Integration of GEMLCA and the P-GRADE portal

Kiss T, Sipos G, Kacsuk PK, Karoczkai K, Terstyanszky G and Delaitre T

Integration of GEMLCA and the P-GRADE Portal

T. Kiss[1], G. Sipos[2], P. Kacsuk[2], K. Karoczkai[2], G. Terstyanszky[1], T. Delaitre[1]
[1]*Centre of Parallel Computing,Cavendish School of Computer Science,*
University of Westminster, 115 New Cavendish Street, London W1W 6UW,
[2]*MTA SZTAKI Laboratory of Parallel and Distributed Systems*
H-1518 Budapest, P.O. Box 63, Hungary

1. Introduction

There are many efforts all over the world to provide new Grid middleware concepts for constructing large production Grids. As a result, the Grid community is in the phase of producing third generation Grid systems that are represented by the OGSA (Open Grid Services Architecture) and WSRF (Web Services Resource Framework) specifications. On the other hand relatively little attention has been paid to how end-users can survive in the rapidly changing world of Grid generations. Moreover, the efforts in this field remained isolated resulting only in limited functionality prototypes for specific user domains and not serving a wider user community.
This paper describes how the integration of two different architectures, the P-GRADE Grid portal [1] and GEMLCA (Grid Execution Management for Legacy Code Architecture) [2] resulted in a more generic solution serving a large variety of Grid systems and application domains. The integrated solution provides a high-level user-friendly Grid application environment that supports users of GT2-based second generation and GT3/GT4-based third generation Grid systems from the same user interface. It also allows the integration of legacy code applications into complex Grid workflows which can be mapped to Grid nodes running this wide variety of middleware.
The integration has happened at different levels. In the first step, the GEMLCA clients were added to the portal providing a user-friendly interface for legacy code deployment, execution and visualisation. On the other hand this integration also enhanced the usability of the originally GT2-based P-GRADE portal making it capable to handle GT3/GT4 Grid services. In the second step, GEMLCA was extended to handle legacy code submission to current GT2-based production Grids, like the UG National Grid Service or the EGEE Grid. This resulted in a legacy code repository that makes it even easier to P-GRADE portal end users to create and execute workflows from previously published legacy components. Finally, the research teams are currently working on a more loosely coupled integration that will allow incorporating advanced functionality, like GEMLCA-based legacy code support, into the portal as a plug-in, resulting in more flexible solution depending on actual user requirements.
The paper introduces GEMLCA and the P-GRADE portal and describes the integration activities outlined above.

2. Baseline technologies

2.1 P-GRADE Portal

The P-GRADE portal [1] is a workflow-oriented Grid portal with the main goal to cover the whole lifecycle of workflow-oriented computational grid applications. It enables the graphical development of workflows consisting of various types of executable components (sequential, MPI or PVM programs), executing these

workflows in Globus-based grids [4] relying on user credentials, and finally analyzing the correctness and performance of applications by the built-in visualization facilities. Workflow applications can be developed in the P-GRADE portal by its graphical Workflow Editor.

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 jobs, while the arc connections define the execution order of the jobs and the data dependencies between them that must be resolved by the workflow manager during the execution. An example for P-GRADE portal workflows can be seen in the middle part of Figure 2. Large rectangles represent jobs while small rectangles around the jobs are called ports and represent data files that the corresponding jobs expect or produce. Directed arcs interconnect pairs of input and output files if an output file of a job serves as an input file for another job.

The semantics of the workflow execution means that a job (a node 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 user-defined input files are available either on the portal server and at the pre-defined grid storage resources. Therefore, the workflow describes both the control-flow and the data-flow of the application.

Managing the transfer of files and recognition of the availability of the necessary files is the task of the workflow manager portal subsystem, currently implemented on the top of Condor DAGMan [3]. The workflow manager is capable to transfer data among Globus VOs [4], thus the different components of the same workflow can be mapped onto different Globus VOs. These VOs can be part of the same grid, or can belong to multiple grids.

2.2 GEMLCA

GEMLCA represents a general architecture for deploying legacy applications as Grid services without re-engineering the code or even requiring access to the source files. The high-level GEMLCA conceptual architecture is represented on Figure 1.

As shown in the figure, there are four basic components in the architecture:

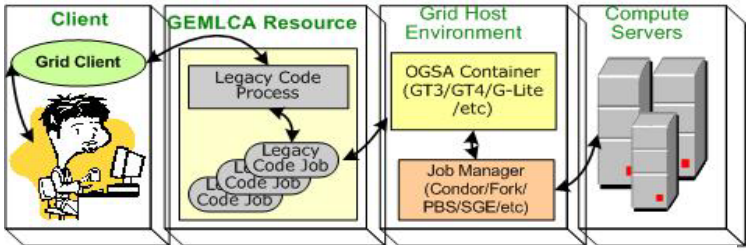

Figure 1 GEMLCA conceptual architecture

1) The **Compute Server** is a single or multiple processor computing system on which several legacy codes are already implemented and available. The goal of GEMLCA is to turn these legacy codes into Grid services that can be accessed by Grid users.
2) The **Grid Host Environment** implements a service-oriented OGSA-based Grid layer, such as GT3 or GT4. This layer is a pre-requisite for connecting the Compute Server into an OGSA-built Grid.
3) The **GEMLCA Resource** layer provides a set of Grid services which expose legacy codes as Grid services.
4) The fourth component is the **GEMLCA Client** that can be installed on any client

machine through which a user would like to access the GEMLCA resources.

The deployment of a GEMLCA legacy code service assumes that the legacy application runs in its native environment on a Compute Server. It is the task of the GEMLCA Resource layer to present the legacy application as a Grid service to the user, to communicate with the Grid client and to hide the legacy nature of the application. The deployment process of a GEMLCA legacy code service requires only a user-level understanding of the legacy application, i.e., to know what the parameters of the legacy code are and what kind of environment is needed to run the code (e.g. multiprocessor environment with 'n' processors). The deployment defines the execution environment and the parameter set for the legacy application in an XML-based Legacy Code Interface Description (LCID) file that should be stored in a pre-defined location. This file is used by the GEMLCA Resource layer to handle the legacy application as a Grid service.

3. Integrating GEMLCA and the P-GRADE portal

GEMLCA provides the capability to convert legacy codes into Grid services just by describing the legacy parameters and environment values. However, an end-user without specialist computing skills still requires a user-friendly Web interface to access the GEMLCA functionalities: to deploy, execute and retrieve results from legacy applications. The P-GRADE portal offers these functionalities besides other capabilities like Grid certificate management, workflow creation, execution visualization and monitoring. This section describes how the integration enhanced the functionalities of both environments and how this integration can be even more effective in the future.

3.1 Extending the P-GRADE portal towards service oriented Grids

The P-GRADE portal supported only GT2 based Grids, originally. On the other hand, GEMLCA aims to expose legacy applications as GT3/GT4 Grid services. The integration of GEMLCA and the portal extended the GT2-based P-GRADE portal towards service oriented Grids. Users can still utilise GT2 resources through traditional job submission, and can also use GT3/GT4 resources by including GEMLCA legacy code services in their workflows. The generic architecture of the

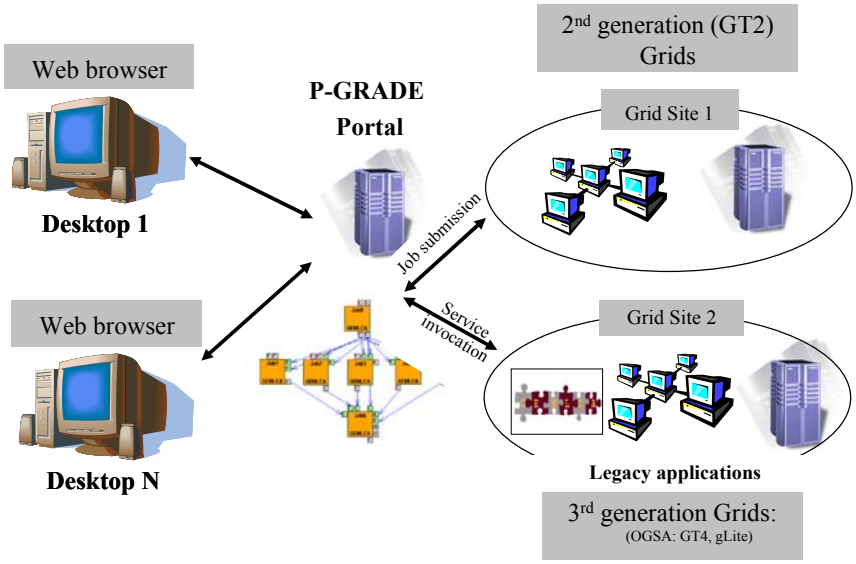

Figure 2 GEMLCA extending the P-GRADE portal towards service oriented Grids

GEMLCA – P-GRADE portal Grid is shown on figure 2.
Integrating the P-GRADE portal with GEMLCA required several modifications in the P-GRADE portal. These are as follows:

1. In the original P-GRADE portal a workflow component can be a sequential or MPI program. The portal was modified in order to include legacy code Grid services as GEMLCA components into the workflow.
2. The *Job properties window* of the P-GRADE portal was changed in order to extend it with the necessary legacy code support. The user can select a GEMLCA Grid resource from drop-down list. Once the Grid resource is selected the portal retrieves the list of legacy code services available on the selected Grid resource. Next, the user can choose a legacy code service from this list. Once the legacy code service is selected the portal fetches the parameter list belonging to the selected legacy code service with default parameter values. The user can either keep these values or modify them.
3. The P-GRADE portal was extended with the GEMLCA Administration Portlet that hides the syntax and structure of the LCID file from users. After filling in a simple Web form the LCID file is created automatically and uploaded by the portal to the appropriate directory of the GEMLCA resource.

After these modifications in the portal, end-users can easily construct workflow applications built from both GT2 jobs and legacy code services, and can map their execution to different Grid resources, as shown in Figure 3.

3.2 Legacy code repository for production Grids

Creating a workflow in the P-GRADE portal requires the user to define input and output parameters and identify ports. For the owner of the code this task is not too complex. However, if another end-user wants to use the same code in a workflow the process have to be repeated by a user who has no deeper understanding of the code itself. In order to help these end-users a legacy code repository based on GEMLCA was created that can be connected to GT2 based production Grid services, like the UK National Grid Service (NGS). The GEMLCA repository enables code owners to publish their applications as GEMLCA legacy codes in the same way as it was described in section 3.1. After this publication other authorised users can browse the repository and simply select the application they would like to use. Parameter values

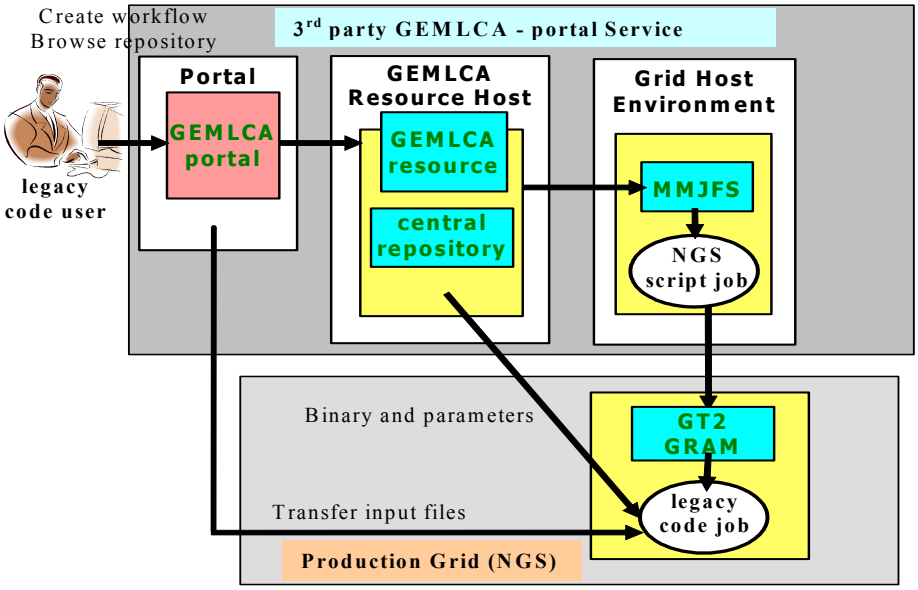

Figure 3 GEMLCA and P-GRADE portal for GT2 based production Grids

can be set by the user. However, there is no need to define parameter types or input/output ports as these are created automatically by the portal based on the GEMLCA description.
The P-GRADE portal extended with the GEMLA repository has been successfully implemented and offered for UK NGS users as a third party service. This implementation of the integrated GEMLCA – P-GRADE portal solution extends the capabilities of both tools. On one hand, GEMLCA is now capable of working with GT2 based Grids by submitting the legacy executables as jobs to the remote GT2 gatekeeper. On the other hand, the usability of the P-GRADE portal has also been enhanced by making it much easier for end-users to create workflows using legacy codes published in the repository. The integrated GEMLCA – P-GRADE portal solution for GT2 based production Grids is shown of figure 3.

3.3 GEMLCA as a plug-in in the P-GRADE portal

Both GEMLCA and the P-GRADE portal are continuously developing products. In order to present their latest features in a uniform way, the two systems must be integrated into a single software from time to time. Although the integration could be done with reasonable effort in case of the first few GEMLCA and portal releases, the task became quite cumbersome with the appearance of additional features and grid middleware technologies.
The P-GRADE portal and the GEMLCA developer teams elaborated a new concept to handle this issue. According to the idea the portal will be transformed into a component-based environment that can be extended with dynamically downloadable plug-ins on the fly. The approach exploits the fact that GEMLCA and P-GRADE jobs are quite similar to each other. They both consist of a binary executable and 1-N input files, a set of input parameters, a resource that has been selected for the execution, etc. The only difference is that the executable and the input files of a GEMLCA job can come from different users, the executable and the input files of a P-GRADE job must be provided by the same party. Nevertheless, this difference can be hidden from the higher software layers during execution by the workflow manager subsystem (an intelligent script can transfer executable and input files to the executor site uniformly, even if those files are coming from different parties), due to the different concept GEMLCA and P-GRADE job developers must be provided with different user interfaces. (As it was described in Section 3.1 other types of parameters must be set on the "Properties" windows for a GEMLCA and for a P-GRADE job.)
While job property windows are hard coded parts of the current version of the workflow editor, the new plug-in based concept enables the dynamic download of these graphical elements, making the P-GRADE and GEMLCA integration task

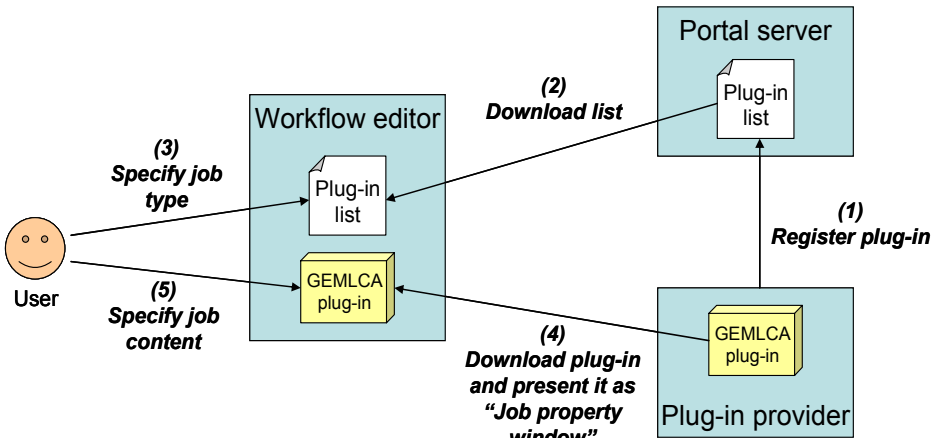

Figure 4 The plug-in based P-GRADE portal workflow editor

almost self evident. The plug-in based editor works in the following way: (See also Figure 4) When the user drops a new job onto the editor canvas and opens its property window, the editor obtains a list of the currently available job types from the portal server (2). Each entry of this list consists of a name (e.g. GEMLCA job) and a reference pointing to a dynamically downloadable plug-in. In subject to the choice of the user (3) the editor downloads the plug-in that belongs to the selected job type and presents it on its GUI as the content of the job property window (4). Since each plug-in can encapsulate customised graphical elements to support the development of one specific type of job (5), no property windows must be hard coded into the workflow editor any more.
Because the workflow editor is a Web Start application plug-ins can be implemented as Java objects. While the list of plug-ins can be retrieved from the portal server using a text format (e.g. an XML message), a binary protocol is needed between the editor and the plug-in provider to make object transmission possible. Java Web Services [6], Jini services [5], EJBs [7] or RMI servers [8] are all suitable for this purpose, thus the plug-in provider can be implemented with any of these technologies. Besides GEMLCA jobs other types of computational jobs can be plugged into the portal in this way. The plug-in provider has to only register its service at the portal server (1).

4. Conclusions and future work

With the integration of the GEMLCA and the P-GRADE portal tools a complex environment has been created that provides solution for a wide range of grid-related problems. Using the integrated system Grid users can deploy legacy and newly developed sequential and parallel applications as software services. They can test and then share these services with a larger community. The members of the community can develop workflows that connect their own computational tasks and the pre-deployed services into an acyclic graph. These workflows can be submitted into Globus-2, 3 and 4 based Grids, can be monitored in a real-time fashion, moreover, the different components can be executed in different Globus VOs.
As the next step, the P-GRADE portal will be transformed into a plug-in based computational framework. With the plug-in concept the portal will be able to support other types of computational services than GEMLCA, without modifying any source code or even stopping the portal server.

References

[1] G. Sipos and P. Kacsuk: Classification and Implementations of Workflow-Oriented Grid Portals, To appear in the Proc. of The 2005 International Conference on High Performance Computing and Communications (HPCC2005), Sorrento, Italy
[2] T. Delaittre, T. Kiss, A. Goyeneche, G. Terstyanszky, S.Winter, P. Kacsuk: GEMLCA: Running Legacy Code Applications as Grid Services, To appear in "Journal of Grid Computing" Vol. 3. No. 1.
[3] T. Tannenbaum, D. Wright, K. Miller, and M. Livny: Condor - A Distributed Job Scheduler. Beowulf Cluster Computing with Linux, The MIT Press, MA, USA, 2002.
[4] I. Foster, C. Kesselman: Globus: A Toolkit-Based Grid Architecture, In I. Foster, C. Kesselmann (eds.) „The Grid: Blueprint for a New Computing Infrastructure", Morgan Kaufmann, 1999, pp. 259-278.
[5] J. Waldo. The Jini architecture for network-centric computing.*Communications of the ACM*, 42(10):76–82, Oct. 1999.
[6] D. Chappell and T. Jewell, "Java Web Services", O'Reilly Press, 2002.
[7] Sun Microsystems: Enterprise JavaBeans: http://java.sun.com
[8] Troy Bryan Downing. *Java RMI: Remote Method Invocation*. Number 0764580434. IDG Books, 1998.