Proceedings of the 5th WSEAS International Conference on Applied Computer Science, Hangzhou, China, April 16-18, 2006 (pp526-532)

Decision Support System Based on Grid Workflow

Mingsheng Hu[1 2],Zhijuan Jia[2],Xueguang Chen[1]
(Institute of Systems Engineering, Huazhong University of Science and Technology, 430074, P.R.China)1
(Institute of Software Science, Zhengzhou Teachers College, 450044, P.R.China)2

Abstract: Grid technology brings about a lot of innovative ideas and technologies for the development of Decision Support System. The existence of description languages, semantics, service publishing methods and Grid middleware have up to now enabled interoperability between decision resources available across the Grid environment. However, there is an emerging need for the ability to dynamically discover how available services, resources and data could be utilized not only to process a task, but to achieve the desired outcome in the most suitable manner. In this paper, we investigates the emerging need for Grid workflow to aid decision support on the Grid and proposes a Grid workflow scheduling system that is a Petri net-based graph model and incorporates a reasoning system designed to perform such a task.

Key-Words: Decision Support System; scheduling; Workflow; Grid

1 Introduction

Decision Support Systems (DSS) are computer-based information systems that help decision-makers solve half-structured or non-structured problems by using data and models. They enable decision-makers to make decisions more effectively. The Web-based DSS have made information sharing on the Internet possible, but they cannot meet the decision-maker's needs in the heterogeneous, autonomic, dynamic and distributed decision support environment, because they only link web pages and lack global mechanism to manage and coordinate decision support resources on the Internet. However, as an advanced technology representing "the third internet revolution", Grid appears as an effective technology coupling geographically distributed resources for solving large-scale problems in wide area network, which support open standard and dynamic services. In addition, it provides highly intelligent communication between computers and human. These characteristics are very suitable for the constructing need of DSS platform. It will improve DSS greatly, and bring profound revolution to DSS theory and its application.

In general, the major problems in this domain relate to the Grid Decision Resource (GDR). However, GDR Management (GDRM), GDR Scheduling (GDRS) and usage models in these environments is a complex undertaking. This is due to the geographic distribution of GDR that are owned by different organizations or peers. The owners of each of these GDR have different usage or access policies and cost models, and varying loads and availability. The existence of description languages, semantics, service publishing methods and Grid middleware have up to now enabled interoperability between decision resources available across the Grid environment. However, there is an emerging need for the ability to dynamically discover how available services, resources and data could be utilited not only to process a task, but to achieve the desired outcome in the most suitable manner. In this paper, we investigates the emerging need for Grid workflow to aid decision support on the Grid and proposes a Grid workflow manager system that is a Petri net-based graph model and incorporates a reasoning system designed to perform such a task.

2 Related Work

With the advent of grid technologies, the idea of Grid-based Open DSS (GBODSS) is becoming a reality. As an application of Grid in the field of DSS, we put forward an improved model of Agent Grid-based Open DSS (AGBODSS) [1].

The improved model characterizes the relationship between the components and layers of the GBODSS and illustrates its openness and dynamic characteristics more clearly.

The Agent Grid layer is the core of the AGBODSS model. This layer has direct connections and interfaces with the other components of the system. The Agent Grid layer is composed of several registered agent groups and grid services, and each agent group is composed of some agents. The Agent Grid provides many Agent Grid services to support the management, interoperation and integration of the agents in this

layer. Presently, the Agent Grid services include register service, brokerage service, logging service, security service and visualization service, etc. Agents in the Agent Grid can be divided into several agent groups according to their type to facilitate the management and utilization of these agents.

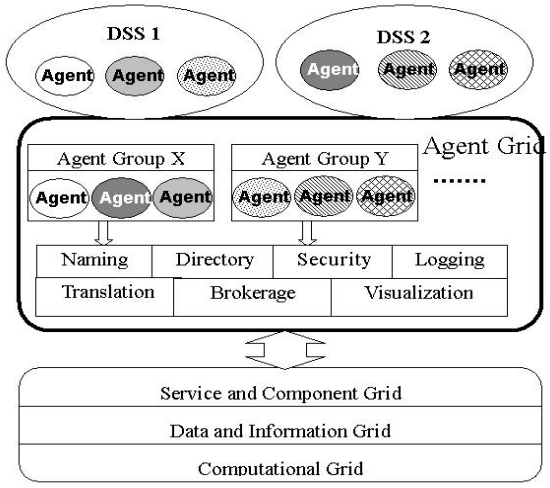

Fig. 1.Open DSS Model Based on the Agent Grid

Considering practical problems, the designer of the DSS can construct different DSSs easily through a quick integration of the agents in the Agent Grid layer. The designer of the DSS needs firstly to design the structure of the AGBODSS, and then chooses suitable agents in the Agent Grid to play the roles in the AGBODSS.

Grid technologies enable the sharing, exchange, discovery, selection and aggregation of geographically distributed, heterogeneous resources—include GDRs. As Grids comprise of a wide variety of GDRs owned by different organizations with different goals, they present a number of challenges in managing shared distributed GDRs. To address these challenges, a distributed computational economy mechanism has been proposed. This mechanism would support efficient management of distributed GDRs by regulating the supply and demand through differentiated pricing strategies and promote sustained sharing of resources by providing incentive for GDR Providers. It also encourages GDR Consumers to specify their Quality-of-Service (QoS) needs based on their actual requirements. Therefore, the use of Grid economy mechanism in managing shared GDR s would lead to the creation of a Grid Market-Place (GMP).

We proposed and explored the usage of an economics-based paradigm for managing GDR allocation in Grid computing environments [2]. The economic approach provided a fair basis in successfully managing decentralization and heterogeneity that is present in human economies. Competitive economic models provide algorithms/policies and tools for GDR sharing or allocation in Grid systems. The models can be based on bartering or prices. In the bartering-based model, all participants need to own GDRs and trade GDRs by exchanges. In the price-based model, the GDRs have a price, based on the demand, supply, value and the wealth in the economic system. Consequently, for the market to be competitive and healthy, coordination mechanisms are required that help the market reach an equilibrium price—the price at which the supply of a service equals the quantity demanded.

Research on Economic models for GDRM suggests that GMP-based architecture is a good solution for managing GDRs because GMP-based links between companies are more efficient than point-to-point links between every buyer and every supplier. We envision a future in which economically intelligent and economically motivated P2P and Grid-like software systems will play an important role in distributed service-oriented computing. Emerging Web Services and Web Services-based process definition standards provide mechanisms for formally defining GDRM that can be clearly understood and quickly deployed in a platform-independent manner. Figure 2 visually describes Service-oriented Economic Model for GDRM that satisfies the key requirements. Since Web Services-based technology solves interoperability problems related to process definition and execution, we propose to use them for implementing the marketplace system for decision resource management. In addition, the proposed Service-oriented Economic Model for GDRM contains the repository of shared ontology, message formats and templates.

Currently, the user community and the technology are still new and not well accepted and established in commercial settings. However, we believe the Grid can become established in such settings by providing an incentive to both consumers and GDR owners for being part of the Grid. Since the Grid uses the Internet as a carrier for providing remote services, it is well positioned to create a computational ecology, cooperative problem solving environment and means for sharing GDRs in a seamless manner. Up until now, the idea of using Grids for solving large computationally intensive applications has been more or less restricted to the scientific community. However, even if, in the scientific community, the

pricing aspect seems to be of minor importance, funding agencies need to support the hardware and software infrastructure for Grids. Economic models help them manage and evaluate GDR allocations to user communities. The system managers may impose quota limitations and value different GDRs with a different number of tokens. In such environments, GDR consumers certainly prefer to use economic-driven schedulers to effectively utilize their tokens by using lightly loaded cheaper resources. For more details about the AGBODSS and the GDRM, please refer to [3].

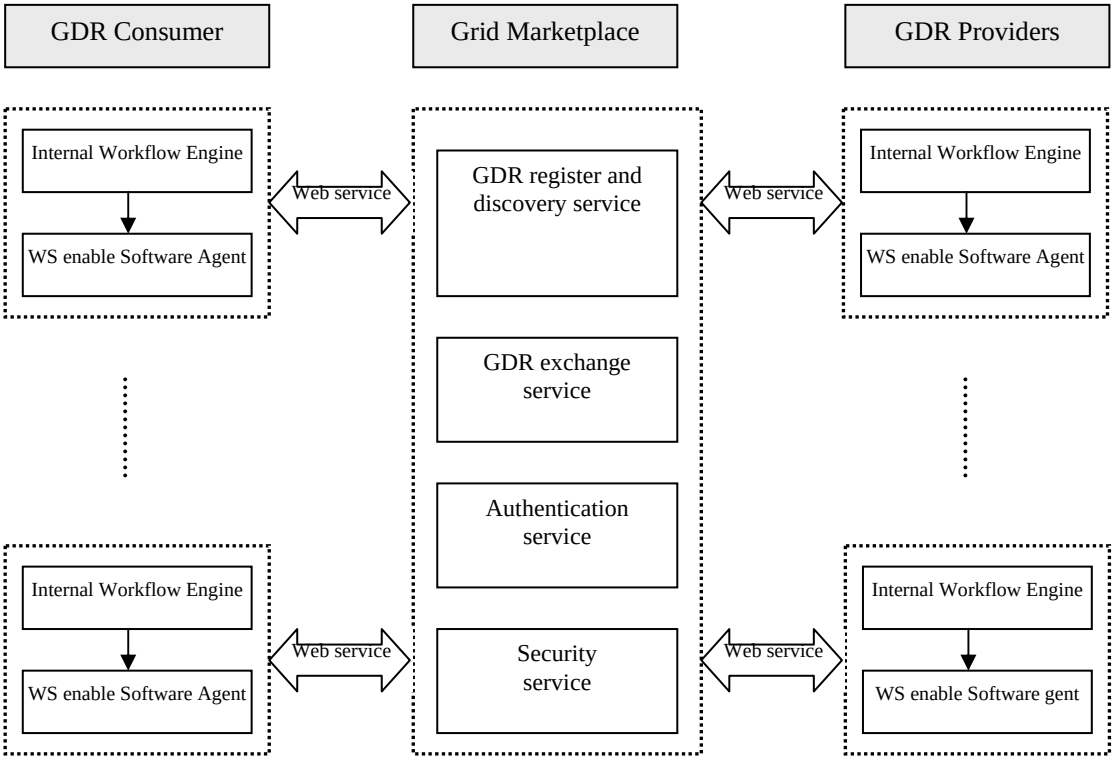

Fig. 2.Service-oriented Economic Models for GDRM

3 Grid workflow orchestration

In order to enable the user to compose complex decision support applications on distributed heterogeneous and unreliable decision resources within Grid environments, the concept of Grid workflows has emerged. It describes patterns of control and dataflow between GDRs, including human actors participating in interactions. To describe the concept of composing Grid Services, Web Services concepts can be reused. For Web Services composition, the terms orchestration and choreography are frequently used. Orchestration describes how Web Services can interact with each other at the message level, including the business logic and execution order of the interactions. Choreography expresses the sequence of messages that involve multiple services. It represents the public message exchanges that occur between Web Services, rather than a specific business process that is executed by a single party. Several techniques have been established in the Grid community in order to define the workflow of Grid jobs. A very promising approach for this purpose is the use of graphs. Graphs offer a very intuitive way of modeling abstract workflows and can be handled easily even by non-expert users. The main limitation of graphs is the fact that they may become very huge if the workflow is too complex. The solution could be the usage of hierarchal graphs that allow graph coarsening and refinement. In many Grid projects, the workflow is modeled using Directed Acyclic Graphs (DAG)—which are popular due to their simple structure, but restrict the kinds of workflows that can be modeled [4].

The OGSA working group defines a workflow as a pattern of process interaction, not necessarily corresponding to a fixed set of processes. All such interactions may be between services residing within a single data center or across a range of different platforms and implementations anywhere. The orchestration of workflows describes the ways in which these processes are constructed from Web

Services and other processes, and how these processes interact. Two main preconditions must be fulfilled in order to enable the orchestration of workflows: an adequate description of the components for deciding which components are functional to solve the problem, and a suitable workflow model for defining how the components should interact within that workflow.

The component description has several objectives:

• It assists the user as well as the Grid application developer in selecting appropriate components and gives an understanding of how the components may fit together by providing a functional and semantic description.

• Orchestration tools use the component description on a more technical level for automatic component discovery and interface check.

• During the execution of workflows, the abstract workflow elements must be mapped onto real instances of GDRs (resource mapping or brokering). Therefore, it is mandatory to describe the requirements and the technical properties of the components and decision resources.

There already exist many domain-specific approaches for describing components and resources, in particular emerging from the semantic web community (RDF, OWL, and DAML), but nevertheless there is no well-accepted standard established within the Grid computing community so far. Most of the existing work is either very specific to the infrastructure or too generic, e.g., for validating the description on the basis of an XML schema. For GBODSS, we developed a so-called GDR Definition Language (GDRDL), considering the principles of extension, inheritance, and the distinction between classes and instances.

The component model of the system supports legacy command line programs as well as OGSI Grid Services in a mixed way. For specifying the interactions between the components, it is mandatory to have an adequate workflow model and a corresponding workflow description language. We detected the following main requirements within the context of a Grid computing architecture:

• The user must be able to define the Grid workflow on an abstract level without knowledge about the infrastructure.

• The Grid workflow model should be mostly universal (e.g., Turing complete) in order to cover a broad range of workflow patterns.

• At the same time, the Grid workflow model must be simple and easy to use.

• It should support the modeling of data as well as control flow.

• Due to the dynamic and unreliable nature of GDRs, dynamic workflows are required that may change their structure during runtime.

At first sight, the requirements “universal” and “simple” seem to be quite contradictory, and actually most of the existing approaches for describing Grid workflows do not fulfill either the requirement to be “simple” (e.g., BPEL) or to be “universal” (e.g., DAG). One approach that combines both – the universality and the simplicity – are Petri nets that we use in our approach. Petri nets are well established as a model for the analysis and the design of complex discrete systems. Here, we extended this model to be usable directly for the automatic execution of workflows on common Grid middleware as well. Therefore, the abstract Petri net elements are linked to component descriptions that themselves can be mapped onto real Grid resources.

Petri Nets are a special class of directed graphs that can model sequential, parallel, loops and conditional execution of tasks. Petri nets allow the graphical definition of arbitrary workflows with only few basic graph elements – just by connecting data (files, parameters) and software components (command line programs, Grid Service method calls). Petri nets belong to a special class of directed graphs. The type of Petri nets we introduce here, are Petri nets with individual tokens (colored Petri nets) and constant arc expressions which are composed of places, denoted by circles (○), transitions, denoted by boxes (□), arcs from places to transitions (○ → □), arcs from transitions to places (□ → ○), individual and distinguishable objects that flow through the net as tokens (●), an initial marking that defines the tokens which each place contains at the beginning, and an expression for every arc that denotes an individual object. A place p is called input place (output place) of transition t if an arc from p to t (from t to p) exists. A brief introduction to the theoretical aspects of colored Petri nets can be found. Petri nets are suitable to describe the sequential and parallel execution of tasks with or without synchronization; it is possible to define loops and the conditional execution of tasks.

In most cases, the workflow within GDRS jobs is equivalent to the dataflow, i.e., the decision when to execute a software component is taken by means of availability of the input data. Therefore, the tokens of the Petri net represent real data that is exchanged between the software components. In this case, we use Petri nets to model the interaction

between software resources represented by software transitions, and data resources represented by data places. In other cases, however, the workflow should be independent from the dataflow, if you want to synchronize several activities – and in addition to the data places and software transitions behave to introduce control places and control transitions. Control transitions evaluate logical conditions. In the case that a software transition is linked to a command line program, the tokens on the output places contain the exit status of the process (e.g., done, failed), whereas in the case of Grid Services, the tokens store the output parameter returned by the method call. The state of the workflow is represented by the marking of the Petri net, which makes it easy to implement tools for workflow check pointing and recovery.

4 Workflow Scheduling System

The workflow scheduler needs to coordinate with diverse local management systems as Grid resources are heterogeneous in terms of local configuration and policies. Taking into account users' QoS constraints is also important in the scheduling process so as to satisfy user requirements. Given the complexity of the grid workflow execution, we have designed a decentralized scheduling system which supports just in-time planning and allows the decisions of the resource allocation to be made and changed at run-time [5].

We believe that decentralized scheduling architecture is more efficient over the centralized scheduling for complex workflow processing, which handles all tasks by one scheduler [6]. In our system, every task has its own scheduler called task manager (TM) which implements a scheduling algorithm and handles the processing of the task, including resource selection, resource negotiation, task dispatcher and failure processing. The lifetimes of TMs, as well as the whole workflow execution, are controlled by a workflow coordinator (WCO).

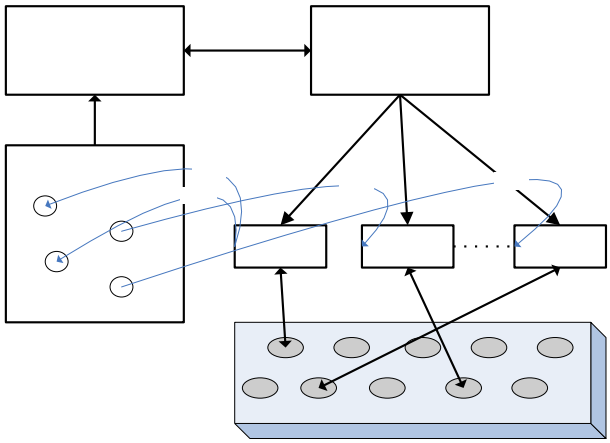

Fig. 3. Workflow Scheduling System

As shown in Figure 3, dedicated TMs are created by WCO for each task. Each TM has its own monitor which is responsible for monitoring the health of the task execution on the remote node. Every TM maintains a resource group which is a set of resources that provided services required for the execution of an assigned task. TMs and WCO communicate through a event service server (ESS).

In the system, the behaviors of task managers and workflow coordinator are driven by events. A task manager is not required to handle communication with others and only generates events according to task's processing status. At the same time, the task managers take actions only according to occurred events without concern for details of other task managers.

The event notification is based on subscription-notification model. WCO and TMs just subscribe the events of interest after activation, and then they can do whatever they want. When a subscribed event occurs, they will be informed.

The benefit of the event-driven mechanism is it provides a loosely-coupled control; hence the design and development of the system are very flexible.

5 Case study

We take the National Economy Mobilization DSS (NEM-DSS) as a practical case. The National Economy Mobilization is a range of government activity to schedule economic and social resources for emergent affairs. The aim of the NEM-DSS is to implement unified and effective management of automobile mobilization activities, including generating plans, simulating plans and executing plans. It helps managers increase the effectiveness of making decisions.

The automobile mobilization involves several kinds of entity units, including components unit, assemble unit, transporting unit, warehouse and mobilization command center. Each entity unit is an autonomic one that needs interoperation and negotiation to form a final plan that should be coordinated in execution.

Based on the above analysis and design steps, we should first create agent model for each entity involved in the automobile mobilization. The design for each entity agent includes its plans, competence, product model, inner spiritual states (e.g. beliefs, goals, intentions), communication module and reasoning module, etc. components unit agent, assemble unit agent, transporting unit agent, warehouse agent and mobilization command center agent are created at this stage.

The AGBODSS has superior openness and excellent scalability. The entity agents can join and leave the AGBODSS dynamically according to the changing circumstances and tasks. The entity agents can sketch out mobilization plans through communication and negotiation, and they can coordinate to solve the problems in the execution of plans. The AGBODSS can efficiently utilize all kinds of decision-making support services and resources on the Grid system.

To completing these complicated decision resources scheduling availably, we applied the Grid approach presented in this article to AGBODSS in order to enhance the scalability and extensibility of the system, and to have a test case to gain experience with a “real world” application in a service-oriented Grid architecture. The Petri net serves as model for the control flow and data flow and it helps in configuring the state management of the application.

Another interesting concern is the life-cycle management and the handling of either stateful or stateless Grid Services. The life-cycle management takes care about when a service instance has to be created and destroyed and if it is a stateful or stateless service.

In our case study, it is very important that the adaptation component preserves its state, as the forecast is based on past data. For this reason, the adaptation component is stateful and is created once per workflow, whereas the Optimization component – for which historic aspects are irrelevant – is stateless. Yet, another benefit of using Grid Service technology is the possibility to offer the service interface as a Web Service. For the Visual-Web component, we developed an internal and an external interface. The external one is for customers to upload the measuring data and to download new prediction data. It can be used with any Web Service client library. The internal interface, however, is only accessible for internal callers and is secured by GSI methods.

7 Conclusion and Future Work

In this paper, we investigates the emerging need for Grid workflow to aid decision support on the Grid and proposes a Grid workflow scheduling system that is a Petri net-based graph model and incorporates a reasoning system designed to perform such a task. The innovation in respect to former work in this domain is the incorporation of a Petri net-based workflow model for orchestrating Grid Service method calls as well as legacy command line applications within a single workflow. This approach allows the definition of arbitrary workflows, including conditions and loops, with only three different abstract graph elements: transitions, places, and arcs. Grid Services and legacy command line programs can be integrated easily to a single, loosely coupled Grid application, regarding the dataflow as well as the control flow.

Our future work will focus on workflow execution optimization. We will be extending resource allocation algorithms to support optimal and QoS (Quality of Service) requirements based scheduling, using computational economy. In addition, we will assist its users in composing powerful grid workflows by means of a self-adapting expert system. All interactions in AGBODSS with the Grid environment will be monitored and evaluated.

References:

[1] Xueguang Chen□a study on the framework of Grid based Decision Support Systems □ proceedings of 2003 International Conference on Management Science & Engineering □ Atlanta

[2] Mingsheng Hu,Xueguang Chen,Zhijuan Jia. Decision Resource Management and Scheduling on the Grid. Greece:1st WSEAS International Symposium on GRID COMPUTING, 2005:3-5

[3] Mingsheng Hu,Xueguang Chen. AGBODSS□ Agent Grid-Based Open Decision Support System. WSEAS transacyions on information science and applications[J],2005,8:2-8

[4] Falk Neubauer, Andreas Hoheisel , Joachim Geiler , Workflow-based Grid applications.

Future Generation Computer Systems,2006, 22:6–15

[5] Jia Yu , Rajkumar Buyya. A Novel Architecture for Realizing Grid Workflow using Tuple Spaces. In 5th IEEE/ACM International Workshop on Grid Computing (GRID 2004), Pittsburgh, USA, IEEE Computer Society Press, Los Alamitos, CA, USA, Nov. 8, 2004.

[6] Mingsheng Hu, Application of PBS Based on Grid Computing. MICROCOMPUTER & ITS APPLICATIONS, China, Vol.24 No.6, 2005, P.7-10