Leveraging Legacy Workflow Capabilities in a Grid Environment

Zhang Liyong[1, 2] Zhao Zhuofeng[1] Li Houfu[1, 2]
1 Research Centre for Grid and Service Computing, Institute of Computing Technology, Chinese Academy of Sciences, 100080, Beijing, China
2 Graduate University of the Chinese Academy of Sciences, Beijing, 100039, China
{zhangliyong, zhaozf, lhfsday}@ software.ict.ac.cn

Abstract

A large number of workflow systems are in existence, offering various orchestration capabilities. How to make full use of the legacy workflow capabilities in a grid environment is of importance for saving investments as well as enabling dynamic collaboration and boundary-crossing problem solving. In this paper, we propose a concept model for leveraging and aggregating legacy workflow capabilities. The model contains a general way to describe capabilities of legacy workflow systems, two main abstractions for transparent workflow capability usage and an abstract way for boundary-crossing workflow definition. The presented is evaluated with a case study.[1]

1. Introduction

As one of the most important kinds of middleware for integrating heterogeneous and distributed resources, workflow systems have been used in many business domains for decades. IBM WebSphere MQ Workflow[2] and Microsoft BizTalk Orchestration[3] are well-known workflow tools among many others. In resent years, with the advent of grid computing, grid workflows that enable regulated resource scheduling and problem solving on top of grid utilities and grid services are widely applied to scientific computing area, e.g. Taverna [1], Kepler [2], etc.

From the perspective of grid computing, legacy workflow systems should be candidates to be included into a grid as a specific kind of resources providing orchestration capabilities. However, workflow systems have evolved in isolation and vary with each other in many aspects, from workflow model to applying pattern and from engine interface to hosting environment. The heterogeneity makes it much complicated for a grid environment to integrate them. It will be beneficial to develop a common framework, so that various legacy workflow systems can be integrated and leveraged in a relatively ease way [3]. Besides, users' requirements for cross organizational research collaborations and cross-disciplinary knowledge sharing become open and often. It may bring grid users much convenience if workflow applications can be developed in a uniform way while their enactment can cooperatively realized by several workflow systems.

We have developed an end-user oriented and service-based workflow system called VINCA [4], which provides a personal problem solving environment for building service-oriented applications from business level. Several core concepts and implementing details, including service virtualization, business service, service community, end-user programming etc., have been investigated and can be found in our previous work[4] [5] [6].The motivation of this paper is to extend VINCA to a meta-workflow system, so that it can incorporate other workflow capabilities in several business and scientific communities. To this end, a concept model is proposed, in which the capabilities of legacy workflow applications and the enactment engines are virtualized as abstract services that can be further used to construct applications from business level. A prototype of the meta-workflow system is implemented, and the presented work is evaluated with a case study.

The rest of the paper is organized as follows. In section 2, we analyze the requirements and problems from real world scenarios. In section 3, we propose and illustrate the concept model in detail. A case study

[1] This work is supported in part by the National Science Foundation of China (Grant No. 90412010) and China Ministry of Science and Technology 863 Program (Grant No. 2006AA01A106)
[2] http://www.ibm.com
[3] http://www.microsoft.com

and discussion are given in section 4. Finally, we summarize in Section 5.

2. Problem Definition

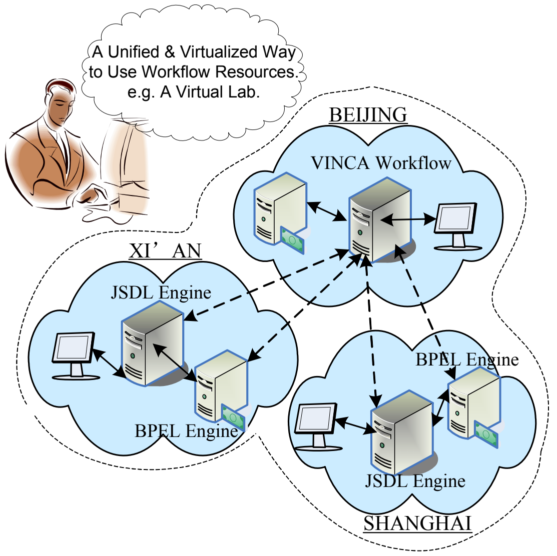

Figure 1. Workflow Systems in CNGrid

The work presented in this paper is actually part of the CNGrid[4] project supported by the 863 Program of Chinese Ministry of Science and Technology. As shown in figure 1, many workflow systems as well as Web service and grid services exist in the CNGrid, such as BPEL4WS (Business Process Execution Language for Web Services) [7] Engines, JSDL(Job Submission Description Language) [8] processing Engines, etc. It is worth mentioning that although JSDL is not a workflow language as declared in its specification, the behavior of JSDL processing engines exhibits much similarity to workflow engines. Therefore, we just view them as a special kind of workflow engines. When applying the CNGrid into real world scenarios, we find that it's often required to coordinate various engines and services. For example, a BPEL Engine and several services may be employed to build a virtual laboratory. However, it's difficult and even impossible to achieve such goals by using any workflow definition language in existence. It's urgent and challenging to integrate these computing capacities in a unified form. The difficulties include:

1) Besides describing legacy workflow system as a general resource in a grid environment, some special feature of workflow should be captured and managed in order to make legacy workflow systems sharable.
2) It is burdensome for users to fulfill their boundary-crossing needs with several different workflow systems directly. Some abstractions of the legacy workflow resources should be introduced to reduce such complexities.
3) Furthermore, a language that can ease the construction of workflow applications based on the above-stated abstractions should be given to describe the boundary-crossing logics, so that users can be relieved from learning different workflow languages.

The goal of the VINCA meta-workflow is not to design yet another workflow system, but rather to provide a general model to leverage and aggregate capabilities of existing heterogeneous workflow systems exist in the CNGrid. By meta-workflow we mean that the model supports not only choreography of general tasks but also an aggregation of existing heterogeneous workflows or a hierarchy of workflows. In the next section, the concept model of VINCA meta-workflow will be discussed in detail.

3. Concept Model of VINCA Meta-Workflow

Motivated by the above-stated consideration, we propose a VINCA Meta-Workflow Framework, which includes a unified way for capability description of legacy workflow systems, two main abstractions for transparent workflow capability usage and an abstract way for boundary-crossing workflow definition.

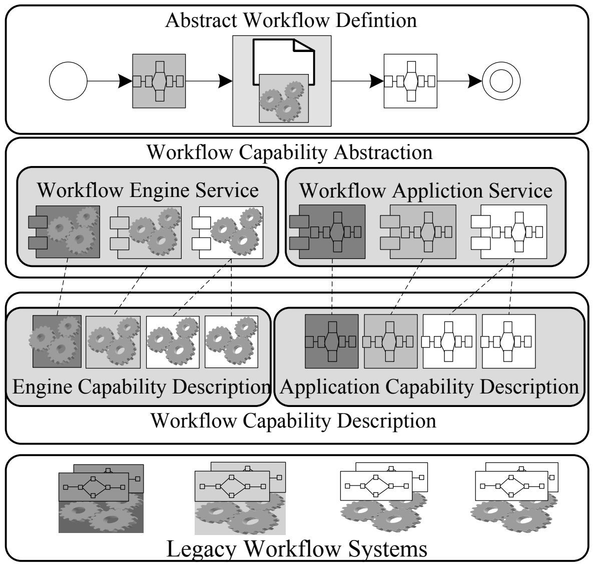
Abstract Workflow Defintion
Workflow Capability Abstraction
Workflow Engine Service
Workflow Appliction Service
Engine Capability Description
Application Capability Description
Workflow Capability Description
Legacy Workflow Systems

Figure 2. Concept Model

[4] http://www.cngrid.org/

The concept model of the Framework is shown in Fig. 1. Four abstraction layers are identified from the bottom up as follows:

- **Workflow Resource Layer**

This layer represents the underlying legacy workflow systems that can be deemed as a specific kind of resources providing orchestration capabilities from the view of grid computing. From the view of workflow users, we consider that a specific workflow system consists of an enactment engine for workflow execution and some deployed workflow applications that represent different business processes. Though we ignore the diversity of these systems, we do pose a hypothesis that the two kinds of capabilities can be accessed in a certain manner, so that the legacy workflow systems can be leveraged in a unified environment.

- **Workflow Capability Description Layer**

This layer intends to provide general and uniform descriptions on the underlying resource. Generally, a workflow system is often much complicated with various functions and modules. Currently, we only concentrate on leveraging two kinds of workflow capabilities provided by the workflow engine and the deployed workflow applications respectively. Therefore, two specific descriptions are included in this layer, namely the *workflow engine capability description* and the *workflow application capability description*.

The *workflow engine capability description* gives engine's orchestration capability from outside. Though the underlying engines vary one from another, they do all take a set of activities that represent inner steps in the execution of the workflow. The model is made up of three aspects about the engine. The *basic information* depicts the engine's name, type, version, provider, etc. The *Interface set* contains all the interfaces and the relevant protocols to use and monitor the engine, e.g. interfaces using RPC to create a workflow instance, to change the state of tasks, to abort an instance, etc. A *QoS specification* is included in the model to specify the non-functional aspects of the engine, such as the performance capability, reliability, hosting environment, etc.

The *workflow application capability description* regulates the description of deployed workflow applications that can be viewed as domain knowledge resources for reuse and repurposing. This description is much similar with the *engine capability description* and depicts the application in three aspects too. The *basic information* contains the process's name, description, type, version, etc. The *functional specification* depicts the application's function in three facets, including inputs, outputs and data type. The *QoS specification* regulates the non-functional facet of the application, e.g. the cost and time for per execution.

- **Workflow Capability Abstraction Layer**

The abstract service layer provides generic abstractions of the underlying workflow resources based on the description of workflow capabilities. The main purpose of this layer is to hide the complexity and heterogeneity of legacy workflows. Here, business service, a unified abstraction mechanism for grid resources, is utilized. Business service was proposed as a business-level concept which virtualizes IT entities for end-users in our previous VINCA project [5]. According to the workflow capabilities which are captured in workflow capability description layer, we coin two special business services by virtualizing workflow engines and legacy workflow applications, which are called *workflow engine service* and *workflow application service* respectively

A *workflow engine service* is a logic view of engines' capability and is formed by abstracting the underlying workflow engines. Two cases may occur during the abstraction. Firstly, engines with the same capability are virtualized to an abstract engine service. Secondly, engines whose capabilities can be coordinated to provide a compound capability can also be virtualized into one engine service. That is to say a one-to-many relation is maintained between the abstract engine service and the concrete engines. Issues for virtualization is much complex and is out of the concern of this paper. Details for virtualization of Web service can be found in the work [5]

Similarly, a *workflow application service* models the workflow applications deployed on workflow engines from a business level. It may employ one or more concrete applications to fulfill the specified business function.

- **Abstract Workflow Definition Layer**

In order to shield the different workflow definition way for end-users, an abstract workflow definition layer is introduced as the top-level of the concept model. On this layer, an end-user oriented workflow definition language VINCA is adopted as a simplex workflow definition way for building boundary-crossing business process. VINCA provides an abstract workflow definition way through utilizing business service as a unified executor for business activities. With the help of VINCA, users can use the specific business services in workflow capability abstraction layer to represent their requirements, so that they can concentrate on their business logic while ignoring the real workflow resources.

The main features of the concept model lie in two aspects. Firstly, with the general and basic description framework, various legacy workflows can be

IEEE COMPUTER SOCIETY

abstracted in a unified form despite the discriminations. Secondly, the workflow capabilities are further virtualized as services that can be used by end-users. The services hide the implementing details, so that users can only concentrate on their business logics while ignoring the complex workflow resources.

4. Case Study

4.1. A Use Case

In this section, a simple use case is given to illustrate how existing workflow resources in CNGrid can be aggregated to build a "Virtual Lab ". This use case contains three steps. First, the genome sequence is aligned. Then a SVG picture is created by analyzing the result data of genome alignment. Finally, the picture is transferred to a web server so that users can retrieve it. Three different kinds of resources are incorporated to build an experiment workflow to implement the use case, as shown in the top part of figure 3. The first task T1 is implemented as a BPEL process that is deployed on a remote ActiveBPEL[5] Engine. The second task T2 is a batch job defined in a JSDL file that can be processed by several GridSam[6] engines in CNGrid. The last task T3 is done by invoking a Web service.

4.2. Realization of the Use Case

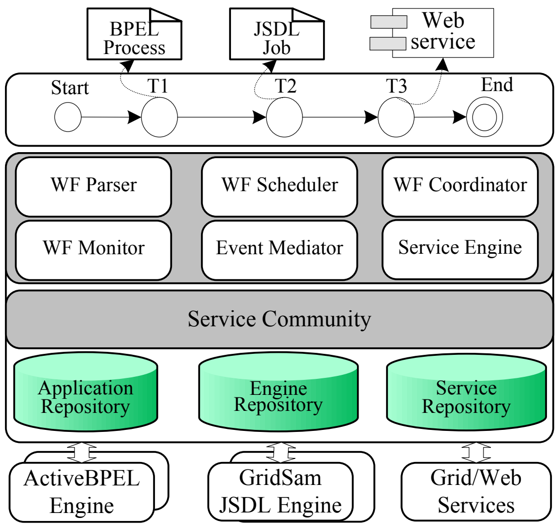

Figure 3. Implementation Architecture

Figure 3 shows the overall architecture of the implementation. Now, we illustrate how it works.

[5] http://www.active-endpoints.com/
[6] http://www.omii.ac.uk/mp/mp_jobsubmission.jsp

(1) The capability information of the ActiveBPEL engine and the GridSam Engine is registered into the engine repository, including the engine's name, type, URL and the interface endpoints. In the use case, one ActiveBPEL engine and two JSDL engine are registered.
(2) The information of pre-deployed BPEL process for genome sequence alignment is registered into the application repository. The process's name, type and the access endpoint is provided. The Web service for transferring picture is registered into the service repository and the JSDL job file is pre-created too.
(3) After registration, the virtualization mechanism is applied and three workflow capability services are generated, which are the BPEL engine service, the JSDL engine service, and the BPEL process service. It's notable that only one JSDL engine service is generated while there are two concrete JSDL engines. The virtualization maintains a one-to-many relation between the engine service and the concrete engines. The virtualized services and the mapping relations are managed in the service community that provides a logic and unified view of the underlying grid resources.
(4) A VINCA workflow definition is constructed and submitted to execution utilities. An event-based meta-workflow engine is designed to enact the workflow by scheduling, activating and coordinating the underlying engines. When the JSDL job is handled, the meta-workflow engine can dynamically choose and schedule the job onto a concrete GridSam engine. Various scheduling strategies can be employed to schedule a concrete engine. We use the strategy by which the engine with few running jobs is chosen.

4.3. Discussion

From the use case, we can see by using abstraction and virtualization, many advantages are achieved. First, the heterogeneity of workflow engines is hidden. For grid users, they are all services that provide certain orchestrating capabilities. Secondly, the complexity and difficulty for reusing legacy workflow applications are reduced. Users may concentrate only on the business logics and use the application resources transparently. Thirdly, the workflow capabilities can be used optimally by a variety of scheduling strategies. Therefore, a better performance can be achieved.

Our work in this paper is an initial effort to leveraging and aggregating capabilities of legacy workflow engines. The concept model is sound and feasible as demonstrated by the scenario above.

However, Due to the complexity and diversity of legacy workflow systems, many challenging problems exist and require further and deeper research before the model can be applied extensively.

In fact, several efforts have been taken to reuse workflows and to aggregate workflow engines. In ^{my}Grid[7], Workflow fragments are repurposed between different research groups with the intension that fragments from one group can support research in another. However, the reuse of fragments is limited in ^{my}Grid environment. In Kepler [2], a framework is proposed to build a new grid workflow system by reusing and altering the components of existing ones. CyberIntegrator [9] integrates workflow engines by wrapping each of them as an executor by hard-coding. Wf-XML provides a way to merge process-oriented tools into the generic invocation framework [10]. It focuses on the protocol that enables the interoperation between workflow engines, but nothing is proposed to aggregate engines. Though the above works are different to ours to some extent, there are still many inspiring ideas that can be referenced by our work.

5. Summary

How to make full use of existing workflow systems to solve boundary-crossing problems in a grid environment is an urgent and challenging problem. In this paper, we have investigated the difficulties and challenges and proposed a concept model for reusing legacy workflow capabilities. In this model, a basic description framework guarantees that a unified view of the legacy workflows can be formed despite the discriminations. Further more, the abstraction of workflow capabilities enables users to concentrate on their business logics while ignoring the complex workflow resources. In addition, a meta-workflow system with a real world scenario is designed and developed, which shows the concept model is feasible to be applied to solve boundary-crossing problems.

However, many challenges are encountered and art to be examined. The future work includes (1) to design a more effective virtualization mechanism to facilitate the generation of abstract services, (2) to define a model for optimal scheduling strategy by which the "best" engine is selected for the abstract engine service, (3) to invent a data type converting mechanism to enforce improve the interoperability between different workflow engines, and (4) to devise a reliable and secure protocol to coordinate and monitor incorporated engines.

[7] http://www.mygrid.org.uk/

6. References

[1] T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Senger, M. Greenwood, T. Carver and K. Glover, M.R. Pocock, A. Wipat, and P. Li, Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics, Oxford University Press, London, UK, 2004, 20(17):3045-3054,

[2] I. Altintas, A. Birnbaum, K. Baldridge, W. Sudholt, M. Miller, C. Amoreira, Y. Potier, and B. Ludaescher, A Framework for the Design and Reuse of Grid Workflows, International Workshop on Scientific Applications on Grid Computing (SAG'04), LNCS 3458, Springer, 2005.

[3] E. Deelman, Y. Gil, Workshop Final Report[A], NSF Workshop on Challenges of Scientific Workflows, National Science Foundation. Arlington, VA. http://www.isi.edu/nsf-workflows06, May 1-2, 2006.

[4] Y. Han, H. Geng, H. Li, J. Xiong, G. Li, B. Holtkamp, R. Gartmann, R. Wagner, N. Weissenberg, VINCA - A Visual and Personalized Business-Level Composition Language for Chaining Web-Based Services, In The First International Conference on Service-Oriented Computing (ICSOC2003), Trento, Italy, 2003, pp.165-177.

[5] J. Fang, Service Virtualization for End-User Programming, PhD. Thesis, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 2006

[6] J.W. Wang, J. Yu, Y Han, A Service Modeling Approach with Business-Level Reusability and Extensibility, In: Proceedings of IEEE International Workshop on Service-Oriented System Engineering, Beijing, 2005.

[7] T. Andrews,F. Curbera,H. Dholakia,Y. Goland,J. Klein, etc., Business Process Execution Language for Web Services Version 1.1, BEA Systems, IBM Corporation, Microsoft Corporation, SAP AG, and Siebel Systems (2002), developerWorks (updated February 1, 2005), http://www.ibm.com/developerworks/library/specification/ws-bpel.

[8] A. Anjomshoaa, F. Brisard, M. Drescher, D. Fellows, A. Ly, S.McGough, D. Pulsipher, A. Sawa (Ed). Job Submission Description Language (JSDL) Specification, Version 1.0. 15 June 2005.

[9] P. Bajcsy, R. Kooper, L. Marini, B. Minsker, J. Myers. A meta-workflow cyber-infrastructure system designed for environmental observatories, Technical Report ISDAO1-2005, NCSA Cyber-environments Division, December 30, 2005.

[10] K.D. Swenson,S. Pradhan,M.D. Gilger, Wf-XML 2.0 XML Based Protocol for Run-Time Integration of Process Engines, WfXML20-200410c, Workflow Management Coalition, Draft October 8, 2004, http://www.wfmc.org/standards/wfxml.htm.

IEEE COMPUTER SOCIETY