A context- and role-driven scientific workflow development pattern

Wanchun Dou1,∗,†, Jinjun Chen2, Shaokun Fan1,3 and S. C. Chueng4

1State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China
2Centre for Information Technology Research, Faculty of Information and Communication Technologies, Swinburne University of Technology, P.O. Box 218, Hawthorn, Melbourne 3122, Australia
3MIS Department, University of Arizona, Tucson, AZ 85721, U.S.A.
4Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China

SUMMARY

Scientific workflow execution often demands data-centric and computation-intensive collaboration efforts, which is typically different from the process-centric workflow execution with fixed execution specifications. Scientific workflow execution often challenges the traditional workflow development strategy in dynamic context management and role definition. In view of this observation, application context spectrums are firstly distinguished from different profiles of scientific workflow development. Then, a role enactment strategy is proposed for enabling workflow execution in certain application context. They jointly enhance the validity of a scientific workflow development through clearly articulating the correlation between computational subjects and computational objects engaged in scientific workflow system. Furthermore, a novel context- and role-driven scientific workflow development pattern is proposed for enacting a scientific workflow system on the Grid. Finally, a case study is presented to demonstrate the generic natures of the methods in this paper. Copyright © 2008 John Wiley & Sons, Ltd.

∗Correspondence to: Wanchun Dou, State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China.
†E-mail: douwc@nju.edu.cn

Contract/grant sponsor: National Science Foundation of China; contract/grant numbers: 60721002, 60673017, 60736015
Contract/grant sponsor: National Grand Fundamental Research 973 Program of China; contract/grant number: 2002CB312002
Contract/grant sponsor: 863 Program of China; contract/grant number: 2007AA01Z178
Contract/grant sponsor: Jiangsu Provincial NSF Project; contract/grant number: BK2007137
Contract/grant sponsor: Program for New Century Excellent Talents in University; contract/grant number: NCET-06-0440
Contract/grant sponsor: Swinburne Dean’s Collaborative Grant Schema 2007–2008
Contract/grant sponsor: Hong Kong RGC/CERG; contract/grant number: HKUST6167/04E

Copyright © 2008 John Wiley & Sons, Ltd.
1. INTRODUCTION

In the past decades, workflow technology has drawn an enormous amount of attentions in various research and development communities. Many of the research interests and the mainstream of workflow products have been focused on business workflows to gain and retain competitive advantages in a competitive business arena [1,2]. With the development of complex scientific applications and experiments (e.g. climate modeling, earthquake engineering, structural biology and chemistry, federated computing for high-energy physics, etc.), a new special type of workflow application, i.e. scientific workflow system, gains more and more attractions in both research and development domains [3,4]. It is often featured by data-centric or computation-intensive execution with great transaction intensity [4]. Besides, scientific workflow execution is often enabled by a large number of activities in a domain-specific environment, especially with the development of Grid technology [5,6]. The data-centric or computation-intensive workflow execution is often orchestrated by distributed participant teams/groups and computational resources spanning domain-specific networks. In this situation, if an application context could not be definitely distinguished from the global workflow execution, the mixed application context would degrade the pertinence of collaboration among the participant teams/groups. Therefore, it often demands the participants to be aware of its application context of their local activities at runtime [7,8].

An application context specifies a special field for certain activity and collaboration goals (e.g. to collaborate with different participants or to access different resources) [9–12]. It makes a situation unique and comprehensible for enacting a certain activity or collaboration. A collection of tasks, agents and computing resources are the typical entities engaged in a context, which are often enabled by certain role enactment policy. Once a participant is aware of his/her context engaged in their certain activity and collaboration, he/she would effectively take part in the workflow execution in a suitable role. Furthermore, a participant could belong to multiple application contexts at the runtime [5,7]. Therefore, a participant could play more than one role in different application contexts for achieving different activity goals. Accordingly, a suitable workflow enactment policy could promote scientific workflow development with effective task assignment and resource usage management [2,8,13].

In view of these observations, a novel context- and role-driven scientific workflow development pattern is proposed in this paper, through application context analysis with certain role enactment policy. The main contribution of this paper is twofold. First, taking advantage of the object-oriented decomposition technique, application context spectrums and their role enactment policy are proposed by articulating the correlation between computational subjects and computational objects engaged in a scientific workflow system. Second, a scientific workflow development pattern is presented for enacting a scientific workflow system on Grid.

The remainder of this paper is organized as follows. In Section 2, a motivating example is presented to highlight some interesting topics that will be explored in this paper. In Section 3, application context spectrums are proposed from different profiles of scientific workflow development.
In Section 4, a role enactment strategy is proposed for enabling workflow execution in certain application context. In Section 5, a context- and role-driven workflow development pattern is put forward based on the context analysis and the role enactment policy. In Section 6, a case study is presented to demonstrate the generic natures of the methods presented in this paper. In Section 7, related works and comparison analysis are elaborated. Finally, the conclusion is presented in Section 8.

2. A MOTIVATING EXAMPLE

In this section, a concrete scientific workflow enactment is presented for highlighting the core research topic presented in this paper. In our life, timely weather forecast is often required, especially for a great meeting. For example, the Organizing Committee of an Olympic Game or a FIFA World Cup is always demanded to provide timely weather forecast service during the meeting. In this case, the weather information is often deduced based on timely weather observations (e.g. nephogram observing and record analysis). Moreover, the participant players may demand further weather information of the match site for preparing their matches well before the meeting, e.g. the average temperature, the average humidity, and the average ultraviolet ray intensity of the match site. This kind of weather information is often calculated based on the historical weather records accumulated in the past years. In addition, the weather forecast should also cover the other countrywide areas for the tourists during the meeting.

Suppose that the weather forecast service is co-performed by seven atmosphere research groups that are located at different sites with the typical geographical partitions of the country. For brevity and without the loss of generality, the labs are indicated by Lab-$i$, $i = 1, 2, \ldots, 7$. The labs share a super-computer that is installed with a sophisticated simulation and analysis program for timely processing the weather records. The weather records consist of the history weather records accumulated through the past years and the timely observing records. The observing records are, respectively, achieved from 25 countrywide observation stations. Here, the observation stations are indicated by ObservationStation-$i$, $i = 1, 2, \ldots, 25$. Note that different labs could only access to the observation stations granted to it. For example, Lab-1 could access to ObservationStation-1, ObservationStation-2, and ObservationStation-3, whereas Lab-2 just could access to ObservationStation-3, ObservationStation-4, and ObservationStation-5. Here, the super-computer is located at a computing center that is collocated with Lab-1. Other labs could share the computing resource (i.e. the super-computer and the simulation and analysis program) in the form of long-distance resource access based on certain usage policy. Once a lab exclusively occupies the super-computer, it would invoke the distant simulation and analysis program, by dispatching the required records to the program from the distant observation stations it could access. Figure 1 demonstrates the scenarios presented in this example.

The numbers illustrated in Figure 1 indicate a round of task execution. It mainly focuses on depicting a workflow execution in terms of where, what, who, when, and other application requirements. Firstly, to achieve a certain computing goal, a lab of Lab-$i$ would invoke some record assets stored in certain observation stations as indicated by ‘1’. Then, Lab-$i$ would apply for its usage permission to occupy the super-computer and the simulation and analysis program. The valid usage permission is often granted in the form of certificate or license. The applying process and

granting process are, respectively, indicated by ‘2’ and ‘4’. Once Lab-\(i\) occupied the super-computer, some records from selected observation stations would be dispatched to the computing center for calculating through the simulation and analysis program. The data dispatching process is indicated by ‘3’. In an ideal situation, the computing result could be achieved and returned timely to Lab-\(i\). The result dispatching process is indicated by ‘4’. During this computing process, if another lab of Lab-\(j\) wants to share the super-computer, its application would be rejected for the super-computer is exclusively occupied by Lab-\(i\). Lab-\(j\) would wait for some time and apply for its usage right again. Moreover, if Lab-\(j\) wants to share the computing result, it should apply for the granting of Lab-\(i\). These application processes are indicated by ‘5’ and ‘6’.

Figure 1 demonstrates a typical scientific workflow execution process. Here, some interesting topics are listed as follows for highlighting the motivations of our research work presented in this paper:

(1) Participant teams or members engaged in this scientific workflow execution should be timely aware of their local application situations and their role specifications for their activity and collaboration goals. This topic would be explored in Sections 3 and 4 in this paper.

(2) As there may be different role specifications for a participant to invoke required resource or to collaborate with other participants, a coordinating mechanism should be taken into consideration in scientific workflow execution. This topic would be explored in Section 5 in this paper.

3. APPLICATION CONTEXT SPECTRUMS ENGAGED IN SCIENTIFIC WORKFLOW DEVELOPMENT

Scientific workflow execution is essentially performed by a group of participants around a Project with data-centric and computation-intensive processing efforts. It demands that performance concerns should be imposed on the objects engaged in scientific workflow development and execution. Here, two concepts of Group and Member are used to stand for the execution subjects of a scientific workflow, i.e. executants. In application, a Project could be decomposed into a group of Tasks and is often implemented by some Group and Member in the form of task execution. Project and
Task stand for the executive objects, i.e. manipulated objects. Group could be materialized by an organization or a group (e.g. Lab-i as indicated in Section 2). It consists of a set of Members and a Member could be either an individual participant or a participant team. Additionally, a Task is often enacted by an individual Member or some Members. A Project is often enacted by some Group. Therefore, the concept of Task is parallel to the concept of Member (i.e. Task ↔ Member), while the Project is parallel to the Group (i.e. Project ↔ Group) in different application levels. Taking advantage of this design rationale, a scientific workflow system could be defined as follows.

Definition 1. A scientific workflow system could be formalized by \(((Project, Task), (Project, Group), (Group, Member), (Member, Task), CResource)\), in which CResource stands for the computational resource engaged in scientific workflow execution.

Form different profiles of the scientific workflow system, the application context spectrums engaged in a scientific workflow development are classified into six styles: (1) context of \((Project, Task)\), (2) context of \((Project, Group)\), (3) context of \((Group, Member)\), (4) context of \((Member, Task)\), (5) context of \((Project/Task, CResource)\), and (6) context of \((Group/Member, CResource)\). Their application logics are depicted as follows.

(1) From the Project–Task perspective, a Project is often decomposed into Tasks among which certain parameter-dependent or knowledge-dependency relations are specified. The context of \((Project, Task)\) typically consists of a Project, a set of Tasks derived from the Project, and the nested or parallel task decomposition logic between them.

(2) From the Project–Group perspective, a Project is often carried out by Groups, which make up of a virtual organization when the Project is carried out in an organizational-across environment. Here, the context of \((Project, Group)\) typically consists of a Project, a set of Groups recruited in Project execution, and the Task assignment schema among the Groups.

(3) Form the Group–Member perspective, a Member often takes on a concrete Task according to his/her knowledge background and experience. Task assignment is often enacted by a Group at the application level of Group–Member. Here, the context of \((Group, Member)\) typically consists of a Group, a set of Members recruited by the Group, and an organization policy enacted inside a Group.

(4) From the Member–Task perspective, a Task is often allocated to a certain Member by a Group. Here, the context of \((Member, Task)\) typically consists of a set of Members, a set of Tasks, and the collaboration relation among the Members based on the correlation of the Tasks derived from the context of \((Project, Task)\).

(5) From the Project/Task–CResource perspective, a Project is often implemented by consuming certain CResource. Here, the context of \((Project/Task, CResource)\) typically consists of a Project, a set of Tasks, a set of CResource assets, a configuration policy between the Tasks and the CResource assets, and a set of permission granting strategy.

(6) From the Group/Member–CResource perspective, the role-based usage policy is often posed on Groups or Members for their CResource accesses. Here, the context of \((Group/Member, CResource)\) typically consists of a set of Groups and their Members, a set of CResource assets, and a usage policy for specifying the accessing permission for different Groups and their Members.
In this paper, the Member/Group would be treated as the computational subjects, while the Project, Task, CResource would be treated as the computational objects. In scientific workflow execution, a computing behavior is often performed by a computational subject for achieving certain computational objects. If a scientific workflow management system (SWMS) could mask the unrelated context information, a scientist could concentrate on his/her domain-specific research (workflow fragment) at an abstract level, and a scientific workflow execution could be orchestrated by the workflow fragments with timely context switching. It is often promoted by certain role enactment policy in certain context.

4. A ROLE ENACTMENT STRATEGY FOR WORKFLOW EXECUTION IN CERTAIN APPLICATION CONTEXT

In practice, there are various jobs engaged in scientific workflow executions, and a participant should be assigned a role based on his/her responsibilities and qualifications in different application contexts. The role-based policy aims at enhancing the adaptability of a participant for scientific workflow execution. Here, a role could be treated as a semantic construct around which the access control policy is formulated [2,13]. It not only represents the competency to do some specific tasks but also embodies authority and responsibility (e.g. programmer, project supervisor, administrator, or item consultant). In practice (e.g. Grid application), the terms of authorization, access right or privilege are often used to indicate an approval for accessing to one or more objects. The holder of the permission is conferred the ability to perform some actions in a positive way. In our research, access control policies are brought into effect through two stages: (1) to specify the relationships between roles and permissions and (2) then to assign proper roles to certain participants engaged in scientific workflows. As application logic is often enabled by an abstract role specification without taking care of concrete actors, it greatly simplifies the management of environment access in a secure and valid way. Furthermore, a role can be granted with new permissions during its execution, as a new application or system is incorporated [9,13]. It greatly enhances the adaptability of a participant at runtime [12].

In this paper, four kinds of role specification would be proposed based on the function analysis of a computational subject. They are (Project, Group; Role 1), (Group, Member; Role 2), (Member, Task; Role 3), and (Group/Member, CResource, Role 4), which are, respectively, enabled in application context of (Project, Group), (Group, Member), (Member, Task), and (Group/Member, CResource) as demonstrated in Figure 2. Role 1 is enacted among the participant organizations/groups for carrying out a Project. Role 2 is enacted among the Members inside an organization or a group for working status identification. Role 3 is enacted among the Tasks and Members for task assignment or task execution. Role 4 is enacted among the Group/Member for CResource access with certain usage policy.

By explicitly defining the roles and specifying the permissions imposed on the roles, role-based application process could be controlled pertinently at runtime. Here, a role’s adaptability could be formalized as follows: (1) $\forall$ group (group ∈ Group $\rightarrow$ $\exists$ role (role ∈ Role1 $\lor$ role ∈ Role 4)) and (2) $\forall$ member (member ∈ Member $\rightarrow$ $\exists$ role (role ∈ Role 2 $\lor$ (role ∈ Role 3) $\lor$ (role ∈ Role 4))).

Bringing the role definitions into its development context, Figure 2 articulates the correlations.
among these entities in the form of UML based on the role-driven enactment strategy in different application contexts.

In Figure 2, the association between \textit{CResource} and \textit{Project} is an aggregation, whereas the relationship between \textit{Task} and \textit{Project} is a composition, to highlight their different combinatorial relationships. Here, some further analyses are presented for role enactment and task assignment.

(1) In practice, there are many roles such as ItemSupervisor, Programmer, Test Engineer, Item Consultant, Analyst Engineer, etc. in a project development. Each function team of member often enacts as one of these roles. Role 1, Role 2, and Role 3 are just specified based on the role taxonomy and permission specification.

(2) In practice, \textit{isA\{\ldots\}} and \textit{hasA\{\ldots\}} are two typical task decomposition logics. \textit{isA\{\ldots\}} is often employed to indicate a task’s parent task or its pre-task. \textit{hasA\{\ldots\}} is often employed to indicate a task’s child tasks or its post-tasks. For example, a task of Task-3 would be specified by \textit{Task-3 = isA\{Task-1\} ∧ hasA\{Task-5, Task-6\}}, in which Task-1 is its parent task, and Task-5 and Task-6 are its two child tasks. In practice, task decomposition logics also navigate the interactions of process fragments in scientific workflow execution.

(3) Generally, the resource sharing process engaged in scientific workflow is often controlled rigorously. The resource usage policy should elaborate on how the resource usage takes place in terms of where, what, who, and when it is allowed [7,9,13]. Different resource users should be discriminated according to usage permission. Role 4’s operation style (e.g. read, write, read&write, etc.), spectrum (e.g. SuperUser, CommonUser, LimitedUser, TemporaryUser, to name a few), resource sharing manner (e.g. exclusive or not), and lifetime are some typical issues related to Role 4’s specification.

\section{5. A CONTEXT- AND ROLE-DRIVEN SCIENTIFIC WORKFLOW DEVELOPMENT PATTERN}

In this paper, the \textit{Project, Task, Group, Member, CResource,} and role engaged in a scientific workflow system would be depicted by six classes for their development in the form of UML scheme as demonstrated in Figure 3.
Here, the CResource engaged in scientific workflow execution is classified into task/process CResource (T/PCResource) and infrastructure/generic CResource (I/GCResource) for highlighting different resource usage policies. T/PCResource could be just invoked by a role of a special computational subject in a lifecycle of permission (e.g. recode assets stored in the observation station, super-computer, and the simulation and analysis program as mentioned in Section 2). However, I/GCResource stands for the public resource that opens for all the members engaged in the scientific workflow execution (e.g. the public resource engaged in the context of (Project, Task) and context of (Project/Task, CResource)). Table I specifies the typical attributes and functions contained in these classes. Table II illustrates the development policies and disciplines recruited for promoting the development process.

As tasks are not always 'workflow aware', a workflow engine should provide a context navigating and switching mechanism for workflow execution. For example, in the workflow execution paradigm advocated by WfMC, a workflow engine is recruited to instantiate a workflow specification, decompose it into tasks or activities, and then allocate them to processing entities for execution. This approach is often unfolded around process definition and process instance, and is often referred to as a scheduler-based paradigm [14]. In this scheduler-based paradigm, the transitions among the start, inactive, active, suspended, and other states are often promoted by a workflow engine. Accordingly, the workflow execution just acts as a state transition machine navigated by a workflow engine [1,15]. In our scientific workflow development pattern, piloted by the context-and role-driven scenarios, the workflow execution could be formalized as follows.
Table I. Specifications of the typical attribute and function contained in the classes as demonstrated in Figure 3.

<table>
<thead>
<tr>
<th>Items</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project.ProjectSpecification</td>
<td>Attribute to depict the Project from motivations, problem specifications, resource requirements, expected goals, schedule, etc. It is often associated with a descriptive document or ontology system</td>
</tr>
<tr>
<td>Project.EngagedGroup</td>
<td>Attribute to indicate the Groups engaged in Project’s implementation</td>
</tr>
<tr>
<td>Project.TaskDecompose()</td>
<td>Function to specify the decomposition logic for partitioning a global Project into Tasks, which aims at specifying Project’s internal problem logic in the form of Task external implementation logics</td>
</tr>
<tr>
<td>Project.Role1Specify()</td>
<td>Function to specify the permissions related to Role 1 and assign a role for each Group engaged in Project implementation. For example, Group-i would be arranged to programming, Group-j would be assigned to program testing, whereas team-m would be appointed to document writing, during software development</td>
</tr>
<tr>
<td>Project.resourceConfigure()</td>
<td>Function to specify CResource configuration for scientific problem solving from T/PCResource configuration and I/GCResource configuration aspect</td>
</tr>
<tr>
<td>Project.OntologySpecify()</td>
<td>Function to define all the ontologies recruited in Project specifications, such as knowledge ontology, method ontology, problem ontology, etc.</td>
</tr>
<tr>
<td>Group.EngagedMembers</td>
<td>Attribute to indicate the Members included in a Group</td>
</tr>
<tr>
<td>Group.TaskAllocate()</td>
<td>Function to allocate Tasks to each Member or team included in a Group</td>
</tr>
<tr>
<td>Group.Role2Specify()</td>
<td>Function to specify the permissions related to Role 2 and to assign a role for each Member included in a Group based on commander–subordinate relationships. For example, Member-i would be assigned as a team leader, Member-j would be appointed as a programmer with certain Tasks</td>
</tr>
<tr>
<td>Group.CResourceApply()</td>
<td>Function to apply for CResource access during Project’s implementation. It assists Project.resourceConfigure() to perfect CResource access when some unexpected CResources are required during Task’s implementation in a dynamical environment. Note that the authorized CResource achieved through this function is often shared by each of its Members, which is different from the function of Member.CResourceApply() as listed in this table</td>
</tr>
<tr>
<td>Group.ContextAware()</td>
<td>Function to identify the current application environment or the infrastructure, in which a Group is enacting, based on data collection and analysis for CResource access or collaboration. Project.Role1Specify(), sensor network devices, and some intelligent middlewares would be helpful to facilitate context awareness</td>
</tr>
<tr>
<td>Group.RoleEnact()</td>
<td>Function to implement its Tasks or access CResource acting in an appropriate role based on Group.ContextAware() and Project.Role1Specify()</td>
</tr>
<tr>
<td>Member.WebLocation</td>
<td>Attribute to indicate the Member’s Web location for facilitating interaction in Web environment</td>
</tr>
<tr>
<td>Member.TeamItBelongTo</td>
<td>Attribute to indicate the Group the Member belongs to</td>
</tr>
<tr>
<td>Member.AssignedTasks</td>
<td>Attribute to indicate the Tasks assigned to the Member</td>
</tr>
<tr>
<td>Member.CResourceApply()</td>
<td>Function to apply for CResource access during task implementation. It assists Group.CResourceApply() to perfect CResource access when some unexpected CResources are required in a dynamical environment. Note that the authorized CResource through this function is consumed individually by its applicant, which is different from the function of Group.CResourceApply()</td>
</tr>
</tbody>
</table>
Table I. Continued.

- **Member.ContextAware()**: Function to identify the current environment or the infrastructure in which a **Member** is enacting. Some sensor network devices and some intelligent middlewares would be helpful to facilitate its implementation.

- **Member.RoleEnact()**: Function to implement its **Tasks** or access **CResource** acting in an appropriate role based on **Member.ContextAware()**, **Group.Role2Specify()**, and **Task.Role3Specify()**.

- **Task.TaskSpecification**: Attribute to indicate pre-conditions, pre-tasks, post-tasks, schedule, expected goals, implementation **Members**, **CResource** configurations, and other specification issues related to **Task** implementation in the form of descriptive document, with the help of the function **Project.OntologySpecify()**.

- **Task.T/PCResourceConfigure()**: Function to specify the **T/PCResource** configuration, in the static perspective, among members engaged in a **Task’s** implementation to facilitate their **CResource** access.

- **Task.Role3Specify()**: Function to specify the role definition based on **isA** logic and **hasA** logic, according to the logic among the **Tasks** derived from global **Project’s** decomposition logic.

- **CResource.WebLocation**: Attribute to indicate the Web location of a **CResource** for facilitating **CResource** access in a Web environment.

- **CResource.Specification**: Attribute to specify the attribute of a **CResource**.

- **CResource.ItsOwner**: Attribute to indicate the owner of a **CResource**.

- **CResource.Authenticate()**: Function to validate a certificate, when a **CResource** access occurs, to guarantee that the consumer is a legal candidate.

- **CResource.Authorize()**: Function to grant a **CResource** access to certain consumers through certificate delivering.

- **CResource.Role4Specify()**: Function to specify the access permissions or usage policy to pilot the **CResource** access in the form of Role 4 as analyzed in Case 3.

- **Role.Style**: Attribute to indicate the style of role such as Role 1, Role 2, Role 3, or Role 4.

- **Role.LifeTime**: Attribute to specify the lifetime of a role after it is initiated.

- **Role 1.PermissionSpecify()**: Function to specify the permissions related to Role 1’s behavior in **Group** organizational context environment.

---

**Definition 2.** A scientific workflow execution could be promoted by a multi-relation, i.e. Workflow \( \text{Exec} = ((\text{Task} \times \text{Member}) \times \text{Role}) \), in which, **Task** ∈ **Project**, **Member** ∈ **Group**, **Role** = \{Role 1, Role 2, Role 3, Role 4\).

It provides a workflow engine development pattern, i.e. to assign a role to a participant in a certain application context initiated by a task. It is essentially navigated by effective context switching at runtime. More specifically, a **Member** judges his/her task execution situation, apperceives the application context for the task goals, and then enters into a concrete application context with a role specification. The taxonomy of the context and role presented in this paper provides the guideline, the context awareness and role-binding foundation for promoting a scientific workflow execution. Note that the workflow execution as defined by Definition 2 is jointly navigated by two application logics with certain state transitions [16]. One is navigated by goal-aware logic around...
Table II. Development policies and implementation methods in scientific workflow development.

<table>
<thead>
<tr>
<th>Context spectrum</th>
<th>Development policy</th>
<th>Implementation prolocutors</th>
</tr>
</thead>
<tbody>
<tr>
<td>In context of (Project, Task)</td>
<td>Object-oriented task decomposition policy</td>
<td>Project.ProjectSpecification ∧ Project.TaskDecompose() ∧ Project.OntologySpecify()</td>
</tr>
<tr>
<td>In context of (Project, Group)</td>
<td>Role 1-driven organization and collaboration policy enacted in a domain-spanning environment</td>
<td>Role 1, i.e. Project.ProjectSpecification ∧ Project.EngagedGroup ∧ Project.Role1Specify() ∧ Group.ContextAware() ∧ Group.RoleEnact() ∧ Role 1.PermissionSpecify()</td>
</tr>
<tr>
<td>In context of (Group, Member)</td>
<td>Role 2-driven process organization and collaboration policy enacted inside an organizational</td>
<td>Role 2, i.e. Group.EngagedMember ∧ Group.Role2Specify() ∧ Member.TeamItBelongTo ∧ Member.ContextAware() ∧ Member.RoleEnact() ∧ Role 2.PermissionSpecify()</td>
</tr>
<tr>
<td>In context of (Member, Task)</td>
<td>Role 3-driven Task assignment and execution policy</td>
<td>Role 3, i.e. Group.TaskAllocate() ∧ Member.AssignedTasks ∧ Task.Role3Specify() ∧ Role 3.PermissionSpecify()</td>
</tr>
<tr>
<td>In context of (Project/Task, CResource)</td>
<td>Project- and task-oriented resource configuration</td>
<td>Project.ProjectSpecification ∧ Project.CRresourceConfigure() ∧ Task.CRresourceConfigure()</td>
</tr>
</tbody>
</table>

In this paper, the scientific workflow system presented in Section 2 is originated from a concrete application of timely weather forecast service. In the following section, a typical collaborative business workflow execution will be proposed to demonstrate the generic natures of the methods investigated in this paper.

6. A CASE STUDY

Figure 4 demonstrates the cross-organizational workflow execution based on a supply chain with an e-service scenario in a Web-based environment. The application background has been discussed in [17], in which it was exploited to demonstrate the technique of workflow view in a positive way.
cross-organizational Web service environment. In this section, it is specifically recruited to demonstrate the methods presented in this paper for refining the workflow view development. There are three types of organizations involved, viz., end-user, system integrators, and parts vendor. The end-user goes through an acquisition workflow, say, for an advanced server system. A system integrator’s workflow starts when a quotation request is received. A parts vendor’s workflow starts when a quotation request is received. Assuming this is the end of the supply chain, the vendor has all the necessary information to reply to the system integrator with up-to-date parts information and prices. If B2B orders on standard parts are performed together with the payment, this workflow ends after the delivery of the ordered parts. The interested readers can refer to [17] for more details.

In Figure 4, four basic roles of financial department, manufacturing department, sale department, and logistic department (e.g. warehouse management, etc.) should be distinguished from different workflow executions. Here, it is crucial to guarantee the collaboration consistency of the process fragments engaged in workflow execution, no matter for an individual workflow execution or for the organization–across workflow interactions. For example, for the parts vendor’s workflow execution,
the delivery process should not be enabled if the payment has not been received from the system integrator’s workflow execution. Similarly, for the system integrator’s workflow execution, the process of ‘PrepareQuotation’ should start after the information of the missed parts is confirmed from the parts vendor’s workflow execution. The collaboration among the departments is often enabled based on certain collaboration context logic and certain role definitions. Even in the end-user’s acquisition workflow execution, the consistent clerical processing routines are always required for achieving the purchase goal. It is a typical context switch process enabled by certain cross-organizational collaborative logic, which reflects the accuracy, reliability, security, privacy, and short processing performance requirements for workflow execution.

As discussed in [17], Figure 5(a) (corresponding to the Figure 3 in [17]) demonstrates a workflow view of the system integrator presented to the end-user based on a global analysis technique. However, as mentioned in [18], such a global analysis technique often mandates every workflow execution unveiling all the individual tasks. It is infeasible when some business parties refuse to disclose their process details for privacy or business reasons. For example, the service execution processes as demonstrated in Figure 5(a) are often tough upon the system integrator’s private technique. A system integrator often refuses to disclose the details to gain and retain business advantages in a competitive business arena. Accordingly, the service execution process should be masked in workflow view development to protect the system integrator’s business privacy. Figure 5(b) demonstrates the refinement of the previous workflow view that first appeared in [17] by considering certain business privacy. Note that the privacy protecting is often a role-based issue. For some collaborators, it is not mandatory to consider the issue of privacy protecting, whereas for others it is forbidden to disclose its business privacy [16]. For example, the system integrator could disclose its service execution processes to some reliable and long-term end-users to show its product quality or to enhance their mutual credits, even with some supplementary details of the parts for some possible update quotations caused by fluctuating prices.

In this situation, two kinds of issues should be considered in workflow view development. One is to whom (i.e. certain role engaged in collaboration) the certain process fragment should be masked or not. The other is what process fragment should be masked (i.e. certain application context information). If we could not distinguish the information from a global workflow view, the workflow view would disclose some private business information. It would degrade the application of the workflow view scenario. Taking advantage of the role and context analysis techniques, the workflow view scenario could be exploited in a general way to satisfy different levels of interaction requirements and collaboration. It is a typical context- and role-driven workflow execution pattern. Here, the ‘service execution’ stands for a task, whose internal application context consists of the sub-tasks of ‘order parts’, ‘assemble system’, ‘install software’, and ‘system testing’, as well as the interaction logics among them. It indicates the internal application context related to certain business process. Once the internal application contexts are definitely distinguished from a global workflow execution, a workflow view could be developed based on certain privacy protecting policy.

Taking advantage of the scenarios presented in this paper, a general policy for workflow view development could be depicted by Definition 2.

**Definition 3.** Let WorkflowExecution = (V, E) be a directed graph in which V is a set of tasks and edges in E are defined on V × V for any task ∈ V. Accordingly, a workflow view could be treated as a sub-graph of WorkflowExecution.
Figure 5. Workflow view of a system integrator towards (a) an end-user based on global analysis technique and (b) an end-user based on certain privacy policy.
Here, we believe that the application context could be specified by the edges in $E$, and the workflow view is often specified for certain role. Therefore, taking advantage of Definition 3, a general policy for workflow view development associated with certain role could be specified by the following logic representation: $\exists \text{role} \in \text{Role} \rightarrow (\exists \text{WorkflowView} \subseteq \text{WorkflowExecution})$. For example, corresponding to Figure 5, let $\text{WorkflowExecution} = (V, E)$ stands for system integrator’s workflow execution, the workflow view toward an end-user as demonstrated in Figure 5(a) could be indicated by $\text{WorkflowView} = (V_{wv1}, E_{wv1})$, in which $V_{wv1} = \text{sub-Task2}$ and $E_{wv1} = E$; however, for Figure 5(b), the workflow view would be refined by $\text{WorkflowView} = (V_{wv2}, E_{wv2})$, in which $V_{wv2} \subset \text{sub-Task2}$ and $E_{wv2} \subset E$. More specifically, the cross-organizational workflow system as demonstrated in Figure 4 could be formalized as follows by taking advantage of these methods for refining workflow view development.

1. $\text{Project} = \text{‘To purchase an advanced server system based on a supply chain in a web-based environment’}$
2. $\text{Group} = \{\text{End-User, System Integrator, Parts Vendor}\}$
3. $\text{Member} = \{\ldots\}$
   3.1 $\text{EndUser}::\text{Member} = \{\text{User, FinancialDepartment}\}$
   3.2 $\text{SystemIntegrator}::\text{Member} = \{\text{FinancialDepartment, ManufacturingDepartment, SaleDepartment, Logistic Department}\}$
   3.3 $\text{PartsVendor}::\text{Member} = \{\text{FinancialDepartment, ManufacturingDepartment, SaleDepartment, Logistic Department}\}$
4. $\text{Task} = \{\text{sub-Task1, sub-Task2, sub-Task3}\}$
   4.1 $\text{sub-Task1} = \{\text{QuotationEnquiry}, \text{QuotationEvaluation}, \text{PurchaseOrder}, \text{Receive\&CheckSystem}, \text{PaymentAuthorization}\}$
   4.2 $\text{sub-Task2} = \{\text{CheckPartsInfo}, \text{PrepareQuotation}, \text{PrepareExtraInfo}, \text{Verify\&ConfirmOrder}, \text{ServicePreparation} = \{\text{OrderMissingParts}, \text{AssembleSystem}, \text{InstallSoftware}, \text{SystemTesting}, \text{Deliver\&Install}\}\}$
   4.3 $\text{sub-Task3} = \{\text{PartsQuotation}, \text{DeliverParts}\}$
5. $\text{CResource} = \{\ldots\}$

7. RELATED WORKS AND COMPARISON ANALYSIS

Generally, the development of SWMS is complex and highly interdisciplinary. In [19], three typical design techniques are presented: (1) by extending existing mature workflow models and engines; (2) to choose proper middle-wares to couple SWMS resources; and (3) to construct a new SWMS by using the prevalent software engineering technologies. In this paper, we focus on exploring a novel context- and role-driven scientific workflow development pattern. Generally, role-driven technique is a prevalent methodology for domain-across application. Honda et al. [9] gave an example of role specification in the object-oriented modeling and programming language domain. Fowler [10] gave an example of personnel roles in a company to be assumed by employees (e.g. engineer, salesmen, director, and accountant). Kendall [11] gave an example of the bureaucracy pattern. There are five roles in the pattern: director, manager, subordinate, clerk, and client. Tamai et al. [12] introduced an adaptive object model with dynamic role binding.

In these related research works, the context environments are treated as fields of collaborations between roles and a role is the first class construct at runtime as well as at model description time. An object adapts to the environment through assuming one of the roles. The encapsulated collaboration
is an independent reuse component to be deployed separately from objects that participate in them. An object can freely enter or leave an environment and could be belong to multiple environments at a time, so that dynamic adaptation or evolution of an object could be realized easily.

Generally, we share similar techniques with these researches. However, these related works [9–12] centralize on role-based model or role-based language design. Their application domains are definitely different from the scientific workflow domain. The context- and role-driven scientific workflow development pattern presented in this paper mainly focuses on guaranteeing the pertinence property in scientific workflow deployment. It is achieved by definitely assigning a computational object to a suitable computational subject in a definite context with certain effective role-binding policy. More specifically, it is achieved through the following steps: (1) a context is definitely specified for discriminating the computational objects and the computational subjects from different execution profiles and (2) a participant would be tailored to a range of application situations with a role specification between a computational subject and the computational object. This development guarantees that the later workflow execution could be effectively enacted by assigning definite computational subjects to suitable computational objects for their data-centric and computational-intensive operations.

The development policies as presented in [3,4,15] are derived from real-life application, whereas the pattern presented in this paper aims at promoting scientific workflow development from software engineering aspect. Compared with the development policies as summarized in [3,4], our research would directly provide some useful software templates to benefit a real-life scientific workflow development. This advantage makes the proposal presented in this paper universal. It is especially suitable for cross-organizational, long-running, loosely coupled, and domain-specific workflow application domain as demonstrated in the motivating example and the case study.

8. CONCLUSIONS

In this paper, we mainly focus on exploring some key issues engaged in scientific workflow development. Firstly, the application context spectrums are distinguished from different profiles of scientific workflow development. Furthermore, a role enactment strategy is proposed for enabling future workflow execution in certain application context. According to the taxonomy of the context and role specification, a context- and role-driven scientific workflow development pattern is proposed, and a context and role-switching mechanism is investigated for promoting a workflow engine development. At last, a case study is proposed to demonstrate the generic natures of the methods presented in this paper. To refine this workflow pattern for more real-life and benchmark scientific applications will be our main research work in the future.

ACKNOWLEDGEMENTS

This paper is partly supported by the National Science Foundation of China under grant numbers 60721002, 60673017 and 60736015, National Grand Fundamental Research 973 Program of China under grant no. 2002CB312002, 863 Program of China under grant no. 2007AA01Z178, Jiangsu Provincial NSF Project under grant no. BK2007137, Program for New Century Excellent Talents in University under grant NCET-06-0440, Swinburne Dean’s Collaborative Grant Schema 2007–2008. Hong Kong RGC/CERG grant HKUST6167/04E.
REFERENCES