A Workflow Engine-Driven SOA-Based Cooperative Computing Paradigm in Grid Environments
Wanchun Dou, Jinjun Chen, Jianxun Liu, S.C. Cheung, Guihai Chen and Shaokun Fan

International Journal of High Performance Computing Applications 2008; 22; 284
DOI: 10.1177/1094342007086227

The online version of this article can be found at:
http://hpc.sagepub.com/cgi/content/abstract/22/3/284

Published by:
SAGE Publications
http://www.sagepublications.com

Additional services and information for International Journal of High Performance Computing Applications can be found at:

Email Alerts: http://hpc.sagepub.com/cgi/alerts
Subscriptions: http://hpc.sagepub.com/subscriptions
Reprints: http://www.sagepub.com/journalsReprints.nav
Permissions: http://www.sagepub.com/journalsPermissions.nav

Citations (this article cites 19 articles hosted on the SAGE Journals Online and HighWire Press platforms):
http://hpc.sagepub.com/cgi/content/refs/22/3/284
A WORKFLOW ENGINE-DRIVEN
SOA-BASED COOPERATIVE
COMPUTING PARADIGM IN GRID
ENVIRONMENTS

Wanchun Dou1,2,3
Jinjun Chen4
Jianxun Liu5
S. C. Cheung3
Guihai Chen1,2
Shaokun Fan1,2

Abstract
The grid has been proposed as a promising service-orien-
ted platform for increasingly complex cooperative com-
puting. The platforms of service-oriented grids are often
Web-based where participants collaborate to achieve a
common goal by sharing scarce Web-Based Computational/
Computing Resources (WBCR). To share the WBCR effec-
tively is a challenging problem in boundary-spanning grid
environments, particularly when these resources are sub-
ject to both static and dynamic usage. To set up the certif-
icate-based usage policy described in this paper, we first
explore a workflow engine-driven SOA-based resource
access control mechanism. Then, aiming at setting up a
cooperative computing paradigm from the resource sharing
perspective, an infrastructure derived from a specific project
of SOA&EDSCE (SOA-Based&Engine-Driven Structured
Cooperative Computing Environment) is proposed for pro-
moting its cooperative computing in grid environment based
on the control disciplines and the WBCR usage policy. The
main contributions of this paper are twofold: 1) a workflow
engine-driven SOA-based WBCR sharing mechanism is
presented in accordance with to the certificate-based usage
policy; and 2) a specific infrastructure of cooperative com-
puting is put forward for the collaboration based on the
WBCR sharing mechanism.

Key words: workflow engine, service-oriented architecture
(SOA), cooperative computing, grid, certificate

1 Introduction
Advances in Internet technology have led to the evolution
of many applications. A representative example is cross-
organizational collaborative work (Holsapple and Luo
2003; Tsai et al. 2005). Typically, the traditional Compu-
ter Supported Cooperative Work (CSCW) focuses on
understanding how remote collaboration is carried out in a
dynamic way (CCF Project Team 2000). In practice, CSCW
applications can induce resource-intensive computations
or resource requirements beyond those originally allo-
cated. This has motivated the development of usage policy
and invocation control disciplines in runtime environ-
ments (Keahey, Ripeanu, and Doering 2003; Dumitrescu
et al. 2005). It also poses some challenges on how to con-
duct resource sharing, and cross-organizational collaboration
in a seamless manner regardless of geographical location
and heterogeneous infrastructure (Keahey, Ripeanu, and
Doering 2003; Dumitrescu et al. 2005). Similarly, to sat-
isfy application requirements, cooperative computing in
grid environments attracts increasing attention as an
emerging computational paradigm of CSCW (Foster and
Kesselman 2003; Holsapple and Luo 2003; Tsai et al.
2005). It is specifically proposed to meet the demand of e-
Science enactment, in which the collaboration typically
accesses multiple heterogeneous resources, spans autono-
mous administrative domains, and so on. Accordingly,
this new paradigm is particularly suitable for those sci-
tific applications that require fast access to a large quan-
tity of distributed computational resources (Bester et al.
1999; Abdelzaher, Shin, and Bhatti 2003; Foster and Kess-
selman 2003).

In an e-Scientific environment, resource sharing and
knowledge interactions are often performed jointly by col-
laborators at different sites (Zhao et al. 2004; Dumitrescu,
Raicu and Foster 2005). Some collaborative systems,
such as the CCF system (CCF Project Team, 2000), view
collaboration from four major aspects: application shar-
ing, data exchange and manipulation, computational transforms, and communication. Those four aspects are instantiated in various forms to cater for different collaboration-aware applications. Accordingly, the cooperative computing in grid environments discussed in this paper is characterized by large-scale resource sharing, cross-organizational applications, and in some cases, high-performance orientation. Here, the concept of resource typically refers to Web-based Computational/Computing Resources (WBCR). In grid environments, the WBCR may include the workstation, database, computing components, repository, networks, computing service, and so on. In practice, as a site of WBCR can be accessed simultaneously by several users, access service is the key to coordinating WBCR sharing; the coordination greatly depends on certain local access and usage policies (Abdelzaher, Shin, and Bhatti 2003; Dumitrescu et al. 2005). Furthermore, to coordinate the cooperative processes, the collaborations in grid environments are often integrated into a dynamic, multi-institutional virtual organization (VO), based on widespread coordinated use of networked WBCR (Foster and Kesselman 2003; Watson and Humphrey 2003; Dumitrescu and Foster 2004).

In light of the observation that the WBCR access control is a key technology and collaboration among participants is often formalized in the form of a workflow, we focus on exploring the workflow engine-driven SOA-based WBCR access and usage policy to promote cooperative computing in grid environments. More specifically, aiming at setting up a cooperative computing paradigm from the resource sharing perspective, the main contributions of this paper are twofold: 1) a workflow engine-driven WBCR access and usage mechanism is presented based on a certificate granting policy, and 2) a cooperative computing infrastructure is put forward based on the workflow engine-driven WBCR access and usage mechanism.

The remainder of the paper is organized as follows. In Section 2, a motivating example is presented for highlighting the key topics that will be discussed in this paper. In Section 3 related work is discussed. In Section 4, a workflow engine-driven and SOA-based resource access control mechanism is explored and some WBCR access control disciplines are presented for setting up a certificate-based resource usage policy. Furthermore, typical implementations of the mechanism are depicted and a case study for demonstrating its performance is presented in this section. In Section 5, a workflow engine-driven and SOA-based cooperative computing paradigm is explored in grid environments, based on a specific project of SOA-Based&Engine-Driven Structured Cooperative Computing Environment (SOA&EDSCCE). Further discussion is presented in Section 6 to highlight some further applications. Lastly, Section 7 concludes the paper and points to our future work.

2 A Motivating Example

In this section, a motivating example is presented for highlighting the topics explored in this paper, although it is contrived to some extent. We imagine that seven atmosphere research groups from different laboratories will collaborate to forecast the countrywide weather for the coming years. The seven laboratories are located at different sites countrywide. Here, the laboratories are indicated by Lab-\(i\), \(i = 1, 2, \ldots, 7\). To achieve the common purpose, they need the weather records accumulated through the past 50 years. The data assets of the records are stored in different nationwide observation stations that open their records to the laboratories during the computation. Furthermore, during the collaboration, a supercomputer is recruited by a global simulation and analysis program, and the program is often enabled by data input from the distributed observation stations or laboratories. In this example, the supercomputer is located in a computing center that is collocated with Lab-1, and the data assets exploited by the simulation and analysis program will be derived from 25 countrywide observation stations. In addition, some local computing programs are dispatched to the computing center from other laboratories to achieve rapid operations.

In this multi-institutional scientific collaboration, the supercomputer, the simulation and analysis program, and the data resources are typical WBCRs. To locate and access to the WBCRs efficiently, there is often an individual resource broker service maintained by the collaboration in order to invoke and balance the resource sharing for simulation and analysis. Here, the broker is located at the computing center. Figure 1 illustrates the

---

**Fig. 1** The sequence diagram in line with the WBCR invocation in scientific collaboration.
sequence diagram in conformance with the WBCR invocation engaged in scientific collaboration.

Some interesting topics relating to this example are summarized as follows to highlight the research issues presented in this paper:

1. The performance of the simulation and analysis program depends on the contributions from all distributed participants and WBCRs. The resource pool and the user populations are large. Furthermore, the user populations often access the resource pool in a dynamic way in collaboration, which poses some access control disciplines in practice.

2. Collaboration or a computation often consists of a group of dynamic processes running on different resources and sites. The processes may communicate using a variety of mechanisms, and the low-level WBCR sharing may be created and destroyed dynamically during program execution. A global resource coordination mechanism is often required in the collaboration.

3. An individual user can be associated with different local name spaces, credentials or accounts at run time for different WBCR accesses. More specifically, a WBCR may require different authentication and authorization mechanisms and usage policies for different users. The unexpected WBCR requirement often challenges the scheduled usage policy for WBCR sharing in collaboration.

In view of those observations, an efficient WBCR usage policy and a collaboration infrastructure should be set up for dynamic resource access in grid environments, based on certain WBCR access control disciplines. These are the main topics that will be explored in this paper.

3 Related Work

In practice, the cooperative computing processes in grid environments are often organized in the form of VO (Foster, Kesselman, and Tuecke 2001; Foster and Kesselman 2003). Generally, a VO consists of a group of participant organizations who seek to share certain WBCRs for a common purpose, which is essentially a cross-organization virtual alliance for integrating participant organizations’ core competences and resources. The alliance often aims at achieving organizational objectives through extending some of their organizational activities externally, based on some matured information technologies (e.g., corporations with an intensive use of telecommuting; Dumitrescu and Foster 2004). In practice, a VO is often characterized by flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and organizations. Other basic features of VO are described in the literature (Foster, Kesselman, and Tuecke 2001; Wasson and Humphrey 2003; Dumitrescu and Foster 2004). Those features can be summarized as: (1) the lifetime of a cooperation is limited; (2) there is access to a wide range of specialized resources during collaboration; (3) there are task- or goal-driven autonomous processes; and (4) there is role-based communication. These features cause some inherent heterogeneity issues in VO development.

Some challenging usage policy issues arise when integrating the participant organizations and the WBCR into a VO, especially when spanning multiple physical institutions for WBCR sharing (Wasson and Humphrey 2003; Dumitrescu and Foster 2004; Dumitrescu, Wilde and Foster 2005). In practice, security mechanisms such as certificates and their authorization and authentication are typically exploited in the areas of grid and its applications. They aim to satisfy the hierarchical level security requirements for cross-organizational WBCR sharing and management (Bester et al. 1999; Foster and Kesselman 2003). More specifically, in grid environments, to meet the requirements of cross-organization WBCR invocation, the resource providers can grant the resource consumers the right, in the form of a certificate, to use certain resources for some agreed-upon time. It is often subject to the local usage policy and service level agreements, which should elaborate how resource access takes place in terms of where, what, who, and when it is allowed (Czajkowski et al. 2001; Liu et al. 2002; Peltz 2003; In et al. 2004). This WBCR usage policy can be elaborated by a SOA infrastructure as demonstrated in Figure 2. Here, the Service-Oriented Architecture (SOA) is essentially a collection of services that can be reified into a unit of work done by a service provider to achieve the desired results for a service consumer (Graham et al. 2001). The SOA-based resource access or collaborative execution can satisfy the requirement through bringing the usage constraints into the WSDL and the SOAP specification.

On the other hand, the collaboration among the participant organizations are often externally implemented in the form of workflow enabled by WBCR access based on certain usage policies. Traditionally, workflow systems are deployed inside a given organization or enterprise. For example, documents in a workflow process are exchanged among the process actors that manipulate the documents according to organization-wide procedures. A document often contains codes representing automated business procedures that will be invoked at various steps during workflow. In the workflow reference model presented by WfMC, the workflow engine is the core mechanism for workflow execution and workflow management (http://www.wfmc.org/standards/model.htm). In recent years, a new trend of inter-organization workflow systems has emerged in the context of electronic commerce
by interconnecting different organizational units to achieve competitive advantages. Moreover, with recent advances in pervasive devices and communication technologies, there are increasing demands in workflow application for ubiquitous access to networked services. These services extend support from Web browsers on personal computers to handheld devices and sensor networks. Taking advantage of the distributed technologies such as the CORBA, mobile agents, and Web service, the Web-based workflow segments can be orchestrated by RPC (Remote Procedure Call), message passing, RMI (Remote Method Invocation) or Applet invoked by a service agent. Despite great interest and improvement in technology in this area, complicated technical issues and organizational challenges remain to be solved (Berger et al. 2003; Maamar, Mostéfaoui, and Yahyaoui 2005).

Because cooperative computing can be treated as an ad-hoc workflow system, the WBCR usage policy should be a key factor for promoting the external workflow execution that spans the boundary of the participant organizations in the form of cross-organizational collaboration. According to the SOA scenario and the workflow specifications, static resource allocation or access schema can be prescribed in advance for later collaborations among resource providers and resource consumers based on their collaborative relations. For example, in grid environments, the single sign-on capability allows a grid user to delegate rights to another grid user, so that this user, in turn, can access some resource or start a computation on the former user’s behalf (Keahey, Ripeanu, and Doering 2003). It is helpful for spanning different administrative domains to exploit the WBCR in heterogeneous grid environments. However, the predefined usage policy does not satisfy all the requirements on resource sharing for the unpredictable nature of the collaborative execution (Dumitrescu and Foster 2004). Accordingly, although initially granted for a limited time, such delegation should be refreshed dynamically when needed during the collaborative execution; in addition, some new delegation should be granted dynamically as required on cross-organizational WBCR access during collaboration. To avoid conflict with the predefined usage policy, the status of target WBCR should be taken into consideration when some unexpected usage arises in a dynamic execution environment (e.g. how to visit the WBCR if it has been occupied by another user, when it is scheduled to release, etc.; Czajkowski et al. 2001; Wasson and Humphrey 2003; In et al. 2004; Dumitrescu et al. 2005).

Taking the above related work into consideration, a workflow engine-driven and SOA-based WBCR access control mechanism are discussed in the next section from the perspective of resource sharing.
4 A Workflow Engine-Driven and SOA-Based Resource Access Control Mechanism

4.1 Workflow Engine-Driven and SOA-Based WBCR Sharing Disciplines Based on Certificate Mechanism

In this section, the workflow engine-driven and SOA-based WBCR sharing disciplines are discussed from both static and dynamic aspects. Specifically, we will address the question: “How to orchestrate the static scheduled UP configuration and dynamic UP release efficiently from the workflow execution perspective.”

Because cooperative computing in grid environments can be treated as an ad hoc collaborative workflow execution, the collaborations among the WBCPs are mainly piloted by certain workflow application logics. The workflow application logic is classified into two styles in this paper. One style of application logic is the collaboration relation around certain problem solving; the other is around WBCR sharing in problem solving. Knowledge interaction piloted by knowledge flow logic or task logic is a typical application logic related to the former style, while database service innovation is a typical application related to the latter one in cooperative computing. In practice, the collaboration can be piloted by one of the two application logics or piloted jointly by the two application logics (Andrew et al. 2005; van Stokkum and Bal 2006). In this section, we concentrate on discussing the collaboration around certain WBCR sharing based on some usage policies from the workflow execution perspective.

Firstly, two key concepts of Web-Based Cooperative Peer (WBCP) and Web-Based Workflow Engine (WBWE) are put forward for better understanding of the collaborations engaged in cooperative computing. In our research, a WBCP can be an individual member, a participant team or a participant organization recruited in cooperative computing. According to the different privileges granted the WBCP, the WBCR is characterized by Private WBCR (Pri-WBCR) and public WBCR (Pub-WBCR) to highlight its different supervisors or users. A Pri-WBCR is often supervised exclusively by a certain WBCP. The WBCP can access and operate its Pri-WBCR freely. The other WBCPs cannot impose an operation on the Pri-WBCR. If possible, the supervisor of a Pri-WBCR can release its Pri-WBCR for the outside candidate WBCPs based on its local usage policy. In this case, the Pri-WBCR is transformed into a Pub-WBCR. In contrasted to the Pri-WBCR, a Pub-WBCR is another kind of resource, whose access is always open for the credible users. Here, the Pub-WBCR access is often piloted by the global WBWE that plays the role of a coordinator for Pub-WBCR management. Accordingly, a Pub-WBCR is always supervised by the WBWE directly, and its usage policy is always imposed on all the WBCPs. For example, in the motivating example presented in Section 2, the WBCRs held by the observation stations are some typical Pub-WBCRs, and the WBCRs held by the laboratories are typical Pri-WBCRs.

In accordance with those discussions, Figure 3 demonstrates the enactment principle for cross-organization WBCR sharing steered by a WBWE-driven and SOA-based certificate mechanism. Because static WBCR usage policy can be configured in the workflow specifications according to the execution temporal logic among the tasks, the WBCR invocations or a collaboration execution can be awakened or triggered automatically (Chen and Yang 2007). In our research, an awakened or triggered process is enabled based on a certificate mechanism. The numbers indicated in Figure 3 are specified by the step specifications as listed from Step 1 to Step 5. Figure 4 illustrates the activity diagram, in UML, in line with those step specifications.

**Step 1:** A WBCP registers itself and its Pri-WBCR to the global WBWE that holds a WBCP index and a WBCR index for later role and resource assignment. A WBCP endows the certificate granting right to the WBWE at the same time, that is, the global WBWE can grant a certificate, on behalf of the WBCP, to other WBCPs for exploiting the WBCP’s Pri-WBCR.
WBWE acts as the central certificate authority (headquarter) for harmonizing WBCR sharing in collaboration. Because the WBCP index and the WBCR index are managed by a central supervision mechanism, consistency is guaranteed between WBCP/WBCR assignment and global workflow specifications. As the certificate granting is conducted by the global workflow specifications, it is helpful not only for avoiding some conflicts in global WBCR allocation, but also for searching WBCR, discovering WBCR, and identifying if the target WBCR is occupied by a user. The occasion for granting a certificate is determined by WBWE according to the global workflow specifications and current state of the workflow execution.

**Step 2:** If an ECA rule predefined in global workflow specifications is triggered, the WBWE assigns a predefined task to a WBCP and initiates the collaboration between the WBCP and other WBCPs. Moreover, if an application for WBCR invocation is approved, the WBWE will grant a certificate to the applicant. The information for WBCR locating is also contained in the certificate’s data structure.

*This step is performed through a general certificate. The general certificate consists of some definitions or specifications as follows: (1) task definition, including task description, expected target, duration of implementation, etc.; (2) context logic of the task, such as the logical transition related to a task’s pre- and post-tasks, etc.; (3) context environment of the collaboration, such as the engaged WBCPs and their role definitions, location of the WBCRs engaged in collaboration, the key for WBCR access and its period of validity, QoS related to certain WBCR invocation, etc. If this step is triggered by Step 5, the WBWE should send a service for error recovery or exception handling. Generally, this step is initiated by ECA rules.*

**Step 3:** After granting the certificate, WBWE should send duplicated information to the target host of Pub-WBCR and to the collaborators of WBCP, for identifying and authenticating the future logging on or visiting.

*In Figure 3, the WBCP-i is a cross-organization WBCR consumer, and the WBCP-j is a Pri-WBCR provider. There is no Pub-WBCR to be accessed.*

**Step 4:** According to the key contained in the certificate and its security level, the certificate holder can access the target WBCR (e.g. exploiting the target WBCR, starting certain computing programs, or invoking some computational services) across borders of different security domains.

*Here, the WBCR access is subject to local usage policy and service level agreements, for instance, the WBCR consumer should comply with the read and write control, although it can cross borders of different security domains.*

**Step 5:** If there is an exception, for example, a task is not completed in the period of validity, the using process is forbidden and the WBCP should apply for a postponement. Additionally, if there is an unexpected requirement for WBCR access, or there is an unexpected requirement on collaboration during task implementation without predefinitions, the WBCP should apply for a new certificate for cross-organization WBCR access. If there is no exception, the collaboration will be successfully ended when meeting the deadline.

*Step 1–Step 4 can be treated as collaboration-compliant WBCR allocation or collaborative definition, while Step 5 is a collaboration-incompliant WBCR invocation or collaborative definition. This step is very important to harmonize the static certificate granting and the dynamic exception handling in resource sharing for cooperative computing.*

### 4.2 Typical Templates for Implementation

In the steps as listed in Section 4.1, because Step 2 contains the concrete collaboration specification, it is the key fragment for piloting later collaborations. Accordingly, some typical implementation templates can be put forward in XML to elaborate the application of Step 2. Please note that the implementation templates are initiated by the motivating example. Figure 5 demonstrates
an integrated implementation template corresponding to the implementation of Step 2. The concrete implementa-

Fig. 5  Step 2’s implementation in XML.

<TaskDefinition>
  <TaskDescription>
    This task is to simulate B area’s temperature fluctuation through an analysis program based on the database derived from ObservationStation1 by using the public super-computer.
  </TaskDescription>
  <TaskID>1DoITask-2</TaskID>
  <Duration Duringtime="3days"/>
  <Start>
    <date>06/01/18</date>
    <clock>8:00</clock>
  </Start>
  <End>
    <date>06/01/20</date>
    <clock>18:00</clock>
  </End>
  <Duration/>
  <Input>
    <item>1stQuarterAverageTemperatureCurve</item>
    <item>2ndQuarterAverageTemperatureCurve</item>
    <item>3rdQuarterAverageTemperatureCurve</item>
    <item>4thQuarterAverageTemperatureCurve</item>
  </Input>
  <Output>
    <item>ExpectedAverageTemperatureCurve</item>
  </Output>
</TaskDefinition>

Fig. 6 Instance of task definition in XML.

<CooperationContext>
  <Password>*****</Password>
  <Username>YYYY</Username>
  <AccessWBCR type="XXX">
    <WBCRItem>YYYY</WBCRItem>
    <Provider>YYYY</Provider>
    <IPAddress>xxx.x.xxx.xxx</IPAddress>
    <Port>xxxx</Port>
    <Duration Duringtime="xx units">
      <Start>
        <date>xx/xx/xx</date>
        <clock>xxxx</clock>
      </Start>
      <End>
        <date>xx/xx/xx</date>
        <clock>xxxx</clock>
      </End>
    </Duration>
  </AccessWBCR>
</CooperationContext>

Fig. 8 Template of cooperation context in XML.

4.3 Collaboration Context Logic Analysis

Generally, the collaboration between two tasks is piloted by certain application logic. In Figure 7, a task’s application context is depicted through specifying its pre- and
In this section, the application context logic of a task is further explored from the temporal logic perspective (Nebel and Bürckert 1995; Zaidi 1999). Firstly, two definitions are put forward as follows for specifying the temporal logic between two tasks, jointly.

**Definition 1.** Task $T_i$'s elapsing length is defined as $[T_{st-i}, T_{et-i}]$, where $T_{st-i} \leq T_{et-i}$; $T_{st-i}$ is the start time; and $T_{et-i}$ is the end time.

**Definition 2.** The basic temporal logic between two tasks of $T_i$ and $T_j$ engaged in a collaboration is defined as follows: Let $\{T_{st-i}, T_{et-i}\}$ and $\{T_{st-j}, T_{et-j}\}$ be a set of time, respectively, where $T_{st-i}, T_{et-i}, T_{st-j}$, and $T_{et-j}$ have the same meaning as presented in Definition 1. A basic temporal logic between $T_i$ and $T_j$ can be formalized as $t_i R t_j$, where $t_i \in \{T_{st-i}, T_{et-i}\}$, $t_j \in \{T_{st-j}, T_{et-j}\}$, and $R \in \{<, =, >\}$ is the relation between these two elements.

According to Definition 1 and Definition 2, the temporal logic between two tasks can be formalized by nine styles as listed in Table 1.

If there is a collaboration relation between two tasks, the temporal logics as listed in Table 1 can indicate the interaction direction. They reflect the collaboration context environment for global cooperative computing. Here, the generic collaboration context logic can be defined by Definition 3 below.

**Definition 3.** The collaboration context logic around a task consists of a group of its pre- and post-tasks and a spectrum of temporal logic specified between the task and its pre- and post-tasks engaged in the collaboration.

### 4.4 A Case Study

In practice, identifying or perceiving a task’s context environment is helpful for promoting the evolution of collaboration in a positive way (Zeng et al. 2004). Here, the motivating example presented in Section 2 is further explored to demonstrate the rationales presented in the previous sections. In this heterogeneous environment, the supercomputer, simulation and analysis program, and data resources are typical WBCRs, and cross-organizational WBCR sharing occurs frequently. The resource broker service plays the role of WBWE. It aims at locating WBCRs efficiently, invocating and balancing resources sharing for simulation and analysis.

Figure 11 illustrates a collaboration process, which is specified below. The numbers illustrated in Figure 11 indicate different tasks. They jointly form a round of collaboration.

1. To achieve its computing goal, Lab-$i$ exploits the data sets stored in observation station-$i'$ (indicated by ‘1’; note that $i' \neq 1$).
2. During its computing process, a certain computing program, which is indispensable for Lab-$i$’s activity, is executed by the supercomputer located at the computing center (indicated by ‘2’).
3. Once the computing program is initiated, some data resources stored in observation station-$i'$ will be invocated by the supercomputer (indicated by ‘3’), and some temporary data produced by Lab-$j$’s computing process are required dynamically (indicated by ‘4’, and please note that $j \neq 1$ and $j \neq i$).
4. The computing results are returned to Lab-$i$ (indicated by ‘5’) and Lab-$j$ (indicated by ‘6’).

It is a typical round of collaborative workflow execution for cooperative computing in grid environments. In the collaboration, the application context environment can be depicted by specifying the temporal logic among the tasks engaged in the collaboration. Table 2 specifies the spectrum of the temporal logic enabled in the context environment.

4.5 Performance Analysis

The central WBWE mechanism presented in Section 4.1 is obviously based on the scenario of SOA. It demonstrates an access control mechanism based on a uniform usage policy. It aims at satisfying the static and dynamic resource invocation requirements across different technologies or platforms in a runtime environment. Some performance measurements are indicated as follows to conclude this section:

1. In the central control mechanism, the global WBCP and WBCR index, and the global workflow specifications can be coordinated synchronously by a centralized authority (headquarters, i.e., WBWE). It facilitates the global harmonization among local workflow collaborations. Therefore, compared to the flat and distributed control strategy recruited by some P2P systems, in which the fragments of

<table>
<thead>
<tr>
<th>No.</th>
<th>Logic item</th>
<th>Logical condition</th>
<th>Legends</th>
<th>Notations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Before()</td>
<td>$T_{at,i} &lt; T_{at,j}$</td>
<td>$T_{t_{i,j}}$</td>
<td>$T_{t_{i,j}}$</td>
</tr>
<tr>
<td>2</td>
<td>Meet()</td>
<td>$T_{at,i} = T_{at,j}$</td>
<td>$T_{t_{i,j}}$</td>
<td>$T_{t_{i,j}}$</td>
</tr>
<tr>
<td>3</td>
<td>Overlap()</td>
<td>$(T_{at,i} &lt; T_{at,j}) \cap (T_{at,j} &lt; T_{at,i})$</td>
<td>$T_{t_{i,j}}$</td>
<td>$T_{t_{i,j}}$</td>
</tr>
<tr>
<td>4</td>
<td>Start-1()</td>
<td>$(T_{at,i} = T_{at,j}) \cap (T_{at,j} &lt; T_{at,i})$</td>
<td>$T_{t_{i,j}}$</td>
<td>$T_{t_{i,j}}$</td>
</tr>
<tr>
<td>5</td>
<td>Start-2()</td>
<td>$(T_{at,i} = T_{at,j}) \cap (T_{at,i} &lt; T_{at,j})$</td>
<td>$T_{t_{i,j}}$</td>
<td>$T_{t_{i,j}}$</td>
</tr>
<tr>
<td>6</td>
<td>During()</td>
<td>$(T_{at,j} &lt; T_{at,i}) \cap (T_{at,i} &lt; T_{at,j})$</td>
<td>$T_{t_{i,j}}$</td>
<td>$T_{t_{i,j}}$</td>
</tr>
<tr>
<td>7</td>
<td>Finish-1()</td>
<td>$(T_{at,i} = T_{at,j}) \cap (T_{at,j} &lt; T_{at,i})$</td>
<td>$T_{t_{i,j}}$</td>
<td>$T_{t_{i,j}}$</td>
</tr>
<tr>
<td>8</td>
<td>Finish-2()</td>
<td>$(T_{at,i} = T_{at,j}) \cap (T_{at,i} &lt; T_{at,j})$</td>
<td>$T_{t_{i,j}}$</td>
<td>$T_{t_{i,j}}$</td>
</tr>
<tr>
<td>9</td>
<td>Equal()</td>
<td>$(T_{at,i} = T_{at,j}) \cap (T_{at,i} = T_{at,j})$</td>
<td>$T_{t_{i,j}}$</td>
<td>$T_{t_{i,j}}$</td>
</tr>
</tbody>
</table>

© 2008 SAGE Publications. All rights reserved. Not for commercial use or unauthorized distribution.
the workflow specification are distributed to the autonomous WBCPs, the central control mechanism is helpful for keeping the consistency among local workflow executions through updating cooperative information synchronously (Fu, Bultan, and Su 2005). Moreover, the flat and distributed coordination is initiated among the WBCPs of their own accord, case by case, without an authoritative coordinator. This is apt to lead to some inconsistency in global collaboration around large-scale cooperative computing. In some cases, the global collaboration can break down if WBCPs do not come to an agreement in exception handling. So, it is suitable for some routines, clerical situations or standard processes with less exception such as production workflow systems and administrative workflow systems (Alonso et al. 2000; Stohr and Zhao 2001). Because global coincidence is needed for dynamic cross-organizational communication and collaboration between WBCRs the central control mechanism presented in this section is suitable for an ad hoc or collaborative workflow system to promote its cooperative computing around large scale or complex problem solving (Alonso et al. 2000; Stohr and Zhao 2001). Taking advantage of the hierarchical control methods, the central control strategy can be efficiently extended based on certain self-governing local policy enacted inside a local workflow fragment.

2. Note that the resource sharing mechanism demonstrated in Figure 3 is not a P2P system, although it is similar to Napster in structure. As one of the earliest P2P implementations and as a typical unstructured P2P system, Napster is characterized by some common properties of P2P systems such as being decentralized, self-organizing, having no central control mechanism, and having no usage policy on security during resource sharing (Iamnitchi, Ripeanu, and Foster 2002; Aberer et al., 2003; Iamnitchi and Foster 2003). In our discussion, Figure 3 is different from a P2P system in operation. For example, Figure 3 is a structured resource sharing mechanism characterized by a central control mechanism of the engine. It has a specific usage policy during resource invocation. Furthermore, the WBCPs are organized by predefined workflow specifications or a scheduled cooperative schema rather than in a self-governing fashion in an unstable environment.

3. WBWE integrates a certificate mechanism, an index of WBCP, an index of WBCR, and some traditional issues related to workflow executions. It enriches the content of the workflow engine compared with traditional workflow execution enacted inside an organization. The WBWE initiates the distributed workflow execution according to the predefined workflow specification. Directory-based metadata of the WBCP index and WBCR index make it easy in practice to update some collaborative information synchronously. Here, it is worth noting that the implementation presented in this section

---

**Table 2**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Equal(1, 1)</td>
<td>During(2, 1)</td>
<td>During(3, 1)</td>
<td>During(4, 1)</td>
<td>During(5, 1)</td>
<td>During(6, 1)</td>
</tr>
<tr>
<td>2</td>
<td>×</td>
<td>Equal(2, 2)</td>
<td>Before(2, 3)</td>
<td>Before(2, 4)</td>
<td>Before(2, 5)</td>
<td>Before(2, 6)</td>
</tr>
<tr>
<td>3</td>
<td>×</td>
<td>×</td>
<td>Equal(3, 3)</td>
<td>Finish-1(3, 4)/Finish-2(3, 4)</td>
<td>Before(3, 5)</td>
<td>Before(3, 6)</td>
</tr>
<tr>
<td>4</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>Equal(4, 4)</td>
<td>Before(4, 5)</td>
<td>Before(4, 6)</td>
</tr>
<tr>
<td>5</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>Equal(5, 5)</td>
<td>Equal(5, 6)</td>
</tr>
<tr>
<td>6</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>Equal(6, 6)</td>
</tr>
</tbody>
</table>

**Fig. 11** A round of typical scientific collaboration.
section is centered on the collaboration around WBCR sharing. How to promote the collaboration around dynamic knowledge sharing is another important topic in cooperative computing. Figure 3 concentrates on cross-organization Pri-WBCR sharing. It takes no consideration of Pub-WBCR sharing. Facilitating the uniform resource management, Pub-WBCR also registers them into the WBWE to perfect the global WBCR index. This topic is discussed in the next section.

5 A Workflow Engine-Driven and SOA-Based Cooperative Computing Paradigm in Grid Environments

In this section, a workflow engine-driven and SOA-based cooperative computing paradigm is explored in grid environments, based on a specific project of SOA-Based Engine-Driven Structured Cooperative Computing Environment (SOA&EDSCCE). Our main objective in this project is to set up a structured Problem Solving Environment (PSE) for promoting cooperative computing in grid environments. In project SOA&EDSCCE, the components engaged in cooperative computing are generally classified into four categories of entities: WBWE, WBCP, Task, and WBCR. The collaboration is navigated by the interaction among these four entities. Specifically, WBCPs collaborate on a computing task by sharing some WBCRs in the form of a VO, which is driven by a WBWE. The role of a WBCP can be defined in the scheduled workflow specification in advance. Facilitating the collaboration among those entities, a VO can be formalized by Definition 4 by integrating those entities.

**Definition 4.** A VO is defined as an eight-tuple as follows: 

\[
VO = (WBCP, WBCR, Task; R_{TT}, R_{TP}, R_{TPr}, R_{PR}, R_{PRr})
\]

where, WBCP, WBCR, and Task stand for a set of WBCPs, a set of WBCRs, and a set of tasks, respectively; 

- \(R_{TT}\) stands for the global logical relations among the tasks based on the application logic presented in Table 1; 
- \(R_{TP}\) stands for organizational discipline imposed on WBCP around \(R_{TT}\); 
- \(R_{TPr}\) stands for collaborative relations among WBCP around \(R_{TT}\); 
- \(R_{PR}\) stands for the resource allocation relation around \(R_{TT}\); 
- \(R_{PRr}\) stands for service relations between resource consumers and resource providers around \(R_{TT}\).

The relations defined in Definition 4 indicate the application logic from different aspects for piloting workflow execution around the cooperative computing. Different workflow views can be developed based on the relations for different service invocations (Chiu et al. 2004; Andrew et al. 2005; Maamar, Mostéfaoui, and Yahyaoui 2005). In the project of SOA&EDSCCE, the relations among these entities are demonstrated in Figure 12 in UML; and the definitions of the classes of WBCP, WBCR, Task and WBWE are illustrated in Figure 13, where the symbol “+” stands for public property, the symbol “-” stands for private property, and the symbol “#” stands for protected property. The attributes and methods contained in these classes are elaborated in Table 3. Here, a WBCP registers itself by the function of WBCP.WBCPRegister(), through which the Pri-WBCR is registered simultaneously. Nevertheless, the Pub-WBCR should register itself alone by the function of Pub-WBCR.WBCRRegister(). The function of Task.TaskDefine() defines an individual task; the function of WBWE.WorkflowFragmentSpecify() prescribes a set of tasks implemented by an individual WBCP in conformance with a local workflow execution; and the
Table 3
Specifications of the attributes/methods listed in the classes as illustrated in Figure 12

<table>
<thead>
<tr>
<th>No.</th>
<th>Item</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>WBCP.IDOfAWBCP</td>
<td>Attribute for indicating the ID of a WBCP.</td>
</tr>
<tr>
<td>2</td>
<td>WBCP.WebLocationOfAWBCP</td>
<td>Attribute for indicating the location of a WBCP on the Internet, such as the IP address, work group name, etc.</td>
</tr>
<tr>
<td>3</td>
<td>WBCP.WBCPRegister()</td>
<td>Function for registering both WBCP and its Pri-WBCR to a structured WBWE or central control unit.</td>
</tr>
<tr>
<td>4</td>
<td>WBCP.MethodForWBCRInvocation()</td>
<td>Function for invocating required resource from Pub-WBCR or Pri-WBCR.</td>
</tr>
<tr>
<td>5</td>
<td>WBCP.CommunicateMethodForCollaborate()</td>
<td>Function for communicating with other WBCP during cooperative computing.</td>
</tr>
<tr>
<td>6</td>
<td>WBCP.ContextAware()</td>
<td>Function for identifying the information contained in the general certificate, or the information related to a PSE such as the roles, the resources, the tasks, the tools, the collaborators, and other engaged factors.</td>
</tr>
<tr>
<td>7</td>
<td>WBCP.ApplyForCertificate()</td>
<td>Function for applying for a certificate from a WBWE or a central control unit to invoke certain Pri-WBCR or Pub-WBCR.</td>
</tr>
<tr>
<td>8</td>
<td>Pub-WBCR.IDOfAWBCR</td>
<td>Attribute for indicating the ID of a WBCR.</td>
</tr>
<tr>
<td>9</td>
<td>Pub-WBCR.WebLocationOfAWBCR</td>
<td>Attribute for indicating the location of a WBCR on the Internet, such as the IP address, server name, etc.</td>
</tr>
<tr>
<td>10</td>
<td>Pub-WBCR.WBCRRegister()</td>
<td>Function for registering a Pub-WBCR to a WBWE or a central control unit before cooperative computing.</td>
</tr>
<tr>
<td>11</td>
<td>Pub-WBCR.AccessControl</td>
<td>Function for access control when WBCR sharing, such as certificate authenticating, Read/Write control, etc.</td>
</tr>
<tr>
<td>12</td>
<td>Pri-WBCR.OwnerOfPri-WBCR</td>
<td>Attribute for indicating the owner of a Pri-WBCR.</td>
</tr>
<tr>
<td>13</td>
<td>Pri-WBCR.LocalUsagePolicySpecify()</td>
<td>Function for releasing local usage policy to steer Pri-WBCR sharing for different WBCR consumer.</td>
</tr>
<tr>
<td>14</td>
<td>Task.IDOfATask</td>
<td>Attribute for indicating the ID of a task.</td>
</tr>
<tr>
<td>15</td>
<td>Task.CandidateWBCPForATask</td>
<td>Function for assigning certain WBCPs for a task implementation.</td>
</tr>
<tr>
<td>16</td>
<td>Task.CandidateWBCRForATask()</td>
<td>Function for specifying certain WBCRs for a task implementation.</td>
</tr>
<tr>
<td>17</td>
<td>Task.TaskDefine()</td>
<td>Function for defining a task from task description, input parameters, expected target, etc.</td>
</tr>
<tr>
<td>18</td>
<td>Task.ContextInformation()</td>
<td>Function for identifying a task’s context based on its context logic presented in Section 4.3.</td>
</tr>
<tr>
<td>19</td>
<td>WBWE.IDOfAWBWE</td>
<td>Attribute for indicating the ID of a WBWE.</td>
</tr>
<tr>
<td>20</td>
<td>WBWE.WBCPDiscover()</td>
<td>Function for discovering and locating a WBCP based on the WBCP index when initiating a certain collaboration.</td>
</tr>
<tr>
<td>21</td>
<td>WBWE.WBCRDiscover()</td>
<td>Function for discovering and locating a WBCR based on the WBCR index when initiating a certain collaboration.</td>
</tr>
<tr>
<td>22</td>
<td>WBWE.WorkflowFragmentSpecify()</td>
<td>Function for depicting a local workflow fragment performed by a WBCP. The local workflow fragment consists of a task or a set of task.</td>
</tr>
<tr>
<td>23</td>
<td>WBWE.GlobalWorkflowSpecify()</td>
<td>Function for specifying the global collaborative workflow system from global control-flow and data-flow point of view. It aims at setting a global interaction and cooperative logic among WBCPs, tasks and WBCRs to pilot cooperative computing.</td>
</tr>
<tr>
<td>24</td>
<td>WBWE.CertificateGrant()</td>
<td>Function for granting a certificate to a valid consumer for certain Pri-WBCR or Pub-WBCR access.</td>
</tr>
<tr>
<td>25</td>
<td>WBWE.ExceptionHandle()</td>
<td>Function for handling some exceptions during cooperation workflow execution. It is responding to Step 5 presented in Section 4.2.</td>
</tr>
</tbody>
</table>
function of WBWE.GlobalWorkflowSpecify() elaborates the global cooperative computing from the global workflow execution perspective. These three functions depict the collaborating behavior from different aspects.

Accordingly, a specific infrastructure derived from the project SOA&EDSCCE is demonstrated in Figure 14 based on the structured mechanism presented in Section 4, and those attributes and functions listed in Table 3. Compared with Figure 2, the WBWE illustrated in Figure 14 plays the role of UDDI Registry, the binding process between WBCP and WBCR is initiated through certificate granting and updating. In Figure 14, the task specification is enclosed in WBWE to facilitate global workflow execution, and the collaboration between two WBCPs is promoted by the task context logic specified in WBWE.

In Figure 14, the WBCP, Task, and WBCR register themselves into the WBWE as the first operation before starting cooperation. The later enactments of cooperation are navigated by the number on the left of the slash. Note that the number on the left of a slash stands for the operation step, and the bracketed numbers on the right of a slash stand for the methods or attributes that will be invoked partially or totally during the operation. For example, No.1/[1,2,3,7] indicates that the No. 1 step is implemented jointly or partially by attribute 1, function 2, function 3, and function 7 that have been defined in Table 3. Here, the bracketed numbers have no strict temporal logic and only indicate the operations that may occur for promoting cooperative computing. More specifically, in Figure 14, No. 1 and No. 2 stand for the WBCP’s register processes; No. 3 stands for the Pub-WBCR’s register process; {No.4, No.5, No.14} indicates the operations for initiating and steering certain collaboration; Both {No.6, No.7} and {No.8, No.9} indicate the processes for navigating Pub-WBCR sharing; Both No. 10 and No. 12 stand alone for Pri-WBCR access inside an organization; Both {No.10, No.11} and {No.12, No.13} point out the authorization process for cross-organization Pri-WBCR sharing.

In Figure 14, the mechanism of WBWE plays a key role in orchestrating the static cooperative workflow specifications and dynamic collaborative executions (Peltz 2003). For example, after registration, a WEC can learn the cooperative relations and its acting role from the global workflow specification, if needed, during its collaboration with other WBCPs. Besides, a WBCP can efficiently discover and locate available WBCRs exploited during collaboration with the help of WBCR index stored in WBWE. It is useful to meet the collaboration-compliant applications. In a runtime environment, the function...
of WBCP.ExceptionHandle() deals with some unexpected dynamic resource requirements or some unexpected collaborations during execution. It is helpful for dealing with some “blind” or collaboration-incompliant applications (e.g. a collaboration-unaware situation). In practice, to refresh a certificate’s validity or to discover a WBCP for an unexpected collaboration is a typical exception handling during cooperation execution. The detailed discussion of general exception handling is outside of the scope of this paper and the reader is referred to Hagen and Alonso (2000).

6 Comparison Analysis and Further Discussion

As stated by some researchers (Foster and Kesselman 2003), grid can act as an integrative application paradigm in the form of VO that prevails as a practical infrastructure in future. It can also act as a Problem Solving Environment (PSE) to promote large-scale cooperative computing. As an infrastructure of a special PSE, grid should enable complex problem solving from two dimensions: (1) facilitating the WBCR access in a cross-organizational collaboration, (2) a secure communication mechanism for the collaboration among the WBCPs.

As such, the workflow application paradigm discussed in this paper is a grid workflow application paradigm, rather than a business workflow paradigm. In addition, the cooperative computing discussed in this paper is explored based on the precondition that it is enacted around a large-scale or a complex scientific problem solving executed in a distributed PSE. Compared with related work (CCF Project Team 2000; Holsapple and Luo 2003; Tsai et al. 2005; Zhang et al. 2006), our research idea originated from workflow engine technologies by taking advantage of the certificate-based dynamic computing paradigm in grid environments. Moreover, the SOA provides a collaboration fashion among WBCPs, and the structured WBCP and WBCR indices regulate the SOA-based applications. This guarantees that the autonomic WBCPs can be under the control of a high-level centralized authority (headquarters, i.e. WBWE). This is helpful for coordinating the collaboration in distributed grid environments.

Initially, the period of a certificate’s validity is assigned according to the usage policy. The period of certificate’s validity reflects the lifetime of collaboration. We can efficiently manage the well-defined authorization assertions. The dynamic WBCR requirement often arises in the collaboration and is often not pre-specified. How to deal with this exception is always a challenge to the well-defined authorization assertions. For example, the CPU sharing or the allocated disk space might change during the collaboration lifetime. Consequently, the enforcement of these allocations should be enacted. The usage policy should reflect the static WBCR access issues on security, and it should also deal with the dynamic WBCR access issues on WBCR sharing. It guarantees that a task can focus on its inside-execution with less negative effect caused by WBCR access. Furthermore, to satisfy some special access requirements, a Pri-WBCR can set its local security control policy according to its service ability, storage space, security level, networking speed, etc. The Pri-WBCR owners can set up their local Service Level Agreements (SLAs) for different users. Accordingly, it is interesting to orchestrate the static definitions of security, the dynamic WBCR requirements, and the hierarchical usage policies.

7 Conclusions and Future Work

In this paper, a workflow engine-driven SOA-based cooperative computing paradigm in grid environments is explored by taking advantage of the certificate-based usage policy exploited in grid applications. Furthermore, an infrastructure in conformance with a specific project SOA&EDSCCE is presented. It provides a helpful reference framework to promote cooperative computing in grid environments, especially for grid workflow development. In our future research, we will concentrate on exploring the application logics to enhance the knowledge interaction for scientific problem solving based on the infrastructure presented in this paper.

Acknowledgments

This paper is partly supported by the NSFC Project under Grants No.60303025, No.60673017, No.60673119, No.60721002, and No.60736015; National Grand Fundamental Research 973 Program of China under Grants No. No.2002CB312002, and No.2006CB303000; 863 Project under Grants No.2007AA01Z178; Jiangsu Provincial NSF Project under Grants No.BK2004411 and No.BK2007137; and Program for New Century Excellent Talents in University.

Author Biographies

Wanchun Dou is an associate professor in the Department of Computer Science and Technology, Nanjing University, PR China. He received his Ph.D. degree from Nanjing University of Science and Technology, P.R.China, in 2001. Then he continued his research work as a post-doctoral researcher in the Department of Computer Science and Technology, Nanjing University from 2001 to 2002. In 2005, he was a visiting scholar at Hong Kong University of Science and Technology. To date he has published more than 50 papers in international journals, international conferences, and some Chinese core jour-
Jinjun Chen received his Ph.D. degree in computer science and software engineering from Swinburne University of Technology, Melbourne, Australia, in 2007. He is currently a lecturer at the Centre for Information Technology Research in the Faculty of Information and Communication Technologies at Swinburne University of Technology, Melbourne, Australia. His research interests include grid and cooperative computing, service oriented computing (engineering, planning, negotiation, agreement, verification and validation), workflow management and application in grid environments, web services environments and generic service oriented computing environments, reliable workflow software systems in grid environments, web services environments and generic service oriented computing environments (verification and validation, testing, QoS and resource scheduling, exception handling, other non-functional issues), semantics and knowledge management.

Jianxian Liu is a professor at the Department of Computer Science and associate director of the Knowledge Grid Laboratory at Hunan University of Science and Technology. He received his master’s degree from Central South University and his Ph.D. from Shanghai Jiao Tong University. He has won the Youth Backbone Teacher prize of Hunan Province, 2003. His work has been reported in more than 30 publications in journals including: Information & Management, Future Generation Computer Systems, Chinese Journal of Electronics, and International Journal of Concurrent Engineering: Research and Application. He participates actively into international conferences and workshops. He is now directing several research projects, such as the NFS project, the 973 Program of China, and the Hunan Provincial NSF Project. His research interests include semantics and knowledge grid, workflow management systems, distribute computing, and e-Commerce.

Shing-Chi Cheung is an associate professor of computer science and associate director of the CyberSpace Center at the Hong Kong University of Science and Technology. Dr. Cheung received his B.Sc. degree in electrical engineering from the University of Hong Kong in 1984. He received his M.Sc. and Ph.D. degrees in computing from the Imperial College of Science, Technology and Medicine, University of London, London, UK in 1988 and 1994 respectively. Dr. Cheung is an associate editor of the IEEE Transactions on Software Engineering. He participates actively in the organization and program committees of many major international conferences on software engineering, distributed systems and web technologies, such as ICSE, FSE, ISSTA, ASE, ICDCS, ER, COMPSAC, APSEC, QSC, EDOC, SCC and CEC. His research interests include software engineering, services computing, ubiquitous computing, and embedded software engineering. His work has been reported in more than 90 publications in leading internationally refereed journals and conferences, which include TOSEM, TSE, ASE, DSS, TR, ICSE, FSE, ESEC and ER. He is also a Chartered Fellow of the British Computer Society and a senior member of the IEEE.

Guihai Chen is a full professor at Nanjing University and a guest professor at Wayne State University. He earned his B.S. degree in computer software from Nanjing University in 1984, his M.E. degree in computer applications from Southeast University in 1987, and his Ph.D. degree in computer science from the University of Hong Kong in 1997. He has published more than 100 peer-reviewed papers, and more than 60 of them are in well-archived international journals and conference proceedings. More than 30 papers are published in well-known SCI journals such as IEEE Transactions on Parallel & Distributed Systems, The Computer Journal, Computer Communications, Information Systems, Information and Software Technology, Advances in Engineering Software, Journal of Supercomputing, International Journal of Foundations of Computer Science, and International Journal of Performance Evaluation. He has a wide range of research interests with focus on sensor networks, peer-to-peer computing, and high-performance computer architecture. He has participated as a chief investigator or co-investigator in many state key research projects, including the National NSF projects, the National 863 Plan for High-Tech Research and Development, the National 973 Program for Grand Fundamental Research, and the National Climbing Plan for Science and Technology.

Shaokun Fan is a master’s candidate in the Department of Computer Science and Technology at Nanjing University, China. He received his bachelor’s degree in physics from Nanjing University, in 2004. He has worked as a software engineer and has much practice in the design and development of software systems. He participated actively in several research projects. His research interests are workflow technology, grid computing, and service computing. He has published some papers in these areas.
References


In, J., Avery, P., Cavanaugh, R., and Ranka, S. (2004). Policy based scheduling for simple Quality of Service in grid computing, In *Proceedings of the 18th international Parallel and Distributed Processing Symposium (IPDPS 04)*, Santa Fe, New Mexico, USA.

Keahey, K., Ripeanu, M., and Doering, K. (2003). Dynamic creation and management of runtime environments in the grid, In *Proceedings of Workshop on Designing and Building Grid Services (WDBGS’03, in conjunction with GGFG)*, Chicago, IL, USA.


