E-SCIENCE WORKFLOW ON THE GRID

Yaohang Li
Department of Computer Science
North Carolina A&T State University, Greensboro, NC 27411, USA
yaohang@ncat.edu

Michael Mascagni
Department of Computer Science and School of Computational Science and Information Technology
Florida State University, Tallahassee, FL 32306, USA
mascagni@fsu.edu

ABSTRACT

Grid computing, which can be characterized as large-scale distributed resource sharing and cooperation, has quickly become a mainstream technology in distributed computing. In this paper, we present the idea of applying certain grid workflow management techniques to mediate various services for grid-based e-science processes. The techniques of adaptable workflow services, aggressive sub-workflow scheduling, and application-level checkpointing are used to improve the performance and trustworthiness of e-science workflow on the grid. A protein structure prediction application is presented as an example to show how the embedding of a large-scale e-science application in a grid-based workflow can be used to achieve high performance simulation in this context.

KEYWORDS

Grid Computing, e-science, Workflow, XML.

1. INTRODUCTION

Grid computing is an emerging technique to support on-demand “virtual organizations” for distributed resource sharing and problem solving on a global scale for data-intensive and computation-intensive applications [Goble, C. and Roure, D. D., 2003]. In the grid-computing environment, large-scale computational resources, global-wide networking connectivity, access to high-end scientific instruments, participation of scientists and experts in different areas, and coordination of organizations make the grid a powerful and cost-effective platform to carry out e-science operations. The combination of these operations permits the implementation of scientific tasks, such as analyzing raw data streams, performing large-scale computations, designing cutting-edge technological products, collaborating on interdisciplinary research, and implementing a particular scientific computing process, to name a few. Despite the attractive characteristics of grid computing, to successfully apply grid techniques to e-science, the grid environment presents a number of challenges that must be overcome. These stem from properties of the grid such as dynamism, cross-domain physicality, heterogeneity, lack of intrinsic trustworthiness, reliability, and performance [Foster I. et al., 2001; Foster I. et al., 2003]. Workflow management services on the grid are designed to efficiently manage and organize dynamic grid resources to provide reliable and effective services for e-science applications [Chen Q. et al, 1999; Cao J. et al, 2003]. In this paper, we present some techniques coming from grid workflow management to address certain issues in grid computing to aid in implementing high-performance and trustworthy e-science applications.

The remainder of this paper is organized as follows. In Section 2, we analyze grid-based workflow as from the point of view of e-science applications. In Section 3, we present grid workflow techniques including adaptive workflow services, aggressive scheduling, checkpointing, and validation to achieve high-performance and trustworthy e-science computations. Finally, Section 4 summarizes our work, and discusses related future research directions.

2. E-SCIENCE WORKFLOW ON THE GRID

2.1 Grid-based Workflow

Originally, workflow was an administrative concept from the field of managing business operations,. Workflow refers to a business process that delivers services from one participant agent to another. In 1996, a workflow was described by the Workflow Management Coalition as [Allen, R., 2001]:

> *The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules.*

The emergence of workflow introduced automated processes which enforced data validation and verification within business operations, overcame constraints in time and space, maintained consistency in the business system, and significantly eliminated possible human errors. Workflow is making important contributions to many types of business applications and processes. Nevertheless, the concept of workflow extends beyond its use in conventional business process management, and is now used more broadly in e-science, e-commerce, and other related areas. More recently, workflow techniques have been introduced into the grid computing environment to efficiently and effectively manage different grid services and resources [Fox, G. and Walker, D., 2003; Cao J. et al, 2003].

2.2 Components in a Grid Workflow

Workflows require many capabilities, and as mentioned in their "e-science gap analysis" [Fox, G. and Walker, D., 2003], four key components are highlighted to describe a workflow:

- Composition/Development: To provide an Integrated Development Environment (IDE) to virtually form the graph of a workflow.
- Language and Programs: To describe the workflow using a formal language. Some specialized workflow languages have been developed for grid services or web services, including BPEL4WS, WSFL, GSFL, SWFL, and XPDL.
- Compiler: To translate the above two steps into the executable form.
- Enactment Engines: To support the execution of the workflow in the execution environment.

These four components correspond to related capabilities in a conventional programming environment.

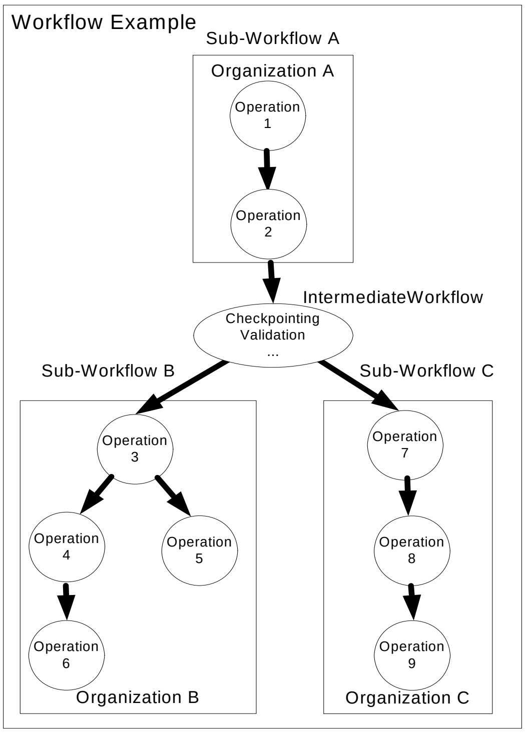
2.3 Describing a Grid Workflow in XML

A scientific computing task on the grid is normally composed of a number of subtasks, and so the corresponding workflow on the grid can as well be decomposed into smaller units. These units can be described as follows:

- Operation: Operations are the smallest elements in a grid workflow. Each operation in a grid workflow corresponds to a computational operation and is usually carried out on an individual grid node.
- Sub-workflow: A sub-workflow is a flow of closely related operations that is to be executed in a predefined order on the grid resources within a virtual organization. Each sub-workflow represents a specific scientific computing subtask in an organization. Sub-workflows may be executed in parallel.
- Intermediate-workflow: An intermediate-workflow mediates sub-workflows running in different organizations. It carries out tasks such as sub-workflows coordination, checkpointing, computation verification, and result validation operations.
- Workflow: A workflow can be represented as a flow of several loosely coupled activities described in a scientific computing process. Each activity consumes various grid resources and can be represented by a sub-workflow.

The grid workflow management service schedules the sub-workflows in a workflow to the appropriate target organizations. Then, operations are executed on the grid resources within the organizations. The intermediate-workflow coordinates the execution of sub-workflows. XML (eXtensible Markup Language) is used to describe a grid workflow. XML is fast becoming the standard for data interchange on the grid due to its well-defined syntax and platform independency. Thus, XML is used as the primary message format for communications within grid services. In fact, an XML document is an information container for reusable and customizable components, which can be used by any receiving services. The grid services in a workflow may use an XML format to explain their “problems,” defining new performatives in terms of existing and/or mutually understood ones. Based on commonly agreed upon tags, services in a workflow may use different style XML DTDs (Document Type Definition) to fit the taste of the units they mediate. Therefore, XML is widely used to address the heterogeneity issue in grid computing and to enable cross-domain cooperation among grid services [Fisher, L. 2002; Bivens, H. P., 2001].

Figure 1 illustrates an example of a grid workflow diagram and the corresponding XML description. The workflow is decomposed into three sub-workflows with each sub-workflow to be scheduled on a grid organization specified by the “organization” tag. The “order” tag indicates the execution order of these sub-workflows, and the “operation” tag carries out the “problem description” of the current operation, including the application-specified actions and requested grid resources.



```
<WorkFlow id = “mainworkflow”>
  <SubWorkFlow id = “subworkflow1”, order = 1>
    <Organization id = “organizationA”> </Organization>
    <DataTransfer> … </DataTransfer>
    <Operation> description of operation 1 and 2
    </Operation>
    <DataTransfer> … </DataTransfer>
  </SubWorkFlow>
  <IntermediateWorkFlow id =”intermediateworkflow1”>
    <DataTransfer> … </DataTransfer>
    <Operation> checkpointing, validation, …
    </Operation>
    <DataTransfer> … </DataTransfer>
  </IntermediateWorkFlow>
  <SubWorkFlow id = “subworkflow2”, order = 2>
    <Organization id = “organizationB”> </Organization>
    <DataTransfer> … </DataTransfer>
    <Operation> description of operation 3, 4, 5, 6
    </Operation>
    <DataTransfer> … </DataTransfer>
  </SubWorkFlow>
  <SubWorkFlow id = “subworkflow3”, order = 2>
    <Organization id = “organizationC”> </Organization>
    <DataTransfer> … </DataTransfer>
    <Operation> description of operation 7,8,9
    </Operation>
    <DataTransfer> … </DataTransfer>
  </SubWorkFlow>
</WorkFlow>
```

Figure 1. Grid Workflow, Sub-workflows, Intermediate-workflow, and Operations

3. TECHNIQUES OF HIGH-PERFORMANCE AND RELIABLE GRID-BASED E-SCIENCE APPLICATIONS

3.1 Workflow Service with Adaptable Behavior

E-science on the grid is a dynamic scientific computing environment. Many e-science applications favor large-scale computational resources and dynamically request available resources to achieve optimal performance. Therefore, grid services for a computational operation in a workflow need to be established dynamically on demand. However, conventional software agents running on the grid, including many web-based service providers, have pre-defined functions but lack the capability of changing role and behavior dynamically. For example, a SETI@home [SETI@home, 2002] agent can only perform SETI computations, while a folding@home [Folding@home, 2000] agent has only protein folding functionality. A folding@home agent is not able to use idle and available resources known to it for SETI@home services. Lacking the ability to adjust their behaviors, conventional agents may be too limited for mediating high performance e-science applications properly. The dynamic nature of e-science requires more "flexible" agents and services in grid organizations to be based on a dynamic ontology. To meet this requirement, grid-based workflow services can be designed to be capable of handling "plug-and-play" e-science computations.

The enactment engine is the key component in a workflow to address the issue of dynamism. The enactment engine has the functionalities of workflow management, which are responsible for binding the workflow expression to the actual grid-service components and setting up the necessary inter-service communication registrations. Based on the "problem description" of the operation in the workflow, it picks up appropriate and available grid services, schedules the operations to the grid resources, executes the workflow operation, and collects the operation results. An enactment engine usually does not have a fixed set of predefined functions, but instead, it carries application-specific actions, which can be loaded and modified on the fly. This allows a grid node running the workflow service using the enactment engine to adjust its capability and play different roles to accommodate changes in the grid organization environment and computational requirements. Figure 2 shows the architecture of a workflow service using an enactment engine.

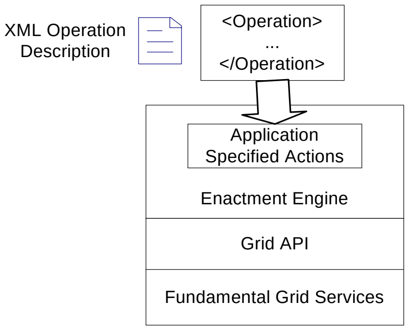

XML Operation Description
<Operation>
...
</Operation>
Application Specified Actions
Enactment Engine
Grid API
Fundamental Grid Services

Figure 2. Architecture of a Grid Agent Providing Workflow Service using an Enactment Engine

An agent providing workflow services in a grid-computing environment utilizes the underlying management grid facilities including distributed communication, object storage, database access, job schedule, GUI, and grid resource management. The fundamental grid services enable the workflow agent to carry data, knowledge, and objects, and execute programs. The data and control logic described in XML form its adaptable part. Their application-specific behaviors are obtained from the workflow process, which instruct the agent to perform high-level operations. Based on these application-specific data, the workflow agent is capable of allocating appropriate grid resources and then processes the corresponding computational operations specified in the workflow. Workflow systems provide flow control for e-science process automation. An e-science application often involves multilevel collaborative operations. Each operation represents a computational piece of work that contributes to the e-science process.

E-science computing processes may be thought of as a kind of multi-agent cooperation, in the sense that an individual or a group of grid workflow agents can be used to perform an operation in a workflow, and a workflow can be used to orchestrate or control the interactions between grid services or agents. Multiple grid services or agents working cooperatively may accomplish a particular part of the workflow process, such as a single computational operation. When an operation is complete in a grid workflow agent, based on the workflow description carried in the XML, the workflow control logic and data will be passed to the next workflow agent for the operation at the next step. Therefore, multiple grid workflow services can cooperate to provide plug-in workflow support.

3.2 Aggressive Grid Sub-Workflow Scheduling and Computational Result Validation

The nodes that provide CPU cycles in a grid system will most likely have computational capabilities that vary greatly. A node might be a high-end supercomputer, or a low-end personal computer, even just an intelligent widget. In addition, these nodes are geographically widely distributed and not centrally manageable. A node may go down or become inaccessible without notice while it is working on its task. Correspondingly, an organization will have greatly varying computational capabilities as well. Therefore, a slow node or organization might become the bottleneck of the whole e-science process. A delayed operation or sub-workflow might delay the accomplishment of the e-science process while a halted subtask might prevent the process from ever finishing.

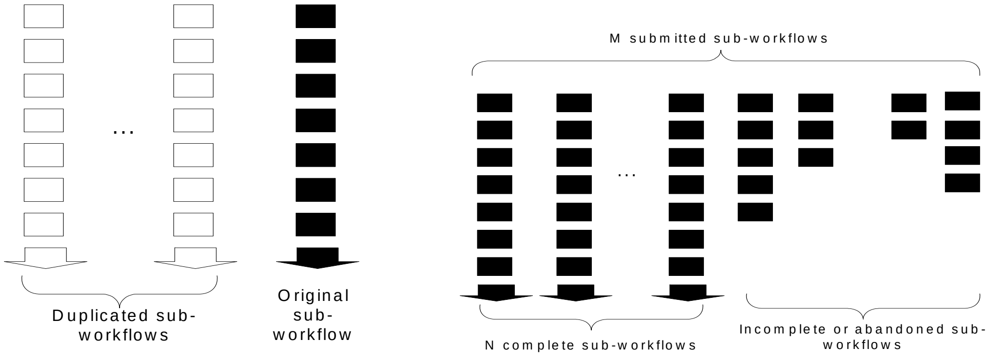


Figure 3. The Replicate Scheduling Schema and the N-out-of-M Scheduling Schema

To address this issue in grid computing, a replicate scheduling mechanism can be used. In replicate scheduling, in addition to the original sub-workflow, replicate sub-workflows are carried out in other organizations as well. When one of these sub-workflows is complete, the workflow can move forward. For those applications that exhibit statistical characteristics, such as most Monte Carlo applications, a more aggressive scheduling mechanism, the N-out-of-M strategy [Li, Y. and Mascagni, M., 2002], can be applied. The N-out-of-M scheduling schema dispatches M sub-workflows based on different random sample sets while only N (N < M) partial results are actually needed. When the N sub-workflows are complete, the e-science workflow can move ahead to the next step and ignore the other M – N computations. Compared to replicate scheduling, the N-out-of M scheduling is more effective [Li Y. et al, 2003; Li, Y. and Mascagni, M., 2003]. Figures 3 illustrates the schemas of both replicate scheduling and N-out-of-M scheduling.

Another advantage of both duplicate and N-out-of-M scheduling is that by scheduling more sub-workflows than needed, one obtains a way to validate the computational results on the grid and detect the possible misbehavior of a grid node. Duplicate checking [Aktouf C. et al, 1998] or majority vote methods [Sarmenta, L. F. G., 2001] can be employed in a duplicate scheduling schema by comparing the results of the same sub-workflows executed in different organizations – a mismatch indicates possible errors or malfunctions. The partial result validation approach [Li, Y. and Mascagni, M., 2002] can be applied in N-out-of-M scheduling by checking the statistical nature of the partial results to identify suspiciously erroneous results. These approaches can significantly improve the reliability and trustworthiness of e-science computations on the grid.

3.3 Application-Level Checkpointing

While performing large-scale computations on the grid offers many advantages, handling possible failures becomes an important concern. A grid workflow corresponding to an e-science process can usually be decomposed into multiple sub-workflows, which may be executed in different organizations and may take a long time to complete. As the number of organizations and grid resources involved increases, the chance of a failure of the e-science process during the computation increases exponentially. Thus, it is vital that a checkpointing and rollback mechanism is incorporated in the e-science workflow management.

Some grid computing systems implement a process-level checkpoint. Condor, for example, takes a snapshot of the process’s current state, including stack and data segments, shared library code, process address space, all CPU states, states of all open files, all signal handlers, and pending signals [Livny M. et al, 1997]. On recovery, the process reads the checkpoint file and then restores its state. Since the process state contains a large amount of data, processing such a checkpoint is quite costly. Also, process-level checkpointing is very platform-dependent, which limits the possibility of migrating the process-level checkpoint to another node in a heterogeneous grid-computing environment.

In grid-based e-science applications, an application-level checkpointing approach can be used to implement a portable and efficient backup and recovery mechanism. At the application level, the program can call a set of checkpointing subroutines to checkpoint a program's data structures and to later restart that program from the checkpointed data. This method is highly portable since it uses no machine-dependent constructs in creating a checkpoint. With this method, the e-science application developer is responsible for writing a program with a structure that is simple enough that the program counter and stack need not be saved. Many scientific applications (potential e-science applications) meet this requirement.

4. CONCLUSIONS

E-science in certain contexts can be thought of as a dynamic distributed computing environment. In this paper, we discussed a solution for placing cooperating and embedding grid services into a workflow to implement e-science process automation, taking advantage of the tremendous amount of computational power and connectivity in a grid-computing environment. We proposed the implementation of grid workflow services with flexible and adaptable behavior to address the issues of dynamism, heterogeneity, reliability, performance, and cross-domain computing in grid-based e-science applications. Multiple grid services can plug in a workflow to carry out complicated scientific computing process.

Nevertheless, more practical and theoretical works still needs to be done in this area to address many other issues including security, privacy control, trustworthiness, scalability, and availability of running e-science applications on the grid. These will become the next phase of our research. At the same time, we plan to develop a workflow portal – a state-of-the-art approach to develop and deploy grid-based e-science applications. More aggressively, the workflow technique can also be extended to other e-society computing applications, although different e-society applications clearly have different requirements.

REFERENCES

Allen, R., 2001. Workflow: An Introduction. Workflow Handbook 2001, *Workflow Management Coalition.*

Foster I. et al, 2001. The Anatomy of the Grid. International Journal of Supercomputing Applications, Vol. 15, No. 3.

Fox, G. and Walker, D., 2003. e-Science Gap Analysis, *Global Grid Forum.*

Chen Q. et al, 1999. Dynamic-Agents, Workflow and XML for E-Commerce Automation. *International Journal on Cooperative Information Systems*, pp. 314-323.

Foster I. et al, 2003. The Physiology of Grid: Open Grid Services Architecture for Distributed Systems Integration, *draft*.

Goble, C. and Roure, D. D., 2003. The Grid: An Application of the Semantic Web. *Grid Computing: Making the Global Infrastructure a Reality*, pp. 437-470.

Cao J. et al, 2003. GridFlow: Workflow Management for Grid Computing. *Proceedings of 3rd International Symposium on Cluster Computing and the Grid, CCGRID2003*.

Fisher, L., 2002. Workflow Handbook, *Workflow Management Coalition*.

Bivens, H. P., 2001. Grid Workflow. Grid Computing Environments Working Group. *Global Grid Forum.*

Li Y. et al, 2003. A Grid Workflow-Based Monte Carlo Simulation Environment. *Journal of Neural Parallel and Scientific Computations (NPSC) Special Issue on Grid Computing.*

Li, Y. and Mascagni, M., 2003. Analysis of Large-scale Grid-based Monte Carlo Applications. *International Journal of High Performance Computing Applications (IJHPCA)*. Vol. 17, No. 4, pp. 369-382.

Folding@home, 2000. Folding@home Distributed Computing. *http://folding.stanford.edu.*

SETI@home, 2002. SETI@home: the Search for Extraterrestrial Intelligence. *http://setiathome.ssl.berkeley.edu.*

Li, Y. and Mascagni, M., 2002. Grid-based Monte Carlo Applications. *Lecture Notes in Computer Science, 2536:13-24, GRID2002, Grid Computing Third International Workshop/Conference, Baltimore.*

Aktouf C. et al, 1998. Basic Concepts & Advances in Fault-Tolerant Computing Design. *World Scientific Publishing Company*.

Sarmenta, L. F. G., 2001. Sabotage-Tolerance Mechanisms for Volunteer Computing Systems. *Proceedings of ACM/IEEE International Symposium on Cluster Computing and the Grid (CCGrid'01).*

Livny M. et al, 1997. Mechanisms for High Throughput Computing. *SPEEDUP Journal*, Vol. 11, No. 1.