

Multi-Grid, Multi-User Workflows in the P-GRADE Grid Portal★

Péter Kacsuk and Gergely Sipos

MTA SZTAKI Computer and Automation Research Institute, P.O. Box 63, H-1518, Budapest, Hungary

E-mail: {kacsuk, sipos}@sztaki.hu

Received 6 June 2005; accepted in revised form 6 December 2005

Key words: collaborative environment, DAGMan, Globus Toolkit, Grid computing, Grid portal, Gridsphere, workflow management

Abstract

Computational Grids connect resources and users in a complex way in order to deliver nontrivial qualities of services. According to the current trend various communities build their own Grids and due to the lack of generally accepted standards these Grids are usually not interoperable. As a result, large scale sharing of resources is prevented by the isolation of Grid systems. Similarly, people are isolated, because the collaborative work of Grid users is not supported by current environments. Each user accesses Grids as an individual person without having the possibility of organizing teams that could overcome the difficulties of application development and execution more easily. The paper describes a new workflow-oriented portal concept that solves both problems. It enables the interoperability of various Grids during the execution of workflow applications, and supports users to develop and run their Grid workflows in a collaborative way. The paper also introduces a classification model that can be used to identify workflow-oriented Grid portals based on two general features: Ability to access multiple Grids, and support for collaborative problem solving. Using the approach the different potential portal types are introduced, their unique features are discussed and the portals and Problem Solving Environments (PSE) of our days are classified. The P-GRADE Portal as a Globus-based implementation for the classification model is also presented.

1. Introduction

The Grid was originally proposed as a global computational infrastructure to solve grand-challenge, computational intensive problems that cannot be handled within reasonable time even with state of the art supercomputers and computer clusters [1]. Grid computing tackles these tasks by aggregating geographically and architecturally dispersed hard-

ware and software resources into large virtual super-resources.

Meanwhile Grids can be realized relatively easily by building a uniform middleware layer, such as Globus [2], on top of the hardware and software resources, the programming concept of such distributed systems is not obvious. There are several different programming models to exploit Grids in a way that can accelerate our computations.

The simplest approach is the desktop Grid concept [3, 4] where the application is divided into a large number of independent work packages and these packages are processed in a parallel way. Though this approach is very efficient, there are applications where the subprograms are more closely coupled and

★ The work described in this paper is supported by the Hungarian Grid project (IHM 4671/1/2003), by the Hungarian OTKA project (No. T042459) and a collaboration project with the University of Reading.

run-time interaction among them is required. For such problems typically MPI [5] is considered as the best supporting concept and tool. MPI works very efficiently on supercomputers and clusters. However, if an MPI application is distributed on several remote sites (using for example the MPICH-G2 [6] version) then – due to the high latency of wide area networks – the performance turns out to be much worse than it was originally expected [7].

Complex problems often require the integration of several existing sequential and parallel programs into a single application in which these codes are executed according to a graph, called workflow. The workflow concept introduces parallelism at two levels. The top-level parallelism comes from the graph concept, i.e., codes contained by independent branches can be executed simultaneously. The bottom level parallelism can be applied if some of the workflow nodes are themselves parallel programs (e.g., MPI [5] or PVM [34]). Both top level and bottom level parallelism can be exploited if the parallel branches contain parallel nodes. In such case several supercomputers or clusters can be used simultaneously, and every parallel program would use one of these systems. Consequently, multi-site parallel application execution can be achieved without any performance degradation. The approach combines the benefits of traditional single-site parallel processing and Grid-like multi-site processing. Although there is a large number of workflow-oriented Grid activities, most of them do not exploit these two possible levels of parallelism [8–10].

After the proper parallel processing approach has been selected the next step is to choose a suitable application developer and execution environment. Grids are typically accessed through portals [11, 12, 45] that serve as both Grid application developer and executor environments. As Grid technology matures the number of production Grids dynamically increases. Although sometimes multiple Grids are served by the same portal, usually different portals are installed for different Grids. Even if a portal is connected to multiple Grids, applications that utilize services from these Grids simultaneously, are not supported. The result is distressing: We are witnessing the formation of separated islands of Grids, without having bridges connecting them. If a user starts to develop an application on one of these ‘islands’ and it turns out that it does not provide the required capacities, service types or quality of

services, then he/she must obtain a certificate for a different Grid, an account for its portal, must learn this portal and finally the application must be migrated into this new environment. If he/she was lucky the new environment supports the same application developer interfaces, otherwise the code has to be modified as well.

Since Grids are inevitably the most complex computational infrastructures of our times, applications built on top of them must be highly advanced as well. These programs have to interface with different types of services (sometimes built with different APIs) and must handle the issues related to the distributed nature of the infrastructure (concurrency, transaction management, etc.). The development of advanced applications cannot be efficient without collaborative tools [24]. Collaboratively designed Grid programs can accumulate the knowledge of multiple persons and can exploit underlying services much more efficiently than single-user applications do. Because the development and execution phases of Grid applications are closely related to each other, traditional collaborative tools cannot support the collaborative development and execution of Grid applications.

The paper argues that next generation Grid portals can be the answer for these two challenges: They could support collaborative applications that utilise services from multiple Grids simultaneously. Section 2 introduces a classification model that evaluates workflow-oriented Grid portals according to their multi-Grid, multi-user capabilities. The aim is to identify the functions that must be implemented by next generation Grid portals if they want to overcome the above-discussed issues. Section 3 presents the P-GRADE Portal [44], a workflow-oriented portal that is used by Section 4 and Section 5 to demonstrate how multi-Grid, multi-user portals can be implemented on top of the Globus middleware [2]. Although Globus and P-GRADE Portal are used through the paper to demonstrate the concept of multi-Grid, multi-user workflows, the approach can be easily generalized for other types of Grid middleware and portals.

2. Classification of Grid Portals

Most of the current Grid portals are used to visualize Grid information (properties of resources, status of

Table 1. Classification of workflow-oriented Grid portals.

	Multiple isolated users (MIxx)	Multiple collaborative users (MCxx)
Single isolated Grid (xxSI)	MISI portals	MCSI portals
Multiple isolated Grids (xxMI)	MIMI portals	MCMI portals
Multiple collaborative Grids (xxMC)	MIMC portals	MCMC portals

jobs, results of jobs, etc.) and to facilitate the submissions of individual jobs [12–14]. Recent Grid portals and PSEs support the development and submission of jobs either as components of parameter studies or workflows. The paper is focusing onto portals that support the most complex (and advanced) form of job group submission, i.e., workflow submission, since we believe that very soon this feature will be a standard part of any Grid portal.

As Flynn classified parallel computers [16] according to the number of applied processors and the number of data units processed in parallel, we propose a classification for workflow-oriented Grid portals. Our classification is based on two parameters. First, on the number of users that can access (edit, execute, steer, etc.) the same application through the portal simultaneously. Second, on the number of Grids a given portal is connected to and can use to execute jobs. Two types of portals can be distinguished by the first property: Multiple isolated user portals and multiple collaborative user portals. Using the second property three kinds of portals can be identified: Single isolated Grid portals, multiple isolated Grid portals and multiple collaborative Grid portals. According to these statements Table 1 can be created.

2.1. Grid Portals and Grids

The first row of the table represents the current production Grid portal approach where each portal installation is connected to a single Grid system, i.e., users of the portal can access resources (services) that belong to one rewarded Grid. This restriction limits the usability of Grids and portals in many ways. For example – just to mention one of the many limits – if the connected Grid is overloaded the users’ jobs cannot be redirected to another, less overloaded Grid. One remedy for this problem could be the access of

several Grids by the same portal. The second and third rows of the table represent portals that can exploit such a multi-Grid connection. These portals can be further classified whether the Grids connected by them are isolated from each other or they can collaborate. Collaborative Grids are able to solve the jobs of the same workflow together, executing the different workflow branches in parallel. For example if a workflow has two branches, then jobs belonging to branch ‘1’ can be executed in Grid ‘A’, meanwhile jobs belonging to branch ‘2’ can be simultaneously executed in Grid ‘B.’ If such a simultaneous execution of the workflow components – coordinated by the Grid portal – is not possible, then we say that the connected Grids are isolated. In such case every component of the workflow is executed either in Grid ‘A’ or in Grid ‘B.’¹ We refer to portals that can connect several Grids in such a restricted way as xxMI portals. This class is represented by the second row of Table 1. XxMC portals, represented by the last row, are able to support the simultaneous, collaborative execution of components of a workflow in several connected Grids (See Figure 1).

2.2. Grid Portals and Users

Current portals support only isolated users, i.e., Grid users cannot collaborate via the portal either to develop applications together or collaboratively run existing applications. Although several Grid portals and PSEs support collaborative work [36, 37], these environments simply extend the isolated user workspaces with traditional information sharing tools such as chatting, message sending, audio and video streaming. As a result users are able to communicate with each other through the portal, but they have to build and execute Grid applications individually. Although some portal implementations enable users to apply each others applications as starting points to develop new programs, defining complex systems in this way is unacceptably time consuming.

In our terminology multi-user portals provide controlled and concurrent access to Grid applications for multiple users during both the application development and execution phases. Workflow-oriented portals that correspond to this definition are

¹ Despite all the components of the workflow are executed in the same Grid the different components can run on different sites in parallel.

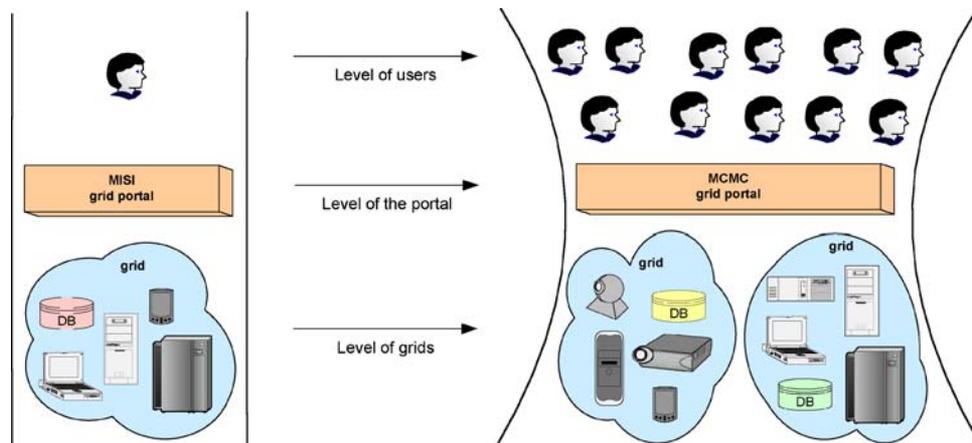


Figure 1. From isolated-user, isolated-Grid portal to collaborative-user, collaborative-Grid portal.

denoted as MCxx portals in our model, and they are presented in the last column of Table 1. Every other portal implementation belongs to the MIxx column.

CVS [24] and BSCV [25] are two of the most widely used collaborative tools. CVS supports the collaborative development of software code, BSCW supports the collaborative preparation of documents. The concept these environments follow is similar to what we propose for multi-user Grid portals: Let multiple persons contribute to a project simultaneously. In case of a software engineering project multiple developers want to design a complex software system (CVS). In case of a publication project multiple authors want to write a joint paper or a book (BSCW). In case of a Grid project multiple users want to define and execute a complex application on top of the Grid. We believe that in the near future – when Grids will be widely used in both industrial and scientific settings – every Grid-related project will require this kind of support. (See Figure 1).

2.3. The Structure of Collaborative Grid Portals

The classification of Table 1 can be justified by the structure of workflow-oriented Grid portals too. Every workflow-oriented portal consists of a workflow GUI and a workflow manager part, each determining one dimension of Table 1.

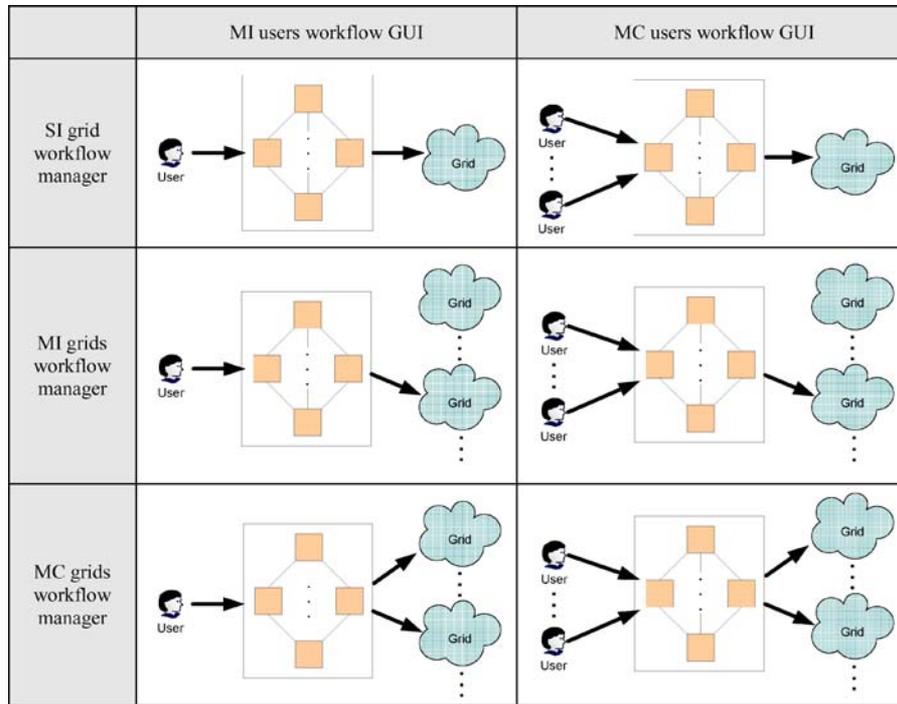
While the workflow GUI is the interface that enables the development, submission and steering of workflows and the visualization of results, the workflow manager is responsible for the execution and scheduling of workflow components in the connected

Grids. If the workflow GUI is able to serve multiple users, but each user should work on separate workflow graphs, then the portal is a MIxx type. If the workflow GUI is able to serve multiple users in a way that they can work on the same graph simultaneously, can visualize and control the execution of the workflow in a collaborative fashion, then the portal is a MCxx type.

Similarly, if the portal workflow manager can use only a single Grid to execute the components of the submitted workflows, then the portal is an xxSI type. If the workflow manager is able to cooperate with multiple Grids but only one Grid can be used at a time to execute workflow components, then the portal is an xxMI type. If the workflow manager can simultaneously utilize multiple Grids when executing different components of a workflow, then the portal is an xxMC type. Table 2 presents the different portal types from the structural point of view.

The introduced classification model can be applied not only for Grid portals, but for traditional Grid user interfaces and PSEs as well. Unicore [27] and Triana [15] are two of the most well-known workflow-oriented PSEs. According to the classification they both belong to the simplest, to the MISI class. They provide neither multi-Grid access, nor collaborative user support. Although the server of Triana is built on top of the GAT API [28] – thus it could abstract the underlying Grid services from their actual implementations – it cannot distinguish security domains from each other, which is a prerequisite of multi-Grid access. (Because different Grids represent different security domains).

Table 2. Classification of workflow-oriented portals from the structural point of view (see also Table 1).



Pegasus [29], a Web-based Grid portal is a MISI environment as well. Based on a special configuration file, filled up by the portal administrator with Globus GRAM and GridFTP [2] site addresses, Pegasus is able to map abstract workflows onto physical resources. At the same time – because of the centrally managed resource list and the single certificate the manager applies during workflow execution – Pegasus cannot be considered a multi-Grid portal.

The GridFlow portal [30] applies a more complex workflow executor subsystem than the above discussed environments. The workflow manager of GridFlow handles workflows at two levels. It manages workflows at a global Grid level and schedules them at the level of different local Grids. It does not provide collaborative development and execution capabilities, so according to our model the GridFlow portal is a MIMC portal implementation.

The remaining part of the paper presents the P-GRADE Portal [44], a Globus-based portal implementation that covers all the six cells of Table 1. While the 2.x version of the P-GRADE Portal fulfills

the requirements of MISI, MIMI and MIMC classes. Its 3.0 version² can operate in all the MISI, MIMI, MIMC, MCSI, MCMI and MCMC modes.

3. The P-GRADE Portal

The P-GRADE Portal is a workflow-oriented Grid portal with the main goal to support all stages of Grid workflow development and execution processes. It enables the graphical design of workflows created from various types of executable components (sequential, MPI or PVM jobs), executing these workflows in Globus-based [2] computational Grids relying on user credentials, and finally, analyzing the monitored trace-data by the built-in visualization facilities. The P-GRADE Portal provides the following functions (see also Figure 2):

1. Defining Grid environments
2. Creation and modification of workflow applications

² P-GRADE Portal 3.0 will be released in 2006.

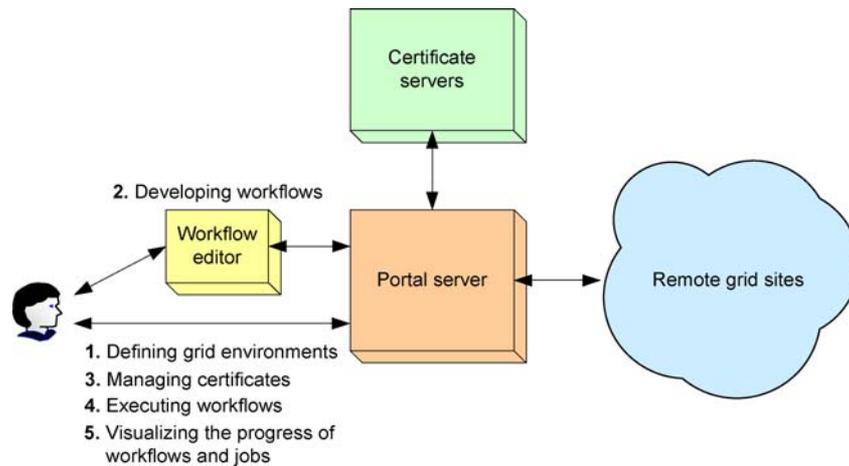


Figure 2. User activities supported by the P-GRADE Portal.

3. Managing Grid certificates
4. Controlling the execution of workflow applications on Grid resources
5. Monitoring and visualizing the progress of workflows and their component jobs³

3.1. Grid Workflows in the P-GRADE Portal

Workflow applications can be developed in the P-GRADE Portal by the graphical Workflow Editor. The Editor is implemented as a Java Web-Start application that can be installed on the client machines ‘on the fly,’ using a standard Web browser. The Editor communicates only with the Portal Server, and it is completely independent from the Grid infrastructure(s) the Server is connected to.

A P-GRADE Portal workflow is an acyclic dependency graph that connects sequential and parallel programs into an interoperating set of jobs. The nodes of such a graph are batch jobs, while the arc connections define data relations among these jobs. Arcs define the execution order of the jobs and the input/output dependencies that must be resolved by the workflow manager during execution. An ultra-short range weather forecast (Nowcast) Grid application (MEANDER) [18] is shown in Figure 3 as an example for a P-GRADE Portal workflow.

Nodes labeled as *delta*, *cummu*, *visib*, *satel* and *ready* represent executable programs of the following

two types: Sequential and parallel MPI. Small squares labeled by numbers around the nodes are called ports and represent input and output data files that the corresponding executables expect or produce. (One port represents one input/output file.) Directed arcs interconnect pairs of input and output ports if an output file of a job serves as an input file for another job. An input file – represented by an input port – can come from three different sources:

1. It can be produced by another job of the workflow. Such input files are represented by input ports that are connected to output ports. During workflow execution the workflow manager transfers such files from the executor site of the producer job to the executor site of the consumer job.
2. From the workflow developer’s desktop machine. Input ports that represent such input files are not connected to any output ports. The files must be uploaded from the desktop machine to the portal server before the submission of the workflow. The Workflow Editor and the Portal Server performs the upload procedure automatically when the workflow is saved by the user. During workflow execution the workflow manager transfers these files from the Portal Server to the appropriate computational resources.
3. From a storage resource. An input port can be a reference to a file located at the resource of a Grid storage provider. The workflow manager transfers such input files from storage resources to computational resources during workflow execution.

³ The application monitoring and visualization capabilities of the P-GRADE Portal are out of the scope of the paper. For more information please refer to [44].

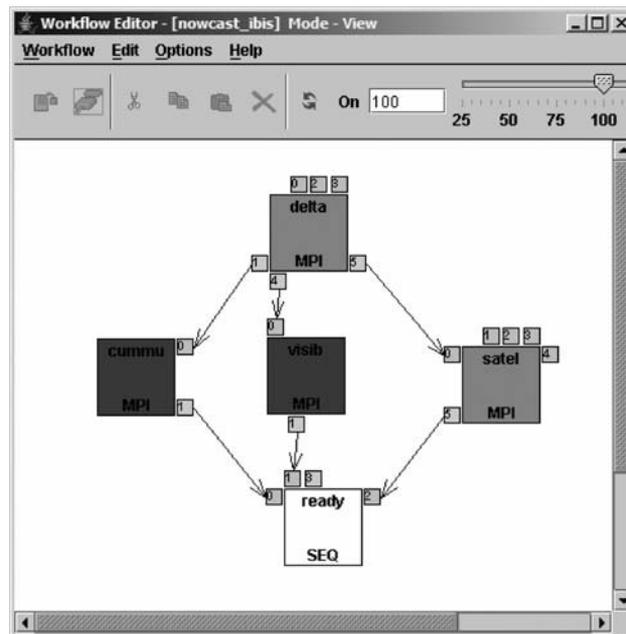


Figure 3. The MEANDER short-range weather forecast meteorology application opened in the Workflow Editor of the P-GRADE Portal.

An output file – represented by an output port – can have the following target locations:

1. A computational resource. If an output port is connected to one or more input ports then the output file represented by that port serves as a data transmitter between a producer and one or more consumer jobs. The workflow manager transfers the file from the executor site of the producer job to the executor site(s) of the consumer job(s).
2. The Portal server. If valuable information is generated by a job into a relatively small file, then the user can instruct the workflow manager to transfer the file back to the Portal Server. After the workflow is terminated such output files can be downloaded from the Portal Server onto the users' desktop machines.
3. A storage resource. If relatively large amount of valuable information is generated by a job into an output file, then the user can instruct the workflow manager to save the file on a Grid storage resource. The users (or other Grid services) can access such files with appropriate clients.

Notice, that an output file can be a data transmitter between jobs and can also contain valuable information for the user. In this case the workflow manager first copies the file from the producer site to every consumer site, then moves the original copy of the file to the Portal Server or to a storage resource.

The semantics of the workflow execution means that a node (job) of the workflow can be executed if, and only if all of its input files are available, i.e., all the jobs that produce input files for this job have successfully terminated, and all the other input files are available on the Portal Server and at the pre-defined storage resources. Therefore, the workflow describes both the control-flow and the data-flow of the application. If all the necessary input files are available for a job, then the workflow manager transfers these files – together with the binary executable – to the computational resource where the job is allocated for execution. Managing the transfer of files and recognition of the availability of the necessary files is the task of the workflow manager component of the Portal Server.

3.2. The Grid-Concept of the P-GRADE Portal

Grid computing enables the flexible, secure, coordinated sharing of resources among dynamic collections of individuals, institutions, and resources – usually referred to as virtual organizations (VO) [41]. To realize VOs first a uniform Grid middleware layer should be built on the top of the resources. This layer hides low level hardware and software differences by high-level standardized service interfaces and protocols. There are several ways to implement the Grid middleware. In fact we can witness the fast evolution

of middleware concepts that means a distracting feature of Grids for end-users. Without a high-level user interface they should re-learn the new versions of middleware services, concepts and commands from time to time, moreover, applications built with the API of one middleware cannot be ported directly onto a different system.

A set of resources, hidden by abstract interfaces still cannot be regarded as a VO. While the middleware creates facility to discover and invoke services on-demand, some high-level policy must define which components and when are allowed to cooperate. Grid middleware only with this access policy can realize VOs.

Globus [2] gives one implementation for the Grid middleware and VO concepts. Most of the current production Grids apply one of the official releases, or some modified version of Globus-2 [20, 21, 31–33, 35]. In these systems the low-level protocols (e.g., GRAM, GridFTP, MDS, BDII) specify the services that clients can apply to access providers, while GSI [26] – independently from these protocols – defines the rules of collaboration. GSI is based on the PKI concept: Every Grid participant (client and service) must have a certificate, signed by a Certificate Authority (CA). Two Globus Grid participant are allowed to cooperate – and are regarded as members of the same VO – if they trust the issuers of each others' certificates. If they do not belong to the same VO they cannot use each other's services even if they speak the same protocol. Consequently, in GSI-based environments it is more important to classify entities according to their VO membership, than according to the Grid they participate in. Even if two providers belong to the same Grid (e.g., LHC Grid) they cannot be accessed with the same certificate if they are members of different VOs⁴ (e.g., Atlas and SEE-GRID VOs). As a result, applications built onto the top of Globus systems have to think in terms of VOs instead of terms of Grids.

The P-GRADE Portal is a workflow-oriented portal implemented on top of Globus. Consequently, the P-GRADE Portal is a xxSI, xxMI, xxMC portal in a sense that it can be used to execute workflows in a single Globus VO, in multiple Globus VOs either in isolated and in simultaneous ways. In the two

latter cases these VOs can be part of the same Grid (e.g., LHC Grid [31]) or part of different Grids (e.g., NGS [21], NorduGrid [32], Grid2003 [33]).

4. Workflow Development

4.1. *Developing and Editing MIxx Workflows*

Users must work on MIxx workflows individually during both the development and execution phases. In the P-GRADE Portal an MIxx workflow can be loaded from the user's private storage space – allocated on the Portal Server – into the client-side Editor, can be edited locally, and the updated version can be loaded back to the Server. The development of a P-GRADE Portal workflow consists of two subtasks:

1. defining the structure of the graph
2. specifying the properties of nodes (jobs and ports)

The graph structure can be defined using the drag and drop GUI elements of the Workflow Editor. The properties of nodes can be specified using property windows: By double clicking a job or a port a corresponding property window can be popped up and the attributes of the affected component can be defined. While the graph definition process is identical for every MIxx workflow, the component definition processes are slightly different for MISI, MIMI and MIMC workflows.

The job component of a workflow node can be defined in the following way: Using the job property window (See Figure 4) the user must specify the client side location and the type (SEQ, MPI or PVM) of the binary executable. Optional start-up parameters can also be given here (e.g., command line attributes). The job can be mapped onto a computational resource in the following way. Using the 'Grid' and 'Resource' dropdown listboxes first a Grid (actually a Globus VO), then a computational resource from that Grid must be chosen. If the portal is a MISI installation than only one VO appears in the 'Grid' listbox. If the portal is a MIMI or MIMC installation, then multiple VOs are on the list. In the MIMI case every job must be mapped onto resources of the same VO, while jobs of MIMC workflows can follow any mapping topology. (Section 4.4 discusses the 'Grid' and 'Resource' lists in more detail.)

⁴ Sometimes multiple VOs trust the same CA and can be accessed with the same certificate. In this case these VOs actually join together into a single VO.

Figure 4 shows a property window for a job component. The fields are: Name: 'satel'; Job Type: 'MPI' (selected); Job Executable: 'workflows/users/nemethcs/howcast-final.gENO.files/satel/satel.g'; Process Number: empty; Attributes: '12 -n -m'; Grid: 'SEE-GRID'; Monitor: unchecked; Resource: 'crogrid1.irb.hr:jobmanager-fork'. There are 'File Browser', 'Instrument', 'Ok', and 'Cancel' buttons.

Figure 4. The property window of a job component of a P-GRADE Portal workflow.

While in P-GRADE Portal 2.1 jobs must be mapped onto computational resources manually, the 2.2 version of the P-GRADE Portal can interface with the broker component of the LHC Grid infrastructure [46], thus it can automatically find the most appropriate resources for jobs.

Figure 5 presents the property window of a port component. The 'Type' field specifies the type of the file the port represents (input or output). For input files the next field ('File type') defines the location of the file: Local means 'the file is on the user's desktop machine,' remote means 'the file is on a Grid storage resource.' For output files the local option means that the file must be saved on the Portal Server, while remote means the file must be saved on a Grid storage resource. The 'File' field specifies the exact source location (for input files) or the exact target location (for output files) of the file. The last field defines whether the output file contains important information for the user (permanent) or it is only used as a data transmitter between jobs (volatile). While permanent files must be saved on the Portal Server or on a Grid storage resource, volatile files must not.

4.2. Developing and Editing MCxx Workflows

With collaborative-user portals (MCxx) multiple persons can work on the same workflow simulta-

neously. During the development phase the developer team can define the structure of the graph and the contents of the different nodes. Every user can add his/her own knowledge and jobs to the graph. The different nodes can be defined by different team members, so one's jobs can produce result files to jobs defined by other team members. The developers do not have to take care how the input files will be generated for their jobs, they have to focus only onto the format and the meaning of the transmitter file contents.

During application development MCxx portals must act as frameworks that enable the concurrent

Figure 5 shows a property window for a port component. The fields are: Port name: '1'; Type: 'In' (selected); File type: 'Local' (selected); File: 'D:\A-TEST\2'; Internal File Name: 'INPUT2'; File storage type: 'Volatile' (selected). There are 'File Browser', 'Ok', and 'Cancel' buttons.

Figure 5. The property window of a port component of a P-GRADE Portal workflow.

engineering of Grid programs. The goal is to let several people contribute to a single workflow simultaneously. It can be achieved in two ways:

1. Associating ‘soft locks’ with workflow components
2. Associating ‘hard locks’ with workflow components

The first approach enables team members to work on any part of an application any time. The concept assumes that the developers previously agreed on the distribution of work, i.e., everybody knows that e.g., user ‘1’ works on ‘branch 4,’ user ‘2’ on ‘job C’ and so on. The developers can check out the workflow application from the Portal Server, can perform their own modifications and can save the updated versions on the server again. During the upload process the portal server merges the different versions together. If the users have adhered to a disjunct distribution of work, then nobody’s contribution is lost during the merging process.

The second approach guarantees exclusive access to components. Users have to lock workflow elements in advance to modification. Once somebody obtained a lock on an item then nobody else can modify it. Although merging different versions of the same application is still required, lost updates are impossible. Because this approach does not assume off-line communication among the users, developers without even knowing each other can contribute to the same graph.

Locking mechanism could be implemented at different levels: Locks could be associated with workflow branches, with workflow components, or even with elementary attributes that describe a component. The 3.0 version of the P-GRADE Portal implements hard locks at the level of workflow nodes. Locks are associated with nodes (a job and ports being connected to it), and users must obtain lock on nodes before they could perform any modification on them. A lock can be acquired using the collaborative version of the Workflow Editor. If nobody else has lock on the requested node then the server sends back an approval message to the Editor, and the Editor turns the node editable. If the item was already locked, then the user has to wait until the owner of the component finishes the work and releases the lock. A new node can be dropped onto the editor canvas at any time, and becomes automatically locked for the user who defined it. However, the node can be

connected to the graph by an arc if both ends of the channel are locked for the same user.

Figure 6 presents a multi-user workflow editing scenario. On the server side the different workflow components are associated with locks. On the client sides the same elements are in editable (sharp) or disabled (light) states. A user is allowed to modify only those components that are locked for him. The server maintains a global version of the workflow and updates it according to the modifications performed at the clients. Users can upload their latest work to the server and download other users’ components any time. In this way any number of users can contribute to and observe a workflow development process.

Beside the additional locking and unlocking actions MCxx Grid applications can be created exactly in the same way as MIxx ones. The result of a multi-user development phase is a collaborative application which contains not only the workflow, but also the name of the users by whom the different components have been defined. This extra information is used by the multi-user workflow manager during the execution phase to estimate whose certificate should be used to execute a given workflow component. (See also Section 5.1).

4.3. *Setting the Grid Environment*

As it has been discussed in Section 4.1, jobs of a P-GRADE Portal 2.1 workflow must be mapped onto computational resources manually. The ‘Grid’ and ‘Resource’ dropdown listboxes on job property windows are provided for this purpose (See Figure 4). Setting the Grid environment is actually the process when the portal administrator and the portal user define entries for these listboxes. Consequently, this action is a prerequisite of workflow development.

The setting process consists of two stages. First, the portal administrator defines Grids (Globus VOs) for the portal. The administrator can associate an information system and any number of default computational resources with every VO.⁵ The information system is used by the portal to obtain and present resource information to the users. The default resour-

⁵ The 2.2 version of the P-GRADE Portal can also associate a broker with every VO. The workflow manager of the Portal contacts these brokers to find suitable resources for workflow jobs. For more information please refer to [44].

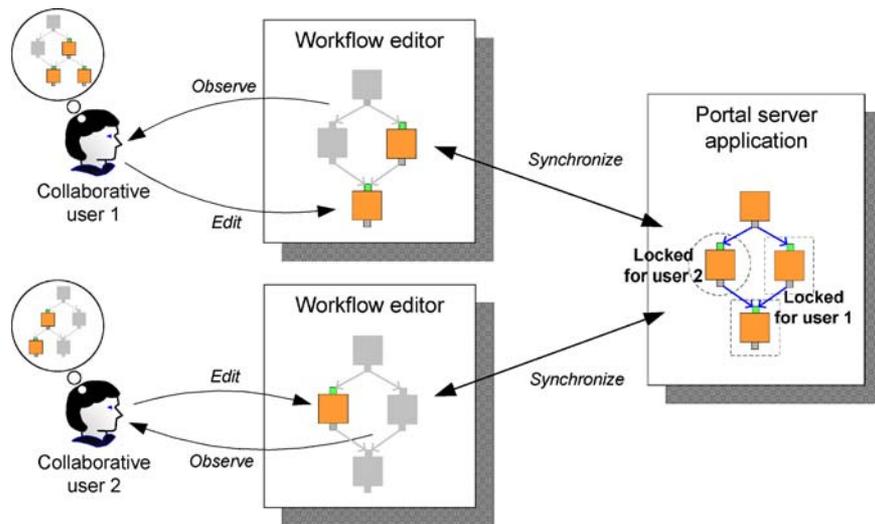


Figure 6. Different workflow components locked by different collaborative users in the P-GRADE Portal.

ces appear in the dropdown listboxes of every portal user. Every user can add further computational resources to his/her own list and can remove resources defined by the administrator from it. There is no way for a user to modify the Grid list either by adding or deleting VOs to and from it. According to the P-GRADE Portal concept a portal installation serves the user community of one or more Globus VOs, so only the portal administrator can modify the VO list. Figure 7 presents the two pages of the ‘Settings’ portlet which can be used by end-users to customize their own resource sets.

As it can be seen in Figure 7.a, three Globus VOs are available for the portal users. Two of these VOs (HUNGRID [42] and SEE-GRID [20]) belong to the LHC Grid [31], while SZTAKI-GRID is an independent VO of the Hungarian Grid research community. Notice, that a VO is presented by its information system. By clicking on the ‘Resources’ button of a VO entry the computational resources being associated with that VO can be browsed (Figure 7.b). This page can be used to define new resources for a VO (Add button) or to delete resources from the VO (Delete button). By clicking on the ‘Load default’ button all the resources defined by the portal administrator will be added to the list.

5. Workflow Execution

Because none of the largest production Grids contain workflow manager services [21, 31–33, 35], work-

flow-oriented portals connected to them must incorporate workflow managers too. The P-GRADE Portal contains a DAGMan-based [22] workflow manager subsystem which is responsible for the scheduling of workflow components in Grids. This section discusses the workflow executor subsystems of the P-GRADE Portal with the aim to present Globus-based implementations for xxSI, xxMI and xxMC workflow managers.

5.1. Grid Certificate Management

A crucial prerequisite of workflow execution is to provide the workflow manager with Grid certificates. These certificates are used by the manager when it performs the elementary operations of the execution process. The execution of a P-GRADE Portal workflow consists of two types of elementary operations: Job execution and file transfer. The task of the P-GRADE Portal workflow manager is to degrade workflow definitions into sequences of these two operations and execute the sequences by invoking services from the connected Grid(s).

Grid certificates (short-term GSI proxy credentials [26]) can be downloaded from MyProxy servers [17] into the P-GRADE Portal (See Figure 2). The ‘certificates’ portlet (See Figure 8) provides a graphical interface to manage and browse the downloaded proxies. New proxies can be downloaded, expired or unnecessary proxies can be deleted here as well. Moreover, if no MyProxy server can provide an appropriate proxy for an execution, then long-term user

The screenshot shows the P-Grade Portal Settings page in Microsoft Internet Explorer. The browser address bar shows the URL: `http://hpportal.hpcc.sztaki.hu:9060/gridsphere/gridsphere?action=doSaveJobManagerConfigurations&id=9`. The page header includes the P-Grade logo and the text "portal". A navigation menu contains "Workflow", "Certificates", "Settings", "Demo", "Help", and "Information System". The main content area is titled "Settings" and contains a section for "GRID configurations".

Name	Type	Information System			[Actions]
		Host	Port	BaseDn	
HUNGRID	LCG2	grid152.kfki.hu	2170	mds-vo-name=local,o=grid	Resources
SEE-GRID	LCG2	grid1.irb.hr	2170	mds-vo-name=HR-RBI,o=grid	Resources
SZTAKI-GRID	MDS2	n0.hpcc.sztaki.hu	2135	Mds-Vo-name=SzuperGRID,o=Grid	Resources

Below the table is a section for "Default visualization size" with input fields for "Width: 600" and "Height: 350", an "OK" button, and a note "(Accept values between 150-1000.)". A message at the bottom states: "Message: Configuration successfully saved."

a) Grid (Globus VOs) set by the administrator of a P-GRADE Portal installation

The screenshot shows the P-Grade Portal Settings page in Microsoft Internet Explorer. The browser address bar shows the URL: `http://hpportal.hpcc.sztaki.hu:9060/gridsphere/gridsphere?action=doEditGridResources&id=8`. The page header includes the P-Grade logo and the text "portal". A navigation menu contains "Workflow", "Certificates", "Settings", "Demo", "Help", and "Information System". The main content area is titled "Settings" and contains a section for "Resources for 'SEE-GRID' GRID".

URL	Job manager	[Actions]
crogrid1.irb.hr	jobmanager-fork	Delete
grid1.netmode.ece.ntua.gr	jobmanager-fork	Delete
n40.hpcc.sztaki.hu	jobmanager-fork	Delete
testbed001.grid.ic.ro	jobmanager-fork	Delete

Below the table are input fields for "URL:" and "Job manager:" with an "Add" button. There are also "Load default" and "Back" buttons. Below this is a section for "Default visualization size" with input fields for "Width: 600" and "Height: 350", an "OK" button, and a note "(Accept values between 150-1000.)". A message at the bottom states: "Message: Resources for SEE-GRID successfully displayed."

b) Resources set for the SEE-GRID VO for the "demo" user

Figure 7. The Settings portlet of the P-GRADE Portal.

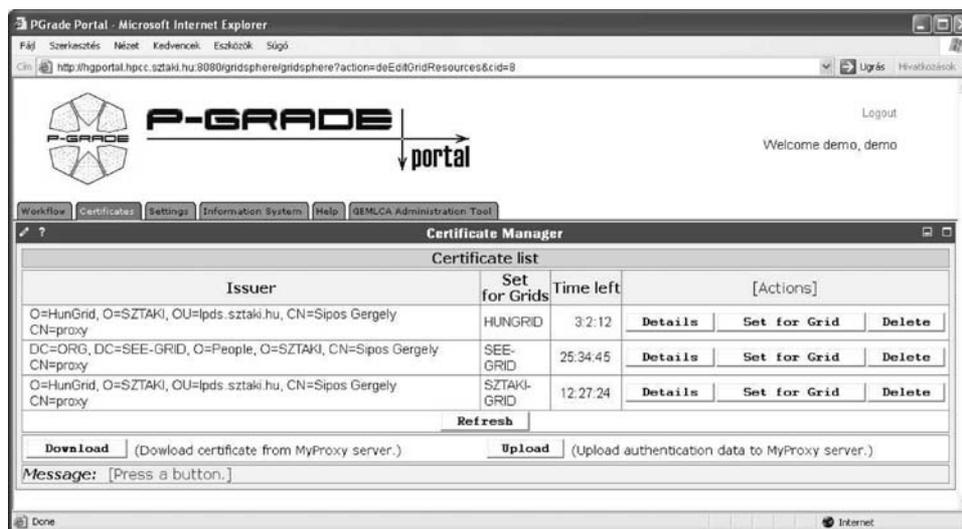


Figure 8. The certificate management portlet of the P-Grade Portal.

certificates can be uploaded into trusted MyProxy servers through this portlet.

Any number of proxies can be available for a user on the Portal Server. As part of the download process each proxy must be associated with one or more Globus VOs. As it has been described in Section 4.3, the list of available Grids is set by the portal administrator. Grids and proxies are in a many-to-one relation: A proxy can belong to multiple VOs, but maximum one proxy can be associated with a VO. As it can be seen in Figure 8, there are three proxies available for the 'demo' user: One for the HUNGRID, one for the SEE-GRID and one for the SZTAKI-GRID VOs. When a workflow is submitted for execution by the 'demo' user, the workflow manager can use these three proxies to access computational and storage resources.

5.1.1. Certificate Management for Mlxx Workflows

Mlxx workflows are developed and executed by a single person. The workflow manager can choose from this person's proxy set to invoke storage and computational services during the execution. In case of a MlSI portal there is only one VO available for the users. Consequently, there can be only one proxy downloaded under the submitter user's portal account, and this proxy identifies the user in the connected VO. The workflow manager applies this proxy to copy files to/from storage and computational resources and to start up jobs on computational resources.

Because the components of a MIMI workflow refer to resources of a single VO, one proxy is enough for the execution. However, since the portal is connected to more than one VOs, multiple proxies can be available under the submitter user's account. Each of these proxies identifies the user in one of the connected VOs. The task of the workflow manager is to choose the proxy that is valid in the VO referred by the workflow definition. Using the proxy-VO association table (see Figure 8) the selection task can be performed by the manager automatically.

Because the components of a MIMC workflow are mapped onto resources of different VOs, multiple proxies are required during the execution. Using the proxy-VO table (See Figure 8) the workflow manager has to choose one suitable proxy for every elementary operation. There are five basic operations during workflow execution that require the invocation of a Grid service, thus require the usage of a proxy. These operations and the applied proxy selection policies are the following:

1. Reading an output file from a computational resource. The manager applies the proxy that is associated with the VO the resource belongs to.
2. Writing an input/executable file onto a computational resource. The manager applies the proxy that is associated with the VO the resource belongs to.
3. Starting up a job on a computational resource. The manager applies the proxy that is associated with the VO the resource belongs to.

4. Reading an input file from a storage resource. The manager applies the proxy that is associated with the VO the consumer job will be executed in.
5. Writing an output file onto a storage resource. The manager applies the proxy that is associated with the VO the producer job was executed in.

(Although reading/writing files from/to the portal server can also occur during the execution, these operations do not require proxies.) An important consequence of policies 4 and 5 is that an input file can be copied from a storage resource to a computational resource only if the two resources are members of the same VO. Moreover, the result file of a job can be saved only on those storage resources that belong to the same VO than the computational resource the job was executed on.

Notice, that during the execution of a MIMC workflow there can be situations when a job executed in VO 'a' produces a file for another job that must be executed in VO 'b.' Such file transfers require different proxies at the source and at the target locations. Because the widely supported Globus GridFTP protocol does not support multiple proxies [23], the workflow manager performs indirect file transfer between the resources, using the Portal Server as a transient storage. (Points 1, and 2 from the previous list specifies the proxies that are used by the manager to read and to write the file from/to the producer/consumer resource.)

5.1.2. Certificate Management for MCxx Workflows

Associating proxies with elementary operations is quite self evident if the workflow belongs to a single person. However, the situation gets much more complicated for collaborative-user workflows. These workflows are owned by groups of users by whom the applications have been designed concurrently and by whom the workflows are submitted and steered. In this case a proxy for a service invocation must be selected in two steps. First, the manager must decide which collaborative user – or users – proxy set(s) to choose from, and second, it must choose a single proxy from this/these proxy set(s).

The user selection process can be based on different policies. For example the manager could take only the submitter user's proxy set into consideration. (The person who clicked the 'submit' button.) In this case one person would be responsible for the invocation of every service during the execution of the workflow. Following another policy the manager

could use the proxy sets of those users who actively contributed to the workflow during the development phase. In this case the responsibility of the execution would be shared among multiple persons. A third workflow manager implementation could select the proxy sets of some dedicated users who have been elected by the collaborative team. Options are endless and all would result reasonable system for slightly different needs. The P-GRADE Portal implements the second policy, thus it applies the active workflow developers' proxy sets to choose proxies for service invocations. The exact policy implemented by the P-GRADE Portal is the following:

1. To read an input file from a storage resource or to write an input (or binary executable) file onto a computational resource the manager chooses a proxy from that user's proxy set who defined the job the file belongs to.
2. To read an output file from a computational resource or to write an output file onto a storage resource the manager chooses a proxy from that user's proxy set who defined the job the file was produced by.
3. To start up a job on a computational resource the manager chooses a proxy from that user's proxy set who defined the job.

Consequence of the policy is that the job and port components of a workflow node must be defined by the same user. This is a logical approach, because in this way collaborative users can add their own jobs with appropriate input and output files to the workflow, and the manager makes every user responsible only for his/her own part during the workflow execution process.

After the manager selected the appropriate user's proxy set it must select a single proxy from it. At this point the same policy can be applied that has been introduced in Section 5.1.1 for MIMC workflows.

Transferring files between VOs is also required during the execution of an MCMC workflow. Just like in the MIMC case such files are transferred using different proxies at the source and at the target locations, copying the data through the Portal Server indirectly.

5.2. Workflow Management

One of the main goals of the P-GRADE Portal is to hide the low level details of Grid systems with high-

level, technology-neutral interfaces that can be easily integrated with different middleware. The GUI of the Portal is built with the GridSphere portal framework [43], thus the various portal functions are implemented as nearly independent portlets. These portlets can be seen in Figure 9.

The ‘Certificate manager’ portlet is responsible for uploading X.509 certificates into MyProxy servers and for downloading short-term GSI proxies into the workflow manager (See Section 5.1). The ‘Settings’ portlet can be used to specify Globus VOs and computational resources for the portal application (See Section 4.3). The ‘Workflow’ portlet is the graphical interface of the workflow manager and can be used to submit and control workflows, to monitor and visualize execution.

In order to achieve the high portability of the portal among different Grids, we had to define the basic set of Grid services the Portal Server can be built on. A natural starting point was the Globus Toolkit and particularly those tools of Globus-2 [2] that are generally accepted and widely used by production Grids today. Globus GRAM, GridFTP and GSI have been selected as the minimal underlying toolset for the P-GRADE Portal.

GRAM is the job management interface that can be used to start up batch jobs on computational resources. It acts as a gateway to access local jobmanagers, and because it does not restrict them, practically any local jobmanager – such as Condor

[22], SGE [38] or LSF [39] – can be accessed by connected into P-GRADE Portal Grids. GridFTP [23] is the file management interface of computational and storage resources. In our architecture GridFTP is used to read and write files from/to computational and storage resources. GSI [26] is the security infrastructure that protects GRAM and GridFTP servers from unauthorized clients. The choice of this infrastructure was justified by connecting the P-GRADE Portal to several Globus-based Grid systems such as the UK NGS [21] the LHC Grid [31] and the GridLab testbed [40]. Notice, that the actual Grid system can have much more services than the P-GRADE Portal needs. The point is that if this basic middleware set is available, then the P-GRADE Portal can be immediately used on that particular Grid.

As it can be seen in Figure 8 the ‘Workflow’ portlet is interfaced with the Condor DAGMan [22] workflow scheduler. DAGMan degrades workflows into elementary file transfer and job submission tasks and schedules the execution of these tasks. Although DAGMan itself cannot invoke Grid services, it supports customized Grid service invocations by its pre/post script concept [22]. One pre and one post script can be attached to every node of a DAGMan workflow. DAGMan guarantees, that it first executes the pre script, then the actual content script and finally the post script when it reaches a workflow node. Consequently, the Portal Server automatically

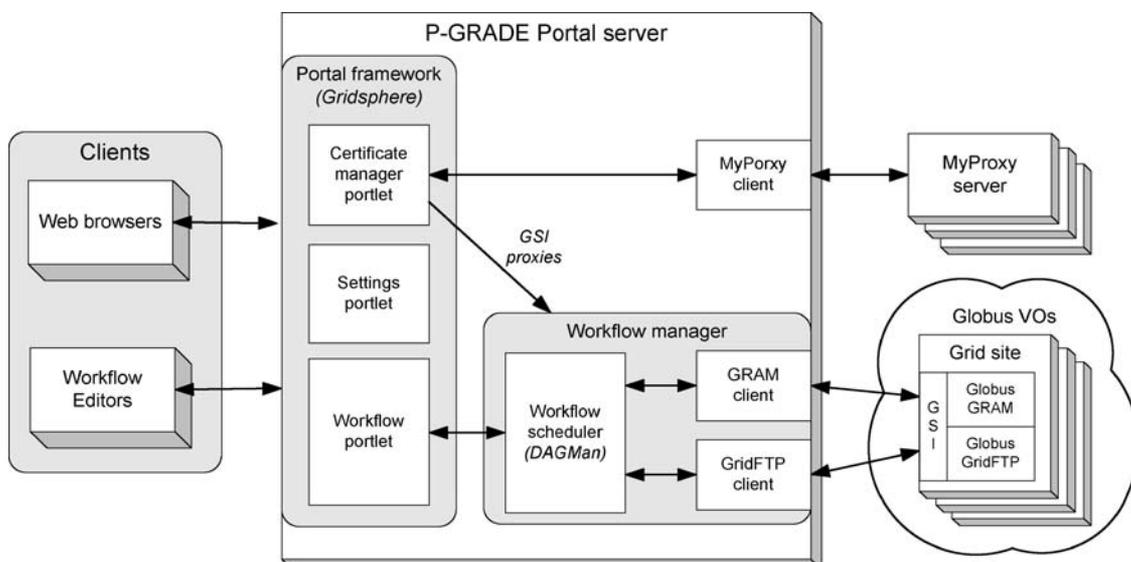


Figure 9. The structure of the P-GRADE Portal.

generates appropriate pre, content and post scripts for every workflow node when the workflow is saved on the server. These scripts – started by DAGMan according to the graph structure – invokes the GridFTP and GRAM clients (See Figure 9) to access files and start up jobs in the connected Grids. DAGMan invokes these scripts in the same way in both single- and multi-Grid configuration. However, as it has been described in Section 5.1, proxies must be selected differently in the various cases. Selecting proxies according to the introduced policies is the task of the GRAM and GridFTP clients.

6. Summary and Conclusions

The paper proposed a model to classify workflow-oriented Grid portals. The classification is based on two parameters: The number of clients that can collaboratively participate in the development and execution of applications and the number of Grids that can be applied to execute these applications. The result is a set of six portal classes. Although a more complex classification taxonomy is given for workflow management systems by Yu and Buyya in [47], their model does not take the discussed two dimensions into consideration. We believe that as the numbers of production Grids and Grid users increase, these two parameters will be highly important.

With the most advanced portals multiple users can work together to define and execute Grid applications that utilize resources of multiple Grids. By connecting previously separated Grids and previously isolated users together, these portals will revolutionize multidisciplinary research.

The P-GRADE Portal gives a Globus-based implementation for the collaborative-Grid, collaborative-user concept. Since almost every production Grid uses Globus middleware today, these Grids could all be accessed by the P-GRADE Portal. Moreover, due to the multi-Grid concept a single portal installation can serve user communities of multiple Grids. These users can define workflows using the high-level graphical notations of the Workflow Editor, can manage certificates, workflows and jobs through the Web-based interface of the Portal Server. They can harness resources from multiple VOs

and can migrate applications among Grids without learning new technologies or interfaces.

P-GRADE Portal 2.1 has been already connected to several European Grids (LHC Grid [31], EU GridLab testbed [40], UK OGSA test-bed [19], UK NGS [21]) and serves as a graphical interface for several production Grids like SEE-GRID [20], HunGrid [42] and UK NGS [21]. On top of that its latest, 2.2 version is connected to the broker, storage and DBII components of the LCG middleware [44].

Acknowledgments

The authors would like to express their appreciation to Vassil N. Alexandrov and Gareth J. Lewis for their valuable help in defining the collaborative-user version of the P-GRADE Portal. The authors are also grateful to the Grid research team of the University of Westminster for their help in connecting the P-GRADE Portal to the UK OGSA test-bed and to the UK NGS. Many thanks are due to the SEE-GRID project members who promoted the P-GRADE Portal as the official portal of SEE-GRID. Finally, special thanks to the P-GRADE Portal team for their excellent work making the Portal a real service for several production Grids all over Europe.

References

1. I. Foster and C. Kesselman, "Computational Grids", *The Grid: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, pp. 15–52, 1998.
2. I. Foster and C. Kesselman, "The Globus project: A status report", in *Proc. of the Heterogeneous Computing Workshop*, IEEE Computer Society, pp. 4–18, 1998.
3. E. Korpela, D. Werthimer, D.P. Anderson, J. Cobb and M. Lebofsky, "SETI@home – Massively distributed computing for SETI", *IEEE Computational Science and Engineering*, Vol. 3, No. 1, pp. 78–83, 2001.
4. A.A. Chien, "Architecture of a commercial enterprise desktop Grid: The Entropia system", *Grid Computing – Making the Global Infrastructure a Reality*, Wiley, Ltd., pp. 337–350, 2003.
5. M. Snir, S.W. Otto, S. Huss-Lederman, D.W. Walker and J. Dongarra, "MPI: The Complete Reference", MIT, 1995.
6. N. Karonis, B. Toonen, and I. Foster, "MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface",

- Journal of Parallel and Distributed Computing*, Vol. 63, No. 5, pp. 551–563, 2003.
7. I. Foster, M. Fidler, A. Roy, V. Sander and L. Winkler, “End-to-end quality of service for high-end applications”, *Elsevier Computer Communications Journal*, Vol. 27, No. 14, pp. 1375–1388, 2004.
 8. Ewa Deelman, et al. “Mapping abstract complex workflows onto Grid environments”, *Journal of Grid Computing*, Vol. 1, No. 1, pp. 25–39, 2003.
 9. I. Taylor et al., “Triana applications within Grid computing and peer to peer environments”, *Journal of Grid Computing*, Vol. 1, No. 2, pp. 199–217, 2003.
 10. Matthew Addis et al., “Experiences with eScience workflow specification and enactment in bioinformatics”, in *Proc. of UK e-Science All Hands Meeting*, S.J. Cox, Editor. 2003.
 11. R. Barbera, A. Falzone and A. Rodolico, “The GENIUS Grid portal”, *Computing in High Energy and Nuclear Physics*, 2003.
 12. UK NGS Portal: <http://portal.ngs.ac.uk>.
 13. M. Thomas, S. Mock and J. Boisseau, “Development of Web Toolkits for computational Science Portals: The NPACI HotPage”, in *Proc. of the 9th IEEE International Symposium on High Performance Distributed Computing (HPDC 2000)*, Pittsburgh, 2000.
 14. R. Barbera et al., “The GENIUS Grid portal”, *Computing in High Energy and Nuclear Physics*, 2003.
 15. I. Taylor et al., “Grid enabling applications using Triana”, Workshop on Grid Applications and Programming Tools, Seattle, 2003.
 16. M.J. Flynn, “Some computer organizations and their effectiveness”, *IEEE Transactions on Computers*, C-21, 948–960, 1972.
 17. J. Novotny, S. Tuecke and V. Welch, “An online credential repository for the Grid: MyProxy”, in *Proc. of 10th IEEE International Symposium on High Performance Distributed Computing*, 2001.
 18. R. Lovas, G. Dózsa, P. Kacsuk, N. Podhorszki and D. Drótos, “Workflow support for complex Grid applications: Integrated and portal solutions”, in *Proc. of Second European AcrossGrids Conference*, (AxGrids 2004), Nicosia, Cyprus, LNCS, Vol. 3165, 2004, pp. 129–138.
 19. UK e-Science OGSa Testbed: <http://dsg.port.ac.uk/projects/ogsa-testbed/>.
 20. Southern Eastern European GRid-enabled Infrastructure Development (SEE-GRID): <http://www.see-Grid.org/>.
 21. UK National Grid Service: <http://www.ngs.ac.uk/>.
 22. D. Thain, T. Tannenbaum and M. Livny, “Distributed computing in practice: The Condor experience”, *Concurrency and Computation: Practice and Experience*, pp. 323–356, 2005.
 23. W. Allcock et al., “Gridftp protocol specification”, *GGF GridFTP Working Group Document*, September 2002.
 24. B. Berliner: CVS II: ParalMizing software development, in *Proc. of Winter 1990 USENIX Technical Conference*, 1990.
 25. W. Appelt, “WWW based collaboration with the BSCW system”, in *Proc. of the Conference on Current Trends in Theory and Informatics*, pp. 66–78, 1999.
 26. R. Butler, D. Engert, I. Foster, C. Kesselman, S. Tuecke, J. Volmer and V. Welch, A national-scale authentication infrastructure. *IEEE Computer*, Vol. 33, No. 12, 60–66, 2000.
 27. D.W. Erwin and D.F. Snelling, “UNICORE: A Grid computing environment”, In *Lecture Notes in Computer Science*, volume 2150, Springer, pp. 825–834, 2001.
 28. G. Allen et al., “Enabling applications on the Grid: A Gridlab overview”, *International Journal of High Performance Computing Applications*, Issue 17, pp. 449–466, 2003.
 29. G. Singh et al., “The Pegasus portal: Web based Grid computing”, in *Proc. Of 20th Annual ACM Symposium on Applied Computing*, Santa Fe, New Mexico, 2005.
 30. J. Cao, S.A. Jarvis, S. Saini, and G.R. Nudd, “GridFlow: Workflow management for Grid computing”, in *Proc. of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03)*, pp. 198–205, 2003.
 31. LHC Computing Grid: <http://lcg.web.cern.ch/LCG/>.
 32. M. Ellert et al., “The NorduGrid project: Using Globus toolkit for building Grid infrastructure,” *Nuclear Instruments and Methods (Section A)* 502, pp. 407–410, 2003.
 33. Grid2003 Team, “The Grid2003 production Grid: Principles and practice”, in *Proc. of the 13th IEEE Intl. Symposium on High Performance Distributed Computing (HPDC-13)*, 2004.
 34. V. Sunderam and J. Dongarra, “PVM: A framework for parallel distributed computing”, *Concurrency: Practice and Experience*, Vol. 2, No. 4, pp. 315–339, 1990.
 35. B. Coghlan, J. Walsh and D. O’Callaghan, “The Grid-Ireland deployment architecture”, in *Proc. Of European Grid Conference*, Amsterdam, The Netherlands, pp. 354–363, 2005.
 36. R. Allan et al., “Virtual research environments: Sakai demonstrator”, in *Proc. Of UK e-Science All Hands Meeting*, 2005.
 37. L. Childers, T. Disz, R. Olson, M.E. Papka, R. Stevens and T. Udeshi, “Access Grid: Immersive group-to-group collaborative visualization,” in *Proc. of the 4th International Immersive Projection Technology Workshop*, 2000.
 38. W. Gantzsch, “Sun Grid engine, towards creating a compute power Grid”, in *Proc. Of the 1st International Symposium on Cluster Computing and the Grid (CCGrid '01)*, IEEE Computer Society, pp. 35–6, 2001.
 39. S. Zhou, “LSF: Load sharing in large-scale Heterogeneous distributed systems”, in *Proc. Of Workshop on Cluster Computing*, 1992.
 40. E. Seidel, G. Allen, A. Merzky and J. Nabrzyski, “GridLab – A Grid application toolkit and testbed”, *Future Generation Computer Systems*, Vol. 18, No. 8, pp. 1143–1153, 2002.

41. I. Foster, C. Kesselman and S. Tuecke, "The anatomy of the Grid: Enabling scalable virtual organizations", *International Journal of Supercomputer Applications*, Vol. 15, No. 3, 2001.
42. The HunGrid Virtual Organisation: [http://www.lcg.kfki.hu/?hunGrid and hunGridgeneral](http://www.lcg.kfki.hu/?hunGrid%20and%20hunGridgeneral).
43. J. Novotny, M. Russell and O. Wehrens: Grid-sphere: A portal framework for building collaborations" in *Proc. of the 1st International Workshop on Middleware in Grid Computing*, Rio de Janeiro, Brazil, 2003.
44. P-GRADE Grid Portal: <http://www.lpds.sztaki.hu/pgportal>.
45. D. Britton et al., "GridPP: Meeting the particle physics computing challenge", in *Proc. Of UK e-Science All Hands Meeting*, 2005.
46. LCG-2 User Guide, 4 August, 2005: <https://edms.cern.ch/file/454439/2/LCG-2-UserGuide.html>.
47. J. Yu and R. Buyya, "A Taxonomy of Workflow Management Systems for Grid Computing", *Journal of Grid Computing*, Vol. 3, No. 3-4, 2006.