2004, she obtained the Ph.D degree from school of computer engineering, Nanyang Technological University, Singapore. She joined the Asia Pacific - Science & Technology Center since January 2003. Her research interest is in the area of Grid Computing, Internet pricing, Quality of Service and network communication.

Jasleen Singh joined Asia Pacific Science and Technology Center as an intern from July to December 2004. He is currently a senior at the School of Computer Engineering at Nanyang Technological University, Singapore. He completed his schooling in India and came to Singapore for his bachelor's degree on a SIA/NOL scholarship. His research interests are in Computer Science, especially Artificial Intelligence and Computer Networks, and loves

Mr. Melvin Koh is currently a Research Engineer in Asia Pacific Science and Technology Centre. He received his Degree in Computer Science from National University of Singapore in 2002. His research interests include combinatorial and optimization, machine learning, agent-based systems, and grid computing.

Dr. Yew Soon Ong received his Bachelors and Masters degrees in EEE from Nanyang Technology University (NTU) in 1998 and 1999, respectively. He then joined the Computational Engineering and Design Center at the University of Southampton, where he received his Ph.D. degree in 2003. He is currently an Assistant Professor with the School of Computer Engineering, NTU. His research interests lie in evolutionary computation spanning: design optimization, surrogate-assisted evolutionary algorithms, memetic algorithms, evolutionary computation in dynamic and uncertain environments, response surface methods for data modeling, and grid-based computing.

A/Prof Simon See is currently the High Performance Computing Technology Director for Sun Microsystems Inc, Asia and also an Adjunct A/Prof. in Nanyang Technological University and an adjunct research fellow in the National University of Singapore. He is the director for the Sun Asia Pacific Science and Technology Center. His research interest is in the area of High Performance Computing, computational science, Applied Mathematics and simulation methodology. He has published over 30 papers in these areas and has won various awards. Dr. See graduated from University of Salford (UK) with a Ph.D. in electrical engineering and numerical analysis in 1993. Prior to joining Sun, Dr See worked for SGI, DSO National Lab. of Singapore, IBM and International Simulation Ltd (UK). He is also providing consultancy to a number of national research and supercomputing centers.